KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION-V3*

SKRIPSI

Disusun Oleh

MUJIIBUR-RAHMAN KAPA

H72217034

PROGRAM STUDI MATEMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN AMPEL

SURABAYA

2022

PERNYATAAN KEASLIAN

Saya yang bertanda tangan di bawah ini,

Nama	:	MUJIIBUR-RAHMAN KAPA
NIM	:	H72217034
Program Studi	:	Matematika
Angkatan	:	2017

Menyatakan bahwa saya tidak melakukan plagiat dalam penulisan skripsi saya yang berjudul " KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION-V3* ". Apabila suatu saat nanti terbukti saya melakukan tindakan plagiat, maka saya bersedia menerima sanksi yang telah ditetapkan.

Demikian pernyataan keaslian ini saya buat dengan sebenar-benarnya.

Surabaya, 18 Februari 2022

Yang menyatakan,

MUJIIBUR-RAHMAN KAPA

NIM. H72217034

LEMBAR PERSETUJUAN PEMBIMBING

Skripsi oleh

Nama	:	MUJIIBUR-RAHMAN KAPA
NIM	:	H72217034
Judul Skripsi	:	KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION-V3*

telah diperiksa dan disetujui untuk diujikan.

Pembimbing I

Nurissaidah Ulmnuha, M. Kom
NIP. 199011022014032004

Pembimbing II

Lutfi Hakim, M.Ag
NIP. 197312252006041001

Mengetahui,
Ketua Program Studi Matematika
UIN Sunan Ampel Surabaya

Aris Fanani, M.Kom
NIP. 197312252006041001

PENGESAHAN TIM PENGUJI SKRIPSI

Skripsi oleh

Nama	:	MUJIIBUR-RAHMAN KAPA
NIM	:	H72217034
Judul Skripsi	:	KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION-V3*

Telah dipertahankan di depan Tim Penguji
pada tanggal 04 Februari 2022

Mengesahkan,
Tim Penguji

Penguji I

Nurissaidah Ulinnuha, M. Kom
NIP. 199011022014032004

Penguji II

Lutfi Hakim, M.Ag
NIP. 197312252006041001

Penguji III

Aris Fanani, M.Kom
NIP. 198701272014031002

Penguji IV

Dian C. Rini Novitasari, M. Kom
NIP. 198511242014032001

Mengetahui,
Dekan Fakultas Sains dan Teknologi
UIN Sunan Ampel Surabaya

Dr. Hj. Evi Hariyatur Rusydiyah, M. Ag
NIP. 196312272005012003

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA
PERPUSTAKAAN
Jl. Jend. A. Yani 117 Surabaya 60237 Telp. 031-8431972 Fax.031-8413300
E-Mail: perpus@uinsby.ac.id

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI
KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika UIN Sunan Ampel Surabaya, yang bertanda tangan di bawah ini, saya:

Nama : MUJIIBUR-RAHMAN KAPA
NIM : H72217034
Fakultas/Jurusan : SAINTEK / MATEMATIKA
E-mail address : mark.rahman22@gmail.com

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Perpustakaan UIN Sunan Ampel Surabaya, Hak Bebas Royalti Non-Eksklusif atas karya ilmiah :
☑ Sekripsi ☐ Tesis ☐ Desertasi ☐ Lain-lain (.....)
yang berjudul :
KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DENGAN ARSITEKTUR INCEPTION-V3

beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Ekslusif ini Perpustakaan UIN Sunan Ampel Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data (database), mendistribusikannya, dan menampilkan/mempublikasikannya di Internet atau media lain secara ***fulltext*** untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan atau penerbit yang bersangkutan.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Perpustakaan UIN Sunan Ampel Surabaya, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Surabaya, 04 Februari 2022

Penulis

(MUJIIBUR-RAHMAN K.)
nama terang dan tanda tangan

ABSTRAK

KLASIFIKASI CITRA PENYAKIT LEUKEMIA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION-V3*

Penderita leukemia memiliki risiko kematian yang sangat tinggi, namun dapat dikurangi dengan dilakukan pengobatan secara dini. Pada umumnya diperlukan seorang dokter ahli, biaya dan waktu yang lama untuk melakukan diagnosis penyakit leukemia secara manual. Proses diagnosis manual yang dilakukan seorang dokter juga berisiko terjadinya kesalahan (*human error*). Hal ini dapat mengakibatkan penderita terlambat dalam pengobatan penyakit leukemia. Pada penelitian ini, dibentuk suatu sistem klasifikasi citra mikroskopis sel tunggal untuk diagnosis penyakit leukemia secara otomatis menggunakan Convolutional Neural Network (CNN) dengan arsitektur Inception-v3. Proses augmentasi diterapkan untuk meningkatkan keragaman data, sehingga mampu mengurangi *overfitting*. Selanjutnya dilakukan uji coba nilai *hyper-parameter* sistem untuk meningkatkan kinerja Inception-v3 dalam klasifikasi citra penyakit leukemia. Uji coba yang dilakukan diantaranya: nilai *learning rate*, *batch size* dan penggunaan metode optimasi RMSProb. Hasil uji coba terbaik klasifikasi penyakit leukemia menggunakan Inception-v3 adalah: akurasi sebesar 100%, sensitifitas sebesar 100%, dan spesitifitas sebesar 100%.

Kata kunci: Citra Mikroskopis, Leukemia, Klasifikasi, CNN, Inception-v3

ABSTRACT

IMAGE CLASSIFICATION OF LEUKEMIA USING CONVOLUTIONAL NEURAL NETWORK WITH INCEPTION-V3 ARCHITECTURE

Leukemia patients have a very high risk of death, but it can be reduced by early treatment. In general, it takes an expert doctor, costs and a long time to diagnose leukemia manually. The manual diagnosis process carried out by a doctor is also at risk of error (human error). This can result in patients being delayed in treating leukemia. In this study, a single cell microscopic image classification system was developed for the diagnosis of leukemia automatically using the Convolutional Neural Network (CNN) with the Inception-v3 architecture. The augmentation process is applied to increase the diversity of the data, so as to reduce *overfitting*. Furthermore, the system's hyper-parameter value was tested to improve the performance of Inception-v3 in image classification of leukemia. The tests carried out include: the value of learning rate, batch size and the use of the RMSProb optimization method. The best trial results for the classification of leukemia using Inception-v3 are: accuracy of 100%, sensitivity of 100%, and specificity of 100%.

Keywords: Microscopic Image, Leukemia, Classification, CNN, InceptionV3

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Leukemia merupakan salah satu jenis kanker darah yang terbentuk pada awal pembentukan sel-sel darah pada sumsum tulang. Tumor ganas leukemia menyerang sel-sel pembentuk darah muda di sumsum tulang, kemudian masuk ke dalam saluran darah dan mempengaruhi metabolisme sel-sel normal di dalam darah tepi (Vogado dkk. , 2018). Kasus yang paling umum terjadi adalah sel ganas leukemia tersebut menyebabkan gangguan dalam pengaturan sel leukosit (sel darah putih) sehingga mengalami *proliferasi* secara tidak teratur, tidak terkendali dan menjadi abnormal (Dia Rofinda , 2012). Menurut World Health Organzation (WHO), terdapat jumlah kasus baru leukemia sebanyak 474.519 jiwa diseluruh dunia pada pertengahan tahun 2020, dengan jumlah kematian sebanyak 311.594 orang. Leukimia juga menjadi jenis kanker paling banyak terdiagnosis pada anak, remaja dan dewasa muda berusia kurang dari 20 tahun dan menyumbang 25,8 persen dari semua kasus kanker pada kelompok usia ini (Leukemia and Lymphoma Society , 2018). Banyaknya jumlah kematian tersebut, menunjukkan bahwasannya penyakit leukemia masih sangat memerlukan perhatian besar, terkait langkah-langkah yang lebih baik diambil dalam upaya pencegahan dan penanganan kasus tersebut.

Penyakit leukemia merupakan salah satu dari banyaknya musibah yang menimpa manusia, dengan risiko tinggi menyebabkan kematiaan. Belum diketahui dengan pasti apa yang menyebabkan timbulnya penyakit tersebut. Namun tidak

seharusnya penyakit ini menyebabkan manusia berputus asa. Sebagai orang yang beriman, sudah sepantasnya meyakini bahwasannya semua musibah itu terjadi atas izin Allah SWT dan sudah sepatutnya senantiasa bersabar dalam menghadapinya. Allah SWT berfirman:

وَلَنَبْلُوَنَّكُمْ بِشَيْءٍ مِّنَ الْخَوْفِ وَالْجُوعِ وَنَقْصٍ مِّنَ الْأَمْوَالِ وَالْأَنفُسِ وَالثَّمَرَاتِ ۗ وَبَشِّرِ الصَّابِرِينَ

Artinya: Dan sungguh akan Kami berikan cobaan kepadamu, dengan sedikit ketakutan, kelaparan, kekurangan harta, jiwa dan buah-buahan. Dan berikanlah berita gembira kepada orang-orang yang sabar (Q.S. Al-Baqarah Ayat 155).

Sabar dalam menjalani setiap ujian yang diberikan Allah SWT, sudah menjadi kewajiban bagi setiap muslim. Namun, masih banyak diantara manusia yang terlarut dalam kesedihan dan keputusasaan dalam menyikapi musibah yang menimpa dirinya. Hal ini disebabkan karena kurangnya berzikir, mengingat Allah SWT. Orang-orang yang senantiasa berzikir mengingat Allah SWT, akan meyakini bahwasanya pasti akan ada petunjuk yang diberikan Allah SWT, dalam menghadapi setiap musibah. Allah SWT berfirman:

مَآ أَصَابَ مِن مُّصِيبَةٍ إِلَّا بِإِذْنِ ٱللَّهِ ۗ وَمَن يُؤْمِنۢ بِٱللَّهِ يَهْدِ قَلْبَهُۥ ۚ وَٱللَّهُ بِكُلِّ شَىْءٍ عَلِيمٌ

Artinya: Tidak ada suatu musibah pun yang menimpa seseorang kecuali dengan izin Allah; dan barangsiapa yang beriman kepada Allah niscaya Dia akan memberi petunjuk kepada hatinya. Dan Allah Maha Mengetahui segala sesuatu (Q.S. At-Taghabun, ayat 11).

Allah SWT senantiasa memberikan petunjuk kepada manusia melalui ciptaan-Nya. Segala yang ada dilangit, dibumi dan terutama pada diri tiap-tiap manusia, terdapat petunjuk bagaimana cara menyikapi setiap musibah yang dihadapi. Oleh

sebab itu, tugas manusia yang dianugrahkan kemampuan berfikir oleh Allah SWT adalah untuk lebih memperhatikan petunjuk tersebut dalam menyikapi segala musibah yang dihadapinya.

Ayat tersebut merupakan sebuah motivasi bagi kaum muslim untuk senantiasa meningkatkan keimanan, dengan cara senantiasa memperluas pengetahuan untuk memahami petunjuk Allah SWT. Ayat ini juga mendorong dilakukannya penelitian ini, yang bertujuan untuk menemukan sebuah solusi terbaik dalam rangka mengatasi banyaknya kematian yang disebabkan oleh penyakit leukemia.

Leukemia merupakan penyakit dengan tingkat kesembuhan yang tinggi melalui deteksi dini. Disisi lain, leukemia juga memiliki tingkat kematian yang tinggi disebabkan terjadi keterlambatan dalam diagnosis. Leukemia sangat sulit terdeteksi karena termasuk dalam jenis tumor cair sehingga tidak terlihat secara fisik, serta gejalanya yang umumnya tidak terlihat hingga dilakukan tes darah yang tepat. Tes darah melalui pengamatan pada citra mikroskopis merupakan satu-satunya cara untuk melihat ada tidaknya sel-sel abnormal. Dengan menganalisis tipe dan banyaknya sel-sel darah abnormal yang terdapat dalam citra mikroskopis, sang ahli kemudian dapat mendiagnosis jenis penyakit leukemia. Berdasarkan jenis sel, penyakit leukemia dikategorikan menjadi leukemia *Myeloblastic* dan *Lymphoblastic*. Sedangkan berdasarkan maturitas, leukemia diklasifikasikan atas leukemia akut dan kronis. Sel kanker leukemia diklasifikasikan akut jika sebagian besar sel-sel immatur (blast), sedangkan leukemia kronis disebabkan sel darah putih dewasa yang tidak mati sesuai siklus yang *apoptosis*. Sel-sel abnormal ini terus berada dalam sirkulasi darah dan sumsum tulang yang dapat menyebabkan pemadatan sumsum sehingga menggganggu jalur produksi sel-sel lain yang tumbuh normal. Sel-sel tersebut juga dapat terakumulasi dalam limfa yang dapat menyebabkan pembengkakan. Lebih lanjut, penyakit leukemia diklasifikasikan dalam empat kelas, yaitu: Acute Myeloblastic

Leukemia (AML), Acute Lymphoblastic Leukemia (ALL), Chronic Myeloblastic Leukemia (CML) dan Chronic Lymphoblastic Leukemia (CLL) (Nugraha , 2017).

Diagnosis penyakit leukemia berdasarkan analisis citra sel-sel mikroskopis yang dilakukan secara manual membutuhkan sosok tenaga ahli pada bidangnya , sehinnga tidak menutup kemungkinan terjadinya kesalahan serta memakan waktu yang panjang. (Pakan , 2018). Untuk mengurangi risiko tersebut serta membantu diagnosis penyakit leukemia secara dini, diperlukan suatu metode untuk diagnosis penyakit leukemia secara otomatis yakni menggunakan Computer Aided Diagnosis (CAD). CAD merupakan sistem komputer dengan tujuan membantu dalam mendeteksi atau mendiagnosis penyakit (Firmino , 2016). *Deep learning* merupakan salah satu metode yang dapat dengan baik digunakan dalam klasifikasi data dalam jumlah besar. *Deep learning* merupakan algoritma jaringan saraf tiruan yang menggunakan random data sebagai input dan memprosesnya melalui banyak lapisan tersembunyi (hidden layer). Setelah itu melakukan transformasi non linier dari data masukan untuk menghitung nilai output (Hasma and Silfianti. , 2018).

Penelitian mengenai klasifikasi leukemia menggunakan metode *deep learning* pernah dilakukan oleh (Rehman dkk. , 2018). Dalam penelitian tersebut dilakukan perbandingan pada beberapa metode lain, yakni: *Nave Bayesian*, KNN, SVM. Dari hasil penelitian ini diketahui *Convolutional Neural Network* (CNN) merupakan metode terbaik dengan akurasi sebesar 97.78%. Convolutional Neural Network (CNN) merupakan salah satu metode yang dapat dengan baik digunakan dalam klasifikasi data berbentuk citra dalam jumlah besar. CNN memiliki banyak arsitektur yang dikembangkan seperti AlexNet, VGG, GoogleNet, ResNet. Diantara srsitektur tersebut. GoogleNet merupakan salah satu metode yang dapat digunakan untuk diagnosis citra dengan akurasi yang tinggi. Hal ini dapat dilihat dari penelitian yang dilakukan oleh (Li dkk. , 2020) mengenai pengenalan hama dilingkungan

menggunakan CNN, menunjukkan bahwasannya GoogleNet merupakan model terbaik yang menghasilkan akurasi sebesar 96.67%. GoogleNet telah menjuari kompetisi *ImageNet Large Scale Visual Recognition Challange* (ILSVRC) 2014 dalam pengenalan dan klasifikasi citra dalam jumlah besar. GoogleNet juga menjadi model pertama yang menghasilkan nilai error sama dengan nilai error yang dihasilkan manusia secara manual, dengan nilai error sebesar 6.7%(Annapurani dan Ravilla , 2019).

GoogleNet atau model Inception merupakan salah satu arsitektur *Convolutional Neural Network* (CNN) yang dikembangkan untuk mengatasi beberapa permasalahan pada proses klasifikasi image menggunakan CNN. Masalah utama yang dijumpai berupa pemilihihan *filter kernel* yang sesuai untuk tipe image dalam proses klasifikasi, serta arsitektur yang terlalu mendalam (*deeper*) sehinnga menyebabkan *overfitting*. GoogleNet menghadirkan solusi dengan menggunakan lebih banyak ukuran *filter kernel* pada tingkat kedalaman yang sama, sehinnga tidak menyebabkan arsitektur semakin mendalam (Szegedy dkk , 2014). Model inception kemudian terus dikembangkan dengan menambah teknik *factorize* dan *dimension reductions* agar proses menjadi lebih efisien dalam komputasi data yang lebih kompleks, serta ditambahkan beberapa metode optimasi lain untuk menghasilkan model yang optimal serta mencegah terjadinya *overfitting*. Pengembangan ini menghasilkan arsitektur Inception-v3 yang telah terbukti lebih baik dari versi-versi sebelumnya, sebagaimana hasil penelitian yang dilakukan oleh (Szegedy dkk , 2015). Dalam penelitian tersebut terlihat arsitektur Inception v3 menghasilkan *error rate* terbaik sebesar 4.2% dibandingkan pendahulunya GoogeNet sebesar 7.89%.

Berdasarkan penelitian-penelitian sebelumnya, dilakukan penelitian mengenai klasifikasi citra penyakit leukemia menggunakan CNN dengan arsitektur Inception-v3. Untuk meningkatkan kinerja model, dilakukan proses augmentasi data. Proses

augmentasi data dilakukan untuk mencegah tejadinya *overfitting*. Penelitian yang dilakukan oleh (Yadav, S. S.) mengenai penerapan CNN berbasis klasifikasi citra medis untuk diagnosis penyakit, menunjukkan bahwa penggunaan augmentasi secara efektif mampu meningkatkan kinerja model. Selain itu, *mini-batch size* juga memiliki pengaruh terhadap kinerja sistem. Penelitian yang dilakukan oleh (F. Schilling , 2016) mengenai pengaruh *batch normalization layers* pada CNN, menunjukkan bahwa *mini-batch size* memiliki pengaruh yang signifikan pada kecepatan *convergence* selama proses pelatihan. Penggunaan ukuran *mini-batch* yang besar menghasilkan fluktuasi akurasi yang lebih sedikit dibanding ukuran *mini-batch* yang kecil. Dalam penelitian tersebut, terlihat bahwa ukuran *mini-batch* yang kecil memberikan hasil akurasi yang lebih baik. Selanjutnya untuk meningkatka kinerja sistem hingga optimal, dilakukan juga ujicoba *learning rate* dan *decay rate* menggunakan optimasi RMSProp. Hasil penelitian ini diharapkan dapat menjadi referensi bagi tenaga kesehatan dalam diagnosis penyakit leukimia secara dini.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang sudah dipaparkan diatas, didapatkan beberapa rumusan masalah sebagai berikut:

1. Bagaimana model optimal klasifikasi citra mikroskopis penyakit leukemia menggunakan *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3?

2. Bagaimana pengaruh penggunaan metode augmentasi image terhadap kinerja sistem *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3 dalam proses klasifikasi citra penyakit leukemia?

3. Bagaimana pengaruh ukuran *mini-batch* terhadap kinerja sistem *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3 dalam proses klasifikasi

citra penyakit leukemia?

1.3. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Dapat mengetahui model optimal klasifikasi penyakit leukemia menggunakan *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3.
2. Dapat mengetahui pengaruh penggunaan metode augmentasi image terhadap kinerja model *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3 dalam proses klasifikasi citra penyakit leukemia.
3. Dapat mengetahui pengaruh ukuran *mini-batch* terhadap kinerja model *Convolutional Neural Network*(CNN) dengan arsitektur Inception-v3 dalam proses klasifikasi citra penyakit leukemia.

1.4. Manfaat Penelitian

Manfaat yang diharapkan dalam penelitian ini:

1. Manfaat Teoritis

   Penelitian ini diharapkan menjadi referensi bagi penelitian lainnya dalam mengembangkan metode klasifikasi citra penyakit khususnya menggunakan Convolutional Neural Network (CNN) dengan arsitektur Inception-v3.

2. Manfaat Praktis

   i. Bagi penulis

   Menambah wawasan baru bagi penulis dalam menerapkan algoritma Convolutional Neural Network (CNN) Inception V3 untuk klasifikasi citra penyakit leukemia.

ii. Bagi ahli medis

Membantu ahli medis dalam mendiagnosis citra penyakit leukemia dengan lebih mudah, cepat dan hasil akurasi yang maksimal.

1.5. Batasan Masalah

Penelitian ini dibatasi dalam lingkup:

i. Metode *Convolutional Neural Network*(CNN) Inception V3 yang digunakan dalam penelitian ini adalah untuk klasifikasi data citra penyakit.

ii. Data yang digunakan pada penelitian ini menggunakan data citra *microscopic* sel tunggal leukemia.

iii. Klasifikasi citra sel darah dikategorikan kedalam tiga kelas, yakni: normal, *acute lymphoblastic leukemia* (ALL), *acute myeloid leukemia* (AML).

iv. Data dibagi menjadi data latih 75% dan data uji 25%.

1.6. Sistematika Penulisan

Sistematika penulisan dalam tugas akhir ini, disusun sebagai berikut:

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah, permasalahan, pembatasan masalah, tujuan dan manfaat penulisan, serta sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi penjelasan tentang seluruh literatur yang digunakan dalam penelitian ini, yakni: leukemia, pengolahan citra digital, CNN dan Inception-v3.

BAB III METODE PENELITIAN

Bab ini menjelaskan metode pengumpulan data, waktu dan tempat penelitian serta Langkah-langkah penelitian.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi hasil dan pembahasan, dilakukan perhitungan setiap langkah-langkah klasifikasi image menggunakan Inception-V3 serta dilakukan ujicoba metode optimasi sistem klasifikasi.

BAB V KESIMPULAN DAN SARAN

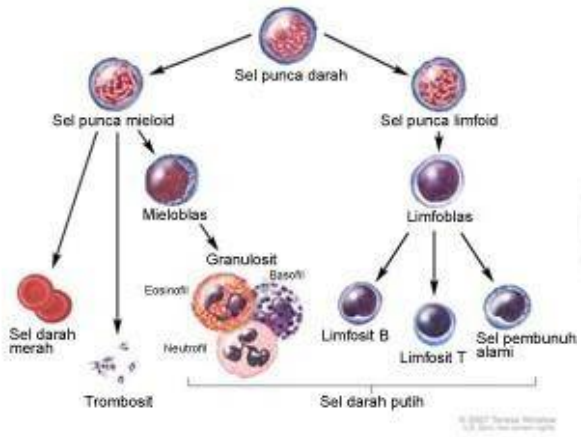
bab ini terdiri atas kesimpulan dan saran terkait dengan metode klasifikasi yang digunakan, serta metode optimasi sistem.

BAB II

TINJAUAN PUSTAKA

2.1. Leukemia

Pada umumnya darah manusia terdiri atas tiga komponen utama, yakni: sel darah merah (*hemogoblin*) yang berfungsi untuk menghantarkan oksigen ke seluruh tubuh, sel darah putih (*leukosit*) yang berfungsi melawan infeksi dan keping darah (*trombosit*) yang berfungsi dalam menghentikan pendarahan. Kelainan pada darah dapat terjadi apabila salah satu komponen tersebut tidak berfungsi dengan semestinya.



Gambar 2.1 Proses pertumbuhan sel darah

Sumber: (Smarter Health , 2020)

Seluruh sel darah terbentuk di dalam sumsum tulang yang mengandung sel darah muda (sel punca darah) (Donna M. Bozzone , 2009). Pada Gambar 2.1 terlihat bagaimana proses pertumbuhan sel darah muda menjadi sel-sel darah dewasa.

Sel darah muda akan berkembang menjadi sel *myeloid* muda dan sel *limfosit* muda, yang selanjutnya berkembang menjadi sel-sel darah dewasa, yakni sel darah merah, sel darah putih dan keping darah.

Sesuai dengan istilahnya, leukemia merupakan salah satu kelainan darah yang terjadi pada sel darah putih manusia. Kelainan ini disebabkan oleh pertumbuhan abnormal pada jaringan hematopoetik yang merupakan jaringan pembentuk sel darah manusia (Rendra, Yaswir, dan Hanif , 2013). Sel-sel darah mudah mieloblas dan limfoblas yang merupakan sel-sel pementuk darah putih. Hal ini menimbulkan beberapa gejala pada penderitanya, seperti pendarahan, kelelahan, demam dan peningkatan resiko infeksi. Sel-sel abnormal ini kemudian dikenal dengan istilah blast atau sel leukemia.

Meski hingga saat ini belum diketahui dengan pasti apa yang menyebabkan timbulnya penyakit leukemia, namun terdapat beberapa faktor yang dapat menjadi penyebab kelainan tersebut. Satu hal yang diketahui adalah sering terjadi perubahan sel yang dimulai dengan mutasi bahan genetis seperti DNA yang merupakan bahan genetis turunan dari orang tua. DNA membentuk sel-sel baru setiap kali membelah dan selama pembelahan, ada kemungkinan terjadi kesalahan dalam replikasi yang dapat menyebabkan pembentukan sel-sel leukemia. Terdapat beberapa faktor lainnya yang diduga menjadi penyebab kelainan tersebut diantaranya: jenis kelamin, usia, sindrom genetik, paparan radiasi, ras dan etnik, riwayat keluarga, paparan zat kimia, obat-obatan, virus, dan riwayat terapi kanker (Rahmawati dan Mutiah , 2014).

Leukemia merupakan penyakit yang memiliki tingkat kesembuhan yang tinggi melalui deteksi dini. Namun sering terjadi keterlambatan dalam deteksi leukemia dikarenakan gejalanya yang umumnya tidak terlihat, hingga dilakukan tes darah yang tepat. Diagnosa leukemia dilakukan dilakukan melalui beberapa tahapan,

yakni: tes fisik, tes darah dan bone marrow biopsi. Pemeriksaan fisik dilakukan apabila terdapat gejala-gejala seperti: Mudah lelah dan badan terasa lemas, terlihat pucat dan mengalami penurunan berat badan yang drastic, Demam, keringat dingin di malam hari, hilangnya nafsu makan, dan atau infeksi berat yang sering terjadi pada pasien, Memar dan mudah berdarah, seperti mimisan atau pendarahan di gusi ketika menyikat gigi, Kelenjar getah bening membengkak dan mungkin menyakitkan, Nyeri tulang atau sendi dan nyeri di perut bagian atas, yang disebabkan oleh pembengkakan hati atau limpa.

Berdasarkan maturitas sel, leukemia diklasifikasikan atas leukemia akut dan kronik. Sel leukemia diklasifikasikan akut jika sebagian besar sel-sel immatur (sel muda) dan memiliki perkembangan sel leukemia yang cepat. Sedangkan jika yang dominan adalah sel matur (sel dewasa) dan perkembangannya cenderung lambat, maka diklasifikasikan sebagai leukemia kronik. Berdasarkan tipe sel, leukemia akut diklasifikasikan dalam dua kelas, yakni: *Acute Myeloblastic Leukemia* (AML) dan *Acute Lymphoblastic Leukemia* (ALL). Leukemia kronik juga diklasifikasikan dalam dua kelas, yakni: *Chronic Myeloblastic Leukemia* (CML) dan Chronic *Lymphoblastic Leukemia* (CLL).

2.2. Citra digital

Menurut (Gonzalez dkk. , 2009), citra digital tersusun atas sejumlah element yang terbatas, dimana tiap element tersebut memiliki nilai dan lokasi tertentu. Element-element tersebut sering juga disebut sebagai *image elements*, *pels* atau *pixels*. Secara matematis, citra digital merupakan fungsi diskrit dan dapat ditulis sebagai fungsi dimensi dua $f(x, y)$. Setiap element mewakili intensitas pada koordinat spatial (x, y) yang terbatas pada ukuran dari citra M dan N. Citra dapat ditampilkan

dalam bentuk matriks dimensi dua, seperti pada Persamaan 2.1.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N) \\ f(1,0) & f(1,1) & \cdots & f(1,N) \\ \vdots & \vdots & \ddots & \vdots \\ f(M,0) & f(M,1) & \cdots & f(M,N) \end{bmatrix} \tag{2.1}$$

Terdapat berbagai jenis citra digital yang sering digunakan dalam pemrosesan suatu citra yakni citra biner, grayscale dan RGB. Citra biner merupakan citra yang tersusun atas *pixels* yang hanya memiliki dua buah nilai intensitas, yakni: 0 (warna hitam) dan 1 (warna putih). Citra grayscale merupakan citra yang tersusun atas *pixels* yang memiliki nilai intensitas atau *grayscale* pada interval 0 sampai 255. Citra RGB merupakan citra yang tersusun atas tiga buah *maps* berbeda dan tiap *maps* mewakili intensitas warna tertentu, yakni: *Red intensity*, *Green intensity*, dan *Blue intensity* (Saifullah , 2020). Ilustrasi citra RGB ditampilkan dalam Gambar 2.2. Citra RGB membentuk matrix 3D dengan ukuran $m \times n \times 3$ yang dapat ditulis dalam Persamaan 2.2.

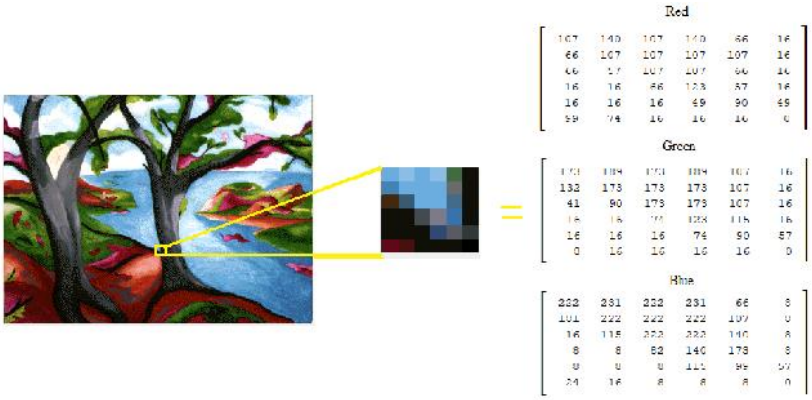
$$\mathrm{I}_{x,y}^{(n)} = z \tag{2.2}$$

dimana:

$\mathrm{I}^{(n)}$ = citra maps ke-n; n $= 1$ (merah), 2 (hijau) dan 3 (biru)

(x, y) = koordinat spatial

z = nilai intensitas warna



Gambar 2.2 Contoh citra RGB

2.3. Image Augmentation

Augmentasi merupakan teknik transformasi image yang dilakukan untuk meningkatkan keragaman data yang digunakan selama proses pelatihan. Selain itu teknik augmentasi juga digunakan untuk membantu menyediakan jumlah data yang sesuai selama proses pelatihan untuk mengurangi resiko *overfitting*, sehingga memperoleh kinerja pelatihan yang lebih baik (Rama, Kumaravel dan Nalini , 2019). Proses augmentasi dilakukan menggunakan beragam teknik transformasi geometri, diantaranya: refleksi, translasi dan rotasi.

1. refleksi

   Refleksi atau pencerminan matrix **I** terhadap garis $x = h$ sehingga menghasilkan bayangan matrix **I'**, dilakukan dengan Persamaan 2.4.

$$\mathbf{I}_{x,y} \xrightarrow{x=h} \mathbf{I'}_{x,y} : \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2h \\ 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.3}$$

   Garis $x = h$ merupakan garis yang membagi image secara simetris, sehingga $2h$ merupakan panjang dari image: $2h = M + 1$. Berdasarkan hal ini Persamaan

2.3 dapat ditulis dalam Persamaan 2.4.

$$\mathbf{I'}(x, y) = \mathbf{I}((\mathrm{M} + 1) - x, y) \tag{2.4}$$

Persamaan 2.4 selanjutnya disebut sebagai Persamaan refleksi image secara horizontal. Sedangkan refleksi secara vertikal dapat dilakukan dengan Persamaan 2.6.

$$\mathbf{I'}(x, y) = \mathbf{I}_{x,(\mathrm{M}_2+1)-y} \tag{2.5}$$

2. Translasi

   Image yang ditranslasikan terhadap nilai perpindahan **T** = $[a, b]$ dapat dilakukan menggunakan Persamaan 2.6.

$$\mathbf{I'}(x, y) = \mathbf{I}_{x+a,y+b} \tag{2.6}$$

3. Rotasi

   Rotasi image terhadap sudut θ dan pusat rotasi (a, b), dapat dilakukan menggunakan Persamaan 2.7.

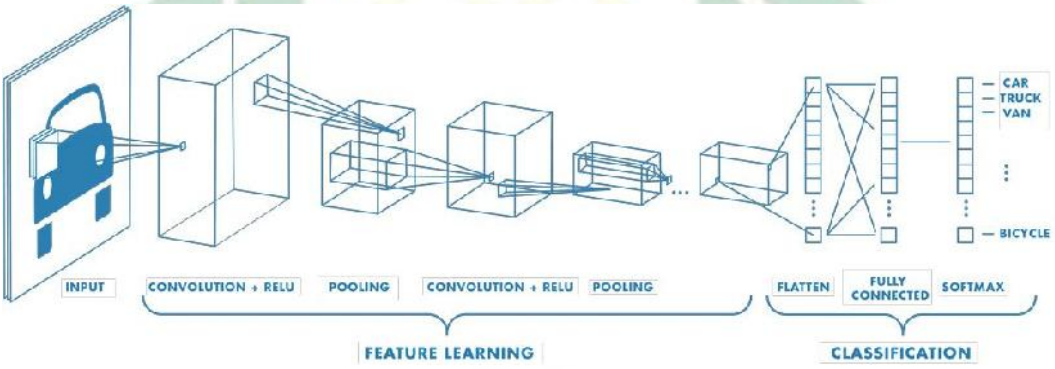
$$\mathbf{I'}(x, y) = \begin{bmatrix} \mathrm{x'} \\ \mathrm{y'} \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} \mathrm{x\text{-}a} \\ \mathrm{y\text{-}b} \end{bmatrix} + \begin{bmatrix} \mathrm{a} \\ \mathrm{b} \end{bmatrix} \tag{2.7}$$

Pusat rotasi (a, b) yang tepat berada ditengah image dengan ukuran $\mathrm{M}_1 \times \mathrm{M}_2$, dapat dihitung dengan Persamaan:

$$a = \frac{\mathrm{M}_1 + 1}{2}$$
$$b = \frac{\mathrm{M}_2 + 1}{2}$$

2.4. Convolution Neural Network

Convolution neural network (CNN) merupakan salah satu metode *Machine learning* (ML), digunakan dalam prediksi, klasifikasi, serta analisis data yang pada umumnya berupa citra digital. CNN menerapkan sebuah operasi matematika convolution yang khusus digunakan dalam operasi linear. Operasi convolution digunakan dalam setidaknya salah satu *layers* (He dkk. , 2019). Terdapat beragam *layers* yang digunakan dalam metode CNN seperti yang ditampilkan dalam Gambar 2.3. Setiap layers memiliki peran tersendiri dalam proses pembelajaran fitur-fitur penting suatu image.

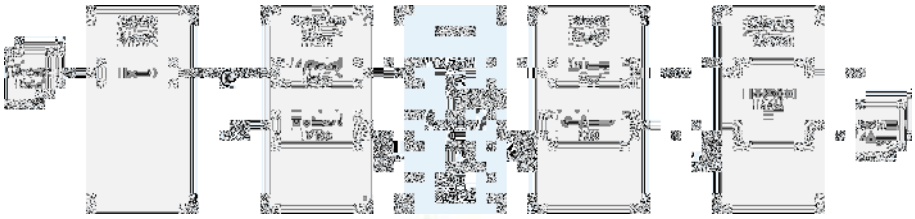


Gambar 2.3 Convolutional Neural Network Layers

Sumber: *Matworks*

Sebagaimana metode *Machine learning* lainnya, proses pembelajaran fitur dalam CNN dapat dibedakan kedalam dua proses, yakni: *forward pass* dan *backward pass*. Selama proses *forward pass* fitur-fitur suatu image akan dipelajari melalui beragam *layers*, hingga akhirnya diklasifikasi kedalam kelas tertentu melalui *classification layers*. Sedangkan selama proses *backward pass*, dilakukan perhitungan *gradient loss* dari tiap *layers* untuk digunakan dalam pembaruan nilai *learnable paremeter* dalam sistem. Kedua proses tersebut terus dilakukan hingga memperoleh

sistem klasifikasi yang optimal (Rafael dan Richard , 2018).



Gambar 2.4 *Forward* dan *Backward Pass* dalam CNN

Sumber: *Matworks*

Seperti yang ditampilkan pada Gambar 2.4, selama proses *forward pass* suatu image dapat dianggap sebagai sebuah *feature maps* (X) yang di-input kedalam sistem. Image tersebut selanjutnya dipelajari dalam layers ke-ℓ, sehingga menghasilkan *feature maps* baru (Z). *Feature maps* yang baru dihasilkan tersebut, selanjutnya digunakan kembali sebagai *feature maps* yang di-input pada layers selanjutnya. Proses ini dapat dituliskan dalam Persamaan 2.8 (Gonzalez dkk. , 2009).

$$X^{(\ell)} = h(Z^{(\ell-1)}); \ell = 1, 2, 3, \cdots, \mathrm{L} \tag{2.8}$$

Untuk mempermudah proses perhitungan, digunakan variabel ($\mathbf{F}$) sebagai variabel dari *feature maps*. Image yang digunakan sebagai suatu input, ditulis sebagai ($\mathbf{F}^{(0)}$) dan *feature maps* yang dihasilkan pada layers ke-ℓ ditulis sebagai ($\mathbf{F}^{(\ell)}$). Persamaan 2.8, selanjutnya dapat dituliskan dalam bentuk Persamaan 2.9.

$$\mathbf{F}^{(\ell)} = h(\mathbf{F}^{(\ell-1)}); \ell = 0, 1, 2, 3, \cdots, \mathrm{L} \tag{2.9}$$

Pada Gambar 2.4 juga diperlihatkan proses *backward pass*. Proses ini dilakukan untuk mendapatkan matrix *gradient loss* dari setiap layers ($\delta^{(\ell)}$). Matrix

gradient loss selanjunya digunakan dalam pembaruan nilai-nilai *learnable parameter*. *learnable parameter* merupakan parameter-parameter yang dapat diperbarui dalam layer tertentu, seperti nilai-nilai bobot dan bias. Perhitungan nilai *gradient loss* dilakukan dengan Persamaan 2.10 (Zhou , 2018).

$$\delta^{(\ell)} = \frac{\partial \mathbf{E}}{\partial \mathbf{F}^{(\ell)}}; \ell = \mathrm{L}, \mathrm{L}-1, \mathrm{L}-2, \cdots, 1 \tag{2.10}$$

Dengan menggunakan aturan *chain rule*, Persamaan 2.10 selanjutnya dapat dituliskan dalam bentuk Persamaan 2.12.

$$\delta^{(\ell)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}^{(\ell+1)}} \frac{\partial \mathbf{F}^{(\ell+1)}}{\partial \mathbf{F}^{(\ell)}} \tag{2.11}$$

$$= \delta^{(\ell+1)} \mathbf{F'}^{(\ell+1)} \tag{2.12}$$

$\delta^{(\ell+1)}$ merupakan *gradient loss* pada layer sebelumnya dan $\mathbf{F'}^{(\ell+1)}$ merupakan turunan pertama fungsi pada layer sebelumnya terhadap *feature maps* pada layer ke-(ℓ).

Terdapat beragam *layers* yang digunakan dalam metode CNN. Setiap layers tersebut menerapkan operasi berbeda untuk mempelajari *feature maps* yang di-input. Seluruh layers tersebut saling terhubung hingga menghasilkan sebuah arsitektur CNN. Layers-layers yang digunakan adalah sebegai berikut:

2.4.1. Input Layer

Input layer merupakan layer yang pertama kali dilalui oleh suatu image yang di-input. Setiap arsitektur CNN memiliki suatu ketetapan nilai dari ukuran image yang dapat diproses dalam layer ini. Contohnya dalam arsitektur Inception-v3 ukur-

an image yang ditetapkan yaitu: $299 \times 299 \times 3$.

Suatu image dengan ukuran $M_1 \times M_2 \times 3$ akan dinormalisasi dalam *input layers* menggunakan Persamaan 2.13 (Singh , 2020).

$$\mathbf{F}_{x,y}^{(n)(1)} = (-1) + \frac{\mathbf{F}_{x,y}^{(n)(0)} - 0}{255 - 0} \times 2$$

$$\mathbf{F}_{x,y}^{(n)(1)} = (-1) + \frac{\mathbf{F}_{x,y}^{(n)(0)}}{255} \times 2 \tag{2.13}$$

keterangan:

$\mathbf{F}^{(0)}$ = Input image

$\mathbf{F}^{(1)}$ = Hasil keluaran pada layer ke-1

(x, y) = Koordinat spatial; $x = 1, 2, 3, \cdots M_1$ dan $y = 1, 2, 3, \cdots M_2$

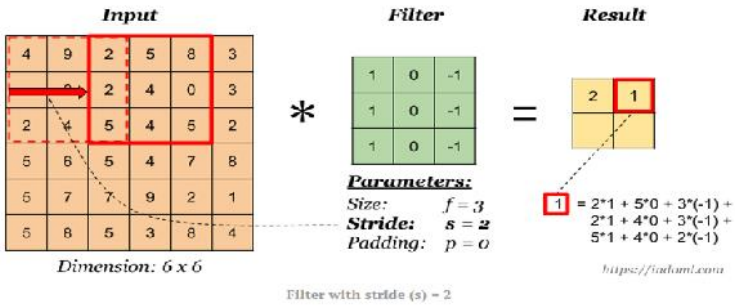
$n = 1, 2, 3$

2.4.2. Convolution Layers

Dalam metode CNN, *convolution layers* merupakan layer utama proses pembelajaran fitur. Proses convolusi antara suatu input image **I** dengan suatu filter atau kernel **W** dapat dituliskan dalam Persamaan 2.14.

$$\mathbf{C} = \mathbf{W} * \mathbf{I} \tag{2.14}$$

Pada Gambar 2.5 diperlihatkan bagaimana proses 2D convolusi antara sebuah feature maps dengan ukuran dimensi 6×6 dan sebuah filter ukuran 3×3.



Gambar 2.5 Ilustrasi proses 2D convolusi

Seperti terlihat pada Gambar 2.5, terdapat beberapa parameter yang perlu diinisialisasi terlebih dahulu sebelum proses perhitungan (Zufar dan Setiyono, 2016).

1. Kernel

   Kernel merupakan matrix filter yang digunakan untuk melakukan transformasi pada suatu input image. Parameter yang perlu diinisialisasi terkait kernel adalah ukuran kernel (K) dan matrix bobot (**W**).

2. Stride

   Stride (s) merupakan vektor dari langkah perpindahan matrix kernel terhadap input image. Perpindahan ini meliputi perpindahan kearah sumbu x dan perpindahan ke arah sumbu y.

3. Padding

   Padding (P) merupakan proses penambahan lapisan pada *border* image. Parameter yang perlu diinisialisasi terkait padding adalah jenis padding dan vektor padding. Jenis padding yang umumnya digunakan pada CNN adalah zero padding, yakni nilai pixels setiap lapisan yang ditambahkan adalah nol. Vektor padding berisi ukuran penambahan baris dan kolom image kearah atas, bawah, kiri dan kanan, sehingga dapat ditulis P = $[p_{top}, p_{botom}, p_{left}, p_{right}]$.

4. *Feature maps*

   feature maps merupakan matrix yang berisikan nilai-nilai fitur yang dihasilkan *convolution layers*. Inisialisasi parameter yang dibutuhkan adalah banyaknya *feature maps* yang dihasilkan (N). *Feature maps* yang dihasilkan berukuran $M_1 \times M_2 \times N$, dengan nilai M dapat dihitung menggunakan Persamaan 2.19.

Forward Pass Convolutional Layers

Proses convolusi merupakan perhitungan jumlah total perkalian *dot product* antara kernel dengan *feature maps* atau image yang di-input, sehingga dapat dituliskan dalam bentuk Persamaan 2.15 (Gonzalez dkk., 2009).

$$\mathbf{C}_{x,y} = \sum_{a=1}^{K_1} \sum_{b=1}^{K_2} \mathbf{W}_{a,b}\, \mathbf{I}_{x-a+1,y-b+1} + b \tag{2.15}$$

Selama proses convolusi, *coordinat spatial* dari input image I juga dipengaruhi oleh nilai stride dan dapat ditulis dalam Persamaan 2.16.

$$\begin{bmatrix} i \\ j \end{bmatrix} = \begin{bmatrix} sx - 1 \\ sy - 1 \end{bmatrix} \tag{2.16}$$

Keterangan:

$\mathbf{C}$ = hasil keluaran proses convolusi

$\mathbf{I}$ = Input image

(x, y) = Koordinat spatial; $x = 1, 2, 3, \cdots \mathrm{M}_1$ dan $y = 1, 2, 3, \cdots \mathrm{M}_2$

$\mathbf{W}$ = Kernel

b = bias

s = stride

Nilai (i, j) selanjutnya disubtitusikan dalam Persamaan 2.15, sehingga menghasilkan Persamaan 2.17.

$$\mathbf{C}_{x,y} = \sum_{a=1}^{\mathrm{K}_1} \sum_{b=1}^{\mathrm{K}_2} \mathbf{W}_{a,b}\, \mathbf{I}_{i-a+1,j-b+1} + b \tag{2.17}$$

Untuk proses convolusi terhadap suatu *input image* yang memiliki banyak *feature maps* dan menghasilkan *feature maps output* sebanyak N, dapat dihitung menggunakan Persamaan 2.18.

$$\mathbf{C}_{x,y}^{(n)} = \sum_{m=1}^{\mathrm{N}_{input}} \sum_{a=1}^{\mathrm{K}_1} \sum_{b=1}^{\mathrm{K}_2} \mathbf{W}_{a,b}^{(m)(n)}\, \mathbf{I}_{i-a+1,j-b+1}^{(m)} + b^{(n)} \tag{2.18}$$

Proses ini menghasilkan *feature maps* baru dengan ukuran dimensi $\mathrm{M}_1 \times \mathrm{M}_2 \times \mathrm{N}_{output}$. Nilai M dapat dihitung menggunakan Persamaan 2.19.

$$\mathrm{M}_{output} = \frac{\mathrm{M}_{input} - \mathrm{K} + 2\mathrm{P}}{\mathrm{s}} + 1 \tag{2.19}$$

Keterangan:

$$\mathrm{K} = \text{ukuran kernel}$$

$$\mathrm{M} = \text{ukuran dimensi maps}$$

$$\mathrm{N} = \text{banyaknya maps}$$

$$\mathrm{P} = \text{padding}$$

$$s = \text{stride}$$

Seperti yang telah dijelaskan dalam Persamaan 2.9, maka Persamaan 2.18 dapat dituliskan dalam bentuk Persamaan 2.20. Persamaan ini selanjutnya disebut sebagai Persamaan *forward pass* pada *convolutional layers*.

$$\mathbf{F}_{x,y}^{(n)(\ell)} = \sum_{m=1}^{\mathrm{N}^{\ell-1}} \sum_{a=1}^{\mathrm{K}_1} \sum_{b=1}^{\mathrm{K}_2} \mathbf{W}_{a,b}^{(m)(n)} \, \mathbf{F}_{i-a+1,j-b+1}^{(m)(\ell-1)} + b^{(n)} \tag{2.20}$$

Keterangan:

$\mathbf{F}^{\ell}$ = *feature maps* hasil keluaran pada layer ke-ℓ

$\mathbf{I}$ = Input image

$*$ = operasi convolusi

(x, y) = Koordinat spatial; $x = 1, 2, 3, \cdots$ M_1 dan $y = 1, 2, 3, \cdots \mathrm{M}_2$

$\mathbf{W}$ = Kernel

b = bias

s = stride

K = ukuran kernel

M = ukuran dimensi maps

$\mathrm{N}^{\ell-1}$ = banyaknya maps pada layer sebelumnya

$n = 1, 2, 3, \cdots \mathrm{N}^{\ell}$

Backward Pass Convolutional Layers

Dalam *convolutional layers*, terdapat *learnable parameter* berupa nilai-nilai bobot kernel **W** dan bias b. Untuk melakukan pembaruan *learnable parameter* dalam layer ini, terlebih dahulu dilakukan perhitungan *gradient loss* ($\delta^{(\ell)}$) terhadap hasil *convolutional layers*. Matrix $\delta^{(\ell)}$ dapat dihitung dengan menggunakan Persamaan 2.12.

Selanjutnya pembaruan nilai-nilai bobot dilakukan dengan menggunakan

Persamaan 2.23 (Zhang , 2016).

$$\mathbf{W}_{x,y}^{(m)(n)(\ell)} = \mathbf{W}_{x,y}^{(m)(n)(\ell)} - \alpha \frac{\partial \mathrm{E}}{\partial \mathbf{W}_{x,y}^{(m)(n)(\ell)}} \tag{2.21}$$

$$= \mathbf{W}_{x,y}^{(m)(n)(\ell)} - \alpha \sum_a \sum_b \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{a,b}^{(n)(\ell)}} \frac{\partial \mathbf{F}_{a,b}^{(n)(\ell)}}{\partial \mathbf{W}_{x,y}^{(m)(n)(\ell)}} \tag{2.22}$$

$$= \mathbf{W}_{a,b}^{(m)(n)(\ell)} - \alpha \sum_a \sum_b \delta_{a,b}^{(n)(\ell)} \mathrm{F}_{x-a+1,y-b+1}^{(m)(\ell-1)} \tag{2.23}$$

Pembaruan bias dapat dilakukan dengan menggunakan Persamaan 2.26:

$$b^{(n)(\ell)} = b^{(n)(\ell)} - \alpha \frac{\partial \mathrm{E}}{\partial b^{(n)(\ell)}} \tag{2.24}$$

$$= b^{(n)(\ell)} - \alpha \sum_a \sum_b \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{a,b}^{(n)(\ell)}} \frac{\partial \mathbf{F}_{a,b}^{(n)(\ell)}}{\partial b^{(n)(\ell)}} \tag{2.25}$$

$$= b^{(n)(\ell)} - \alpha \sum_a \sum_b \delta_{a,b}^{(n)(\ell)} \tag{2.26}$$

Keterangan:

$\mathbf{F}^{(\ell)}$ = Feature maps pada layer ke-ℓ

$\mathbf{W}^{(\ell)}$ = bobot kernel pada layer ke-ℓ

$b^{(\ell)}$ = bias pada layer ke-ℓ

$\delta^{(\ell)}$ = *gradient loss* pada *convolution layer* ke-ℓ

$\partial \mathbf{E}$ = Perubahan loss

$\partial \mathbf{F}^{(\ell)}$ = Perubahan *feature maps output* pada layer ke-ℓ

(a, b) = Koordinat spatial; $a = 1, 2, 3, \cdots$ K dan $b = 1, 2, 3, \cdots$ K

Selanjutnya untuk menghitung *gradient loss* pada layer selanjutnya, dibutuhkan matrix $\mathbf{F'}^{(n)(\ell)}$. Matrix tersebut merupakan *feature maps* hasil turunan per-

tama fungsi pada layer convolusi ini terhadap *feature maps* pada layer sebelumnya. Proses perhitungan ini dapat dilakukan dengan Persamaan 2.29.

$$\mathbf{F'}_{x,y}^{(n)(\ell)} = \frac{\partial \mathbf{F}_{a,b}^{(n)(\ell)}}{\partial \mathbf{F}_{x,y}^{(n)(\ell-1)}} \tag{2.27}$$

$$= \frac{\partial}{\partial \mathbf{F}_{x,y}^{(n)(\ell-1)}} \left[\sum_m \sum_a \sum_b \mathbf{W}_{a,b}^{(m)(n)} \, \mathbf{F}_{i-a+1,j-b+1}^{(m)(\ell-1)} + b^{(n)} \right] \tag{2.28}$$

$$= \mathbf{W}_{x-a+1,y-b+1}^{(m)(n)} \tag{2.29}$$

gradient loss pada layer berikutnya, dapat dihitung dengan menggunakan Persamaan 2.33.

$$\delta_{x,y}^{(n)(\ell-1)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{x,y}^{(n)(\ell-1)}} \tag{2.30}$$

$$= \sum_m \sum_a \sum_b \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{a,b}^{(m)(\ell)}} \frac{\partial \mathbf{F}_{a,b}^{(m)(\ell)}}{\partial \mathbf{F}_{x,y}^{(n)(\ell-1)}} \tag{2.31}$$

$$= \sum_m \sum_a \sum_b \delta_{a,b}^{(m)(\ell)} \mathbf{F'}_{x,y}^{(n)(\ell+1)} \tag{2.32}$$

$$= \sum_m \sum_a \sum_b \delta_{x,y}^{(m)(\ell)} \mathbf{W}_{x-a+1,y-b+1}^{(m)(n)} \tag{2.33}$$

Keterangan:

$\mathbf{F}^{(\ell)}$ = Feature maps pada layer ke-ℓ

$\mathbf{F'}^{(\ell)}$ = turunan pertama hasil convolusi layer ke-ℓ

$\mathbf{W}^{(\ell)}$ = bobot kernel pada layer ke-ℓ

$b^{(\ell)}$ = bias pada layer ke-ℓ

$\delta^{(\ell)}$ = *gradient loss* pada *convolution layer* ke-ℓ

$\partial\mathbf{E}$ = Perubahan loss

$\partial\mathbf{F}^{(\ell)}$ = Perubahan *feature maps output* pada layer ke-ℓ

(a, b) = Koordinat spatial; $a = 1, 2, 3, \cdots$ K dan $b = 1, 2, 3, \cdots$ K

2.4.3. Batch Normalization layers

Feature maps yang diperoleh dari *convolution layer*, selanjutnya dinormalisasi melalui *Batch Normalization layers*. Proses ini dilakukan untuk mengurangi interval perubahan *covariant* atau nilai dari *feature maps* yang diinput saat proses training. Dalam layer ini dilakukan perhitungan nilai *mean* dan *varian* untuk menorrmalisasi output layer sebelumnya. Kedua nilai statistik tersebut lebih baik digunakan dalam penenetapan ukuran *mini-batch* yang besar. Selain itu pengunaan beberapa parameter tambahan yakni *scale* dan *shift*, dapat mempercepat proses training dalam menghasilkan model yang optimal (Ioffe dan Szegedy, 2015). Beberapa parameter yang digunakan dalam layer ini, yaitu:

1. nilai mean (μ)

$$\mu = \frac{1}{\mathrm{M}_1 \times \mathrm{M}_2} \sum_{i=1}^{\mathrm{M}_1} \sum_{j=1}^{\mathrm{M}_2} \mathbf{F}_{i,j}^{(n)(\ell-1)} \tag{2.34}$$

2. nilai standar deviasi (σ^2)

$$\sigma^2 = \frac{1}{\mathrm{M}_1 \times \mathrm{M}_2} \sum_{i=1}^{\mathrm{M}_1} \sum_{j=1}^{\mathrm{M}_2} (\mathbf{F}_{i,j}^{(n)(\ell-1)} - \mu)^2 \tag{2.35}$$

3. *scale* (γ) dan *shift* (β)

   Scale dan *shift* merupakan parameter yang di inisialisasi sebelum dilakukan proses perhitungan pada layer ini. Kedua parameter ini ditambahkan sebagai parameter identititas transformasi. Parameter ini dapat ditulis dalam Persamaan:

$$y = \gamma \widehat{x} + \beta \tag{2.36}$$

Forward Pass Batch Normalization Layers

Proses normalisasi dalam *Batch Normalization layers* dilakukan dengan Persamaan 2.37.

$$\mathbf{F}_{x,y}^{(n)(\ell)} = \gamma^{(n)} \widehat{x}_{x,y} + \beta^{(n)} \tag{2.37}$$

dimana:

$$\widehat{x}_{x,y} = \frac{\mathbf{F}_{x,y}^{(n)(\ell-1)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

Backward Pass Batch Normalization Layers

Terdapat dua *learnable paramater* dalam layer ini, yaitu: *scale* (γ) dan *shift* (β). Untuk melakukan pembaruan *learnable parameter* dalam layer ini, terlebih dahulu dilakukan perhitungan *gradient loss* dari hasil normalisasi layer ini. Hal ini dilakukan dengan Persamaan2.40.

$$\frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{a,b}^{(n)(\ell)}} \frac{\partial \mathbf{F}_{a,b}^{(n)(\ell)}}{\partial \widehat{x}_{x,y}} \tag{2.38}$$

$$= \delta_{x,y}^{(n)(\ell)} \gamma \tag{2.39}$$

$$= \delta_{x,y}^{(n)(\ell)} \tag{2.40}$$

Selama proses training kedua parameter *scale* (γ) dan *shift* (β) diperbarui menggunakan Persamaan 2.43 dan 2.46.

$$\gamma^{(n)(\ell)} = \gamma^{(n)(k)} - \alpha \frac{\partial \mathrm{E}}{\partial \gamma^{(n)(\ell)}} \tag{2.41}$$

$$= \gamma^{(n)(k)} - \alpha \sum_x \sum_y \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}} \frac{\partial \widehat{x}_{x,y}}{\partial \gamma^{(n)(\ell)}} \tag{2.42}$$

$$= \gamma^{(n)(k)} - \alpha \sum_x \sum_y \delta_{x,y}^{(n)(\ell)} \widehat{x}_{x,y} \tag{2.43}$$

$$\beta^{(n)(k)} = \beta^{(n)(k)} - \alpha \frac{\partial \mathrm{E}}{\partial \beta^{(n)(k)}} \tag{2.44}$$

$$= \beta^{(n)(k)} - \alpha \sum_x \sum_y \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}} \frac{\partial \widehat{x}_{x,y}}{\partial \beta^{(n)(k)}} \tag{2.45}$$

$$= \beta^{(n)(k)} - \alpha \sum_x \sum_y \delta_{x,y}^{(n)(\ell)} \tag{2.46}$$

Untuk menghitung *gradient loss* pada layer selanjutnya, terlebih dahulu dilakukan perhitungan sebagaimana berikut:

$$\frac{\partial \mathrm{E}}{\partial \sigma^2} = \frac{\partial \mathrm{E}}{\partial \widehat{x}} \frac{\partial \widehat{x}}{\partial \sigma^2} \tag{2.47}$$

$$= \frac{\partial \mathrm{E}}{\partial \widehat{x}} \left(\mathbf{F}^{(\ell)} - \mu\right) \frac{-1}{2} \left(\sigma^2 + \epsilon\right)^{\frac{-3}{2}} \tag{2.48}$$

$$\frac{\partial \mathrm{E}}{\partial \mu} = \frac{\partial \mathrm{E}}{\partial \widehat{x}} \frac{\partial \widehat{x}}{\partial \mu} \tag{2.49}$$

$$= \frac{\partial \mathrm{E}}{\partial \widehat{x}} \frac{-1}{\sqrt{\sigma^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma^2}(-2)\left(\mathbf{F}^{(\ell)} - \mu\right) \tag{2.50}$$

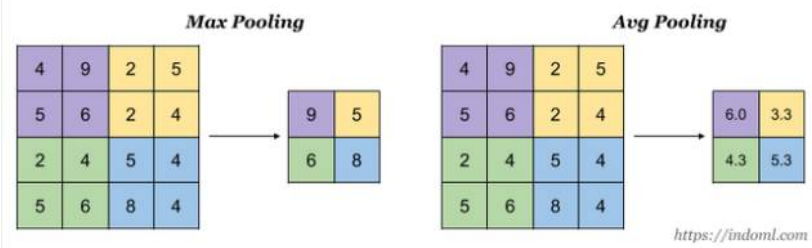
$$\delta_{x,y}^{(\ell-1)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}^{(\ell-1)}} \tag{2.51}$$

$$= \frac{\partial \mathrm{E}}{\partial \widehat{x}} \frac{\partial \widehat{x}}{\partial \mathbf{F}^{(\ell-1)}} + \frac{\partial \mathrm{E}}{\partial \sigma^2} \frac{\partial \sigma^2}{\partial \mathbf{F}^{(\ell-1)}} + \frac{\partial \mathrm{E}}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{F}^{(\ell-1)}} \tag{2.52}$$

$$= \frac{\partial \mathrm{E}}{\partial \widehat{x}} \frac{-1}{\sqrt{\sigma^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma^2}(-2)\frac{\left(\mathbf{F}^{(\ell)} - \mu\right)}{m} + \frac{\partial \mathrm{E}}{\partial \mu}\frac{1}{m} \tag{2.53}$$

2.4.4. Pooling Layers

Pooling layer berfungsi dalam mereduksi data dengan menggabungkan output pada *neuron clusters* suatu layer menjadi neuron tunggal pada layer selanjutnya (Arrofiqoh dan Harintaka, 2018). Terdapat beberapa metode dalam melakukan proses *pooling*, seperti *max pooling* dan *average pooling*. Ilustrasi proses pooling dapat dilihat pada Gambar 2.6.



Gambar 2.6 Ilustrasi *Max pool dan avarage pool*

Forward Pass Pooling layers

Selama proses *forward pass*, *feature maps* yang dihasilkan layer sebelumnya direduksi menggnakan metode max pooling atau average pooling. Pada metode

max pooling nilai yang dipilih adalah nilai terbesar pada tiap cluster, seperti yang ditampilkan pada Gambar 2.6. Proses ini dapat dituliskan dalam Persamaan 2.54 (Boue , 2018).

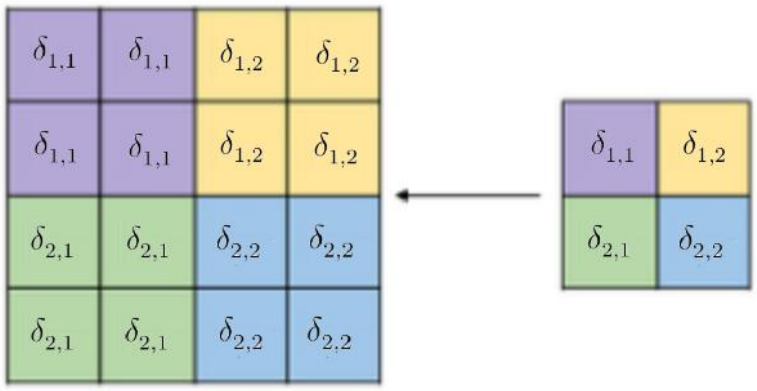
$$\mathbf{F}_{x,y}^{(\ell)} = argmax(\mathbf{F}_{a,b}^{(\ell-1)}) \tag{2.54}$$

Sedangkan pada avg pooling, nilai yang digunakan pada layer selanjutnya adalah nilai hasil rata-rata pada setiap clusters (Arrofiqoh dan Harintaka , 2018). Proses ini dapat dituliskan dalam Persamaan 2.55.

$$\mathbf{F}_{x,y}^{(\ell)} = \frac{1}{M_1 \times M_2} \sum_a \sum_b \mathbf{F}_{a,b}^{(\ell-1)} \tag{2.55}$$

Backward Pass Pooling layers

Selama proses *backward pass*, matrix *gradient loss* ($\delta^{(\ell)}$) akan dirubah ukurannya menggunakan Persamaan 2.56. Proses ini menghasilkan $\delta^{(\ell)}$ baru dengan ukuran yang sama dengan *feature maps* yang di-input dalam *pooling layers* ($\mathbf{F}^{(\ell-1)}$) selama proses *forward pass*. Pada Gambar 2.7 ditampilkan proses transformasi tersebut.



Gambar 2.7 Ilustrasi transformasi matrix *gradient loss* dalam pooling layer

$$\delta_{baru}^{(\ell)} = reshape(\delta^{(\ell)}) \tag{2.56}$$

Selanjutnya *gradient loss* pada layer berikutnya, dapat dihitung dengan Persamaan 2.59.

$$\delta_{x,y}^{(n)(\ell-1)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{x,y}^{(\ell-1)}} \tag{2.57}$$

$$= \frac{\partial \mathrm{E}}{\partial \mathbf{F}_{a,b}^{(\ell)}} \frac{\partial \mathbf{F}_{a,b}^{(\ell)}}{\partial \mathbf{F}_{x,y}^{(\ell-1)}} \tag{2.58}$$

$$= reshape(\delta^{(\ell)}) \mathbf{F'}_{x,y}^{(\ell)} \tag{2.59}$$

Perhitungan $\mathbf{F'}^{(\ell)}$ pada *max pooling* dan *avg pooling* dapat dilakukan dengan menggunakan Persamaan 2.60 dan 2.61.

Untuk *max pooling*:

$$\mathbf{F'}_{x,y}^{(\ell)} = \begin{cases} 1, & \text{jika } \mathbf{F}_{x,y}^{(\ell)} \text{merupakan maximum cluster} \\ 0, & \text{lainnya} \end{cases} \tag{2.60}$$

Untuk *avg pooling*:

$$\mathbf{F'}_{x,y}^{(\ell)} = \frac{1}{M_1 \times M_2} \tag{2.61}$$

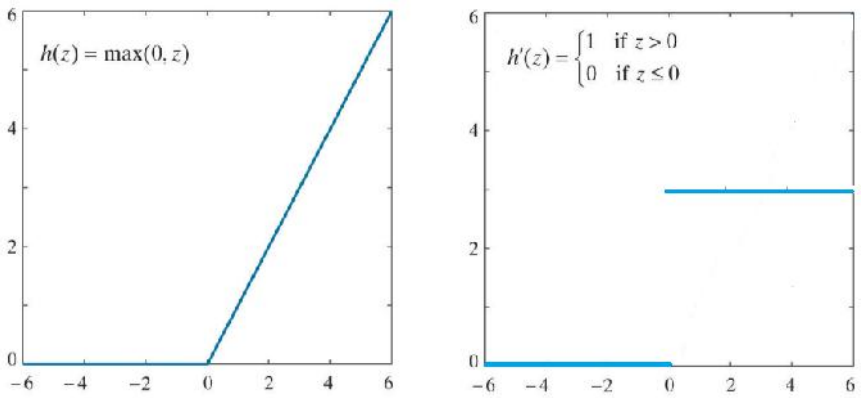
2.4.5. Rectifier linear unit (ReLU)

Rectifier linear unit (ReLU) merupakan fungsi aktifasi yang paling umum digunakan dalam CNN. ReLu merubah nilai-nilai pada *future maps* mengikuti Persamaan 2.62. Distribusi data yang dihasilkan ditampilkan dalam grafik 2.8.

$$\mathbf{F}_{x,y}^{(\ell)} = \max\left(0, \mathbf{F}_{x,y}^{(\ell-1)}\right) \tag{2.62}$$

Fungsi aktifasi ReLu juga sering digunakan karena memiliki turunan fungsi yang sederhana, seperti pada Persamaan 2.63. Hal ini membantu dalam mengurangi biaya komputasi sistem.

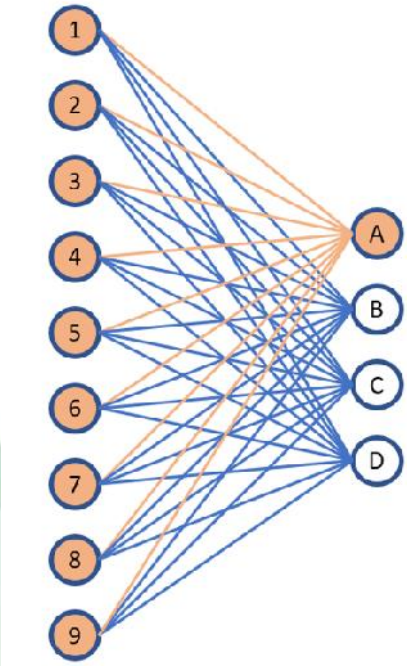
$$\mathbf{F'}_{x,y}^{(\ell)} = \begin{cases} 1, & \text{jika } \mathbf{F}_{x,y}^{(\ell)} \geq 0 \\ 0, & \text{lainnya} \end{cases} \tag{2.63}$$



Gambar 2.8 Grafik fungsi ReLu

2.4.6. Fully Connected Layers

Prinsip klasifikasi menggunakan *fully connected* (FC) sama seperti neural network pada umumnya. Sebelumnya, *Future maps* yang dihasilkan pada *layers* sebelumnya, ditransformasikan dalam bentuk *pattern vector* dalam *flatten layer*, sehingga dapat diklasifikasi dalam layer FC.



Gambar 2.9 Ilustrasi *single layer fully connected layers*

Pada Gambar 2.9 diperlihatkan Gambar *single layer fully conected layers*, dimana setiap nodes pada layer FC terhubung pada setiap nodes layer selanjutnya. Terdapat 1×9 feature input dan 1×4 output. Sehingga untuk melakukan perhitungan dibutuhkan 9×4 matrix bobot dan 1×4 matrix bias.

Forward Pass Fully Conected Layers

Proses *forward pass* pada layer FC dapat dilakukan dengan menggunakan persamaaan 2.64 (Zhang , 2016).

$$\mathbf{F}_j^{(\ell)} = \sum_{i=1}^{M} \mathbf{W}_{i,j}^{(\ell)} \mathrm{F}_i^{(\ell-1)} + b_j^{(\ell)} \tag{2.64}$$

Fitur-fitur yang dihasilkan Persamaan 2.64, selanjutnya diklasifikasi pada

kelas-n berdasarkan nilai tertinggi fungsi probabilitas *softmax*. Nilai probabilitas ini dihitung menggunakan Persamaan 2.65.

$$p_n = \frac{e^{F_n}}{\sum_{i=1}^{M} e^{F_i}} \tag{2.65}$$

Langkah terakhir adalah dilakukan perhitungan nilai error (loss) dari sistem. Perhitungan nilai loss dilakukan menggunakan metode *cross entropy*. Perhitungan ini dilakukan menggunakan Persamaan 2.66.

$$\mathrm{E} = -\sum_{n=1}^{\mathrm{N}} t_n ln(p_n) \tag{2.66}$$

Backward Pass Fully Conected Layers

Terdapat *learnable parameters* dalam FCL berupa nilai bobot dan bias. Untuk memperbarui bobot dan bias pada FCL, terlebih dahulu dicari nilai *gradient error* terhadap hasil output pada *fully connected layers*. Hal ini dilakukan dengan Persamaan 2.67 (Zhang , 2016).

$$\delta_j^{(\ell)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}_j^{\ell}} \tag{2.67}$$

$$= \sum_i \frac{\partial \mathrm{E}}{\partial p_i} \frac{\partial p_i}{\partial \mathbf{F}_j^{\ell}} \tag{2.68}$$

dimana:

$$\frac{\partial \mathrm{E}}{\partial p_i} = \frac{\partial(-\sum_{n=1}^{\mathrm{N}} t_n ln(p_n))}{\partial p_i} \tag{2.69}$$

$$= -\frac{t_i}{p_i} \tag{2.70}$$

$$\frac{\partial p_i}{\partial \mathbf{F}_j^{\ell}} = \begin{cases} p_i(1-p_i), & \text{jika } i = j \\ -p_i p_j, & \text{lainnya} \end{cases} \tag{2.71}$$

Keterangan:

$\mathbf{F}^{(\ell)}$ = Feature maps pada layer ke-ℓ

p = nilai aktivasi pada softmax layers

$\mathbf{E}$ = nilai loss

Persamaan 2.70 dan 2.71 kemudian disubtitusikan dalam Persamaan 2.67.

$$\delta_j^{(\ell)} = \sum_{i,i\neq j} -\frac{t_i}{p_i}(-p_i p_j) + (-\frac{t_i}{p_i})p_i(1-p_i) \tag{2.72}$$

$$= \sum_{i,i\neq j} t_i p_i - t_i + t_i p_i \tag{2.73}$$

$$= \sum_{i} t_i p_i - t_i \tag{2.74}$$

$$= p_i - t_i \tag{2.75}$$

Keterangan:

$\delta^{(\ell)}$ = nilai gradient loss pada layer ke-ℓ

p = nilai aktivasi pada softmax layers

t = nilai target aktual

Untuk memperbarui bobot pada tiap *future maps* digunakan Persamaan 2.78,

$$\mathbf{W}_{i,j} = \mathbf{W}_{i,j} - \alpha \frac{\partial \mathrm{E}}{\partial \mathbf{W}_{i,j}} \tag{2.76}$$

$$= \mathbf{W}_{i,j} - \alpha \frac{\partial \mathrm{E}}{\partial \mathbf{F}_j^{\ell}} \frac{\partial \mathbf{F}_j^{\ell}}{\partial \mathbf{W}_{i,j}} \tag{2.77}$$

$$= \mathbf{W}_{i,j} - \alpha \delta_j^{(\ell)} \mathbf{F}_i^{\ell-1} \tag{2.78}$$

serta Persamaan 2.81 untuk memperbarui bias.

$$b_j = b_j - \alpha \frac{\partial \mathrm{E}}{\partial b_j} \tag{2.79}$$

$$= b_j - \alpha \frac{\partial \mathrm{E}}{\partial \mathbf{F}_j^{\ell}} \frac{\partial \mathbf{F}_j^{\ell}}{\partial b_j} \tag{2.80}$$

$$= b_j - \alpha \delta_j^{(\ell)} \tag{2.81}$$

gradient loss pada layer berikutnya, dapat dihitung dengan menggunakan

Persamaan 2.85.

$$\delta_j^{(\ell-1)} = \frac{\partial \mathrm{E}}{\partial \mathbf{F}_j^{(\ell-1)}} \tag{2.82}$$

$$= \sum_i \frac{\partial \mathrm{E}}{\partial \mathbf{F}_i^{(\ell)}} \frac{\partial \mathbf{F}_i^{(\ell)}}{\partial \mathbf{F}_j^{(\ell-1)}} \tag{2.83}$$

$$= \sum_i \delta_i^{(\ell)} \mathbf{F'}_j^{(\ell+1)} \tag{2.84}$$

$$= \sum_i \delta_i^{(\ell)} \mathbf{W}_{i,j} \tag{2.85}$$

Keterangan:

$\mathbf{F}^{(\ell)}$ = Feature maps pada layer ke-ℓ

$\mathbf{W}$ = bobot

b = bias

$\mathbf{E}$ = nilai loss

$\delta^{(\ell)}$ = nilai gradient loss pada layer ke-ℓ

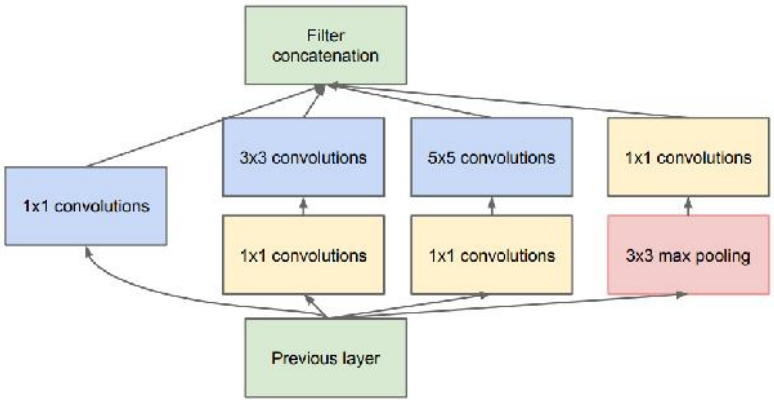
α = *learning rate*

2.5. Inception V3

Terdapat beragam arsitektur yang terus dikembangkan untuk meningkatkan performa dari CNN, dimulai dari: LeNet, AlexNet, ZFNet, GoogLeNet, VGG, hingga yang terbaru ResNet. Perbedaan dari tiap arsitektur dapat dilihat dari ukuran input image, banyaknya layers, pemilihan ukuran kernel dan lainnya (Swapna, Sharma dan Prasadh , 2020).

Inception V3 merupakan salah satu arsituktur convolutional neural network

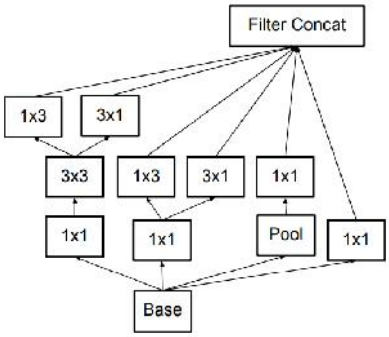
(CNN) yang dikembangkan dari versi sebelumnya GoogleNet (Inception) dan inception V2. Googlenet pertama kali diperkenalkan pada tahun 2014 dan dinobatkan sebagai arsitektur terbaik pada kompetisi *ImageNet Large Scale Visual Recognition Challange* (ILSVRC) 2014. Inception memperkenalkan teknik baru dalam arsitektur CNN, yakni dilakukan pemfaktoran sebuah layer convolution menjadi *multi-layers* dengan ukuran kernel yang lebih kecil. Pengunaan teknik ini, mampu mengurangi jumlah parameter dengan cara berbagi bobot dalam *multi-layers* tersebut. Selain itu penggunaan teknik ini juga mampu menghasilkan arsietektur yang leih dalam tanpa menambah *computational cost*.



Gambar 2.10 Model inception

Seperti yang ditampilkan pada Gambar 2.10, Sebuah layar convolution dengan ukuran 7×7 digantikan dengan *multy-layers* convolution yang berisi tiga jenis convolution layers dengan ukuran kernel berbeda. Kernel yang digunakan berukuran 1×1, 3×3 dan 5×5. Proses konvolusi dalam ketiga layer tersebut menghasilkan *feature maps* dangan ukuran yang sama dan selanjutnya akan digabungkan dalam *concatenated layer*.

Pada perkembangan selanjutnya (Inception-v2), kernel berukuran 3x3 dan 5x5 difaktorkan ke bentuk *asymetric kernel* seperti pada Gambar 2.11. Kernel berukuran 3x3 dan 5x5 difaktorkan ke bentuk $1 \times n$ dan $n \times 1$.



Gambar 2.11 Model inception V2

Sebanyak 315 layers digunakan untuk membentuk arsitektur inceptionV3 seperti yang ditampilkan dalam Gambar 2.15. Diantaranya, terdapat 94 *convolution layers* yang digunakan, dengan rincian seperti pada tabel 2.1. Inception-v3 dikembangkan untuk meningkatkan performa inceptionV2. InceptionV3 memiliki arsitektur yang lebih dalam dibandingkan dengan inceptionV2. *Convolutional layers* dengan ukuran kernel 7×7 ditambahkan dalam arditektur InceptionV3 dan difaktorkan kedalam *multy-layers* convolution dengan ukuran 1×7 dan 7×1.

Tabel 2.1 Ukuran kernel dalam convolution layers

Ukuran Kernel	Jumlah layers
1 x 1	40
1 x 3	4
3 x 1	4
3 x 3	17
5 x 5	3
1 x 7	13
7 x 1	13
Total	94

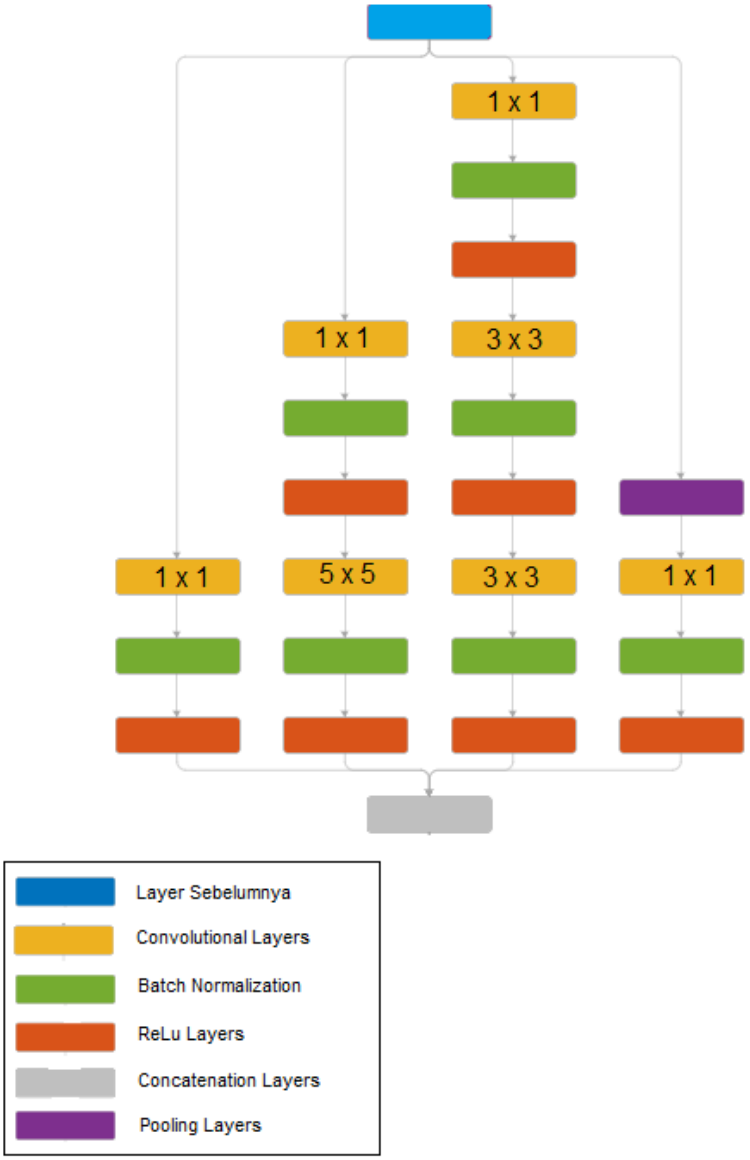
Terdapat tiga model arsitektur CNN yang dikembangkan dalam Inception-v3. Ketiga model ini seperti yang ditampilkan dalam Gambar 2.12, Gambar 2.13 dan Gambar 2.14. Selain penambahan layers baru kedalam arsitektur, inceptionV3 juga dikembangkan dengan menambah beberapa metode optimasi, seperti penggunaan RMSProb optimizer.

RMSProp (*Root Mean Square Propagation*) merupakan metode optimasi yang bekerja denga cara menyimpan nilai moving avarage hasil perhitungan *loss* selama proses pelatihan. Selanjutnya RMSProp Optimization membagi *leraning rate* dengan nilai tersebut dan sebuah nilai *exponentially decaying average* (Ruder , 2016). Perhitungan RMSProp dilakukan dengan menggunakan Persamaan 2.87.

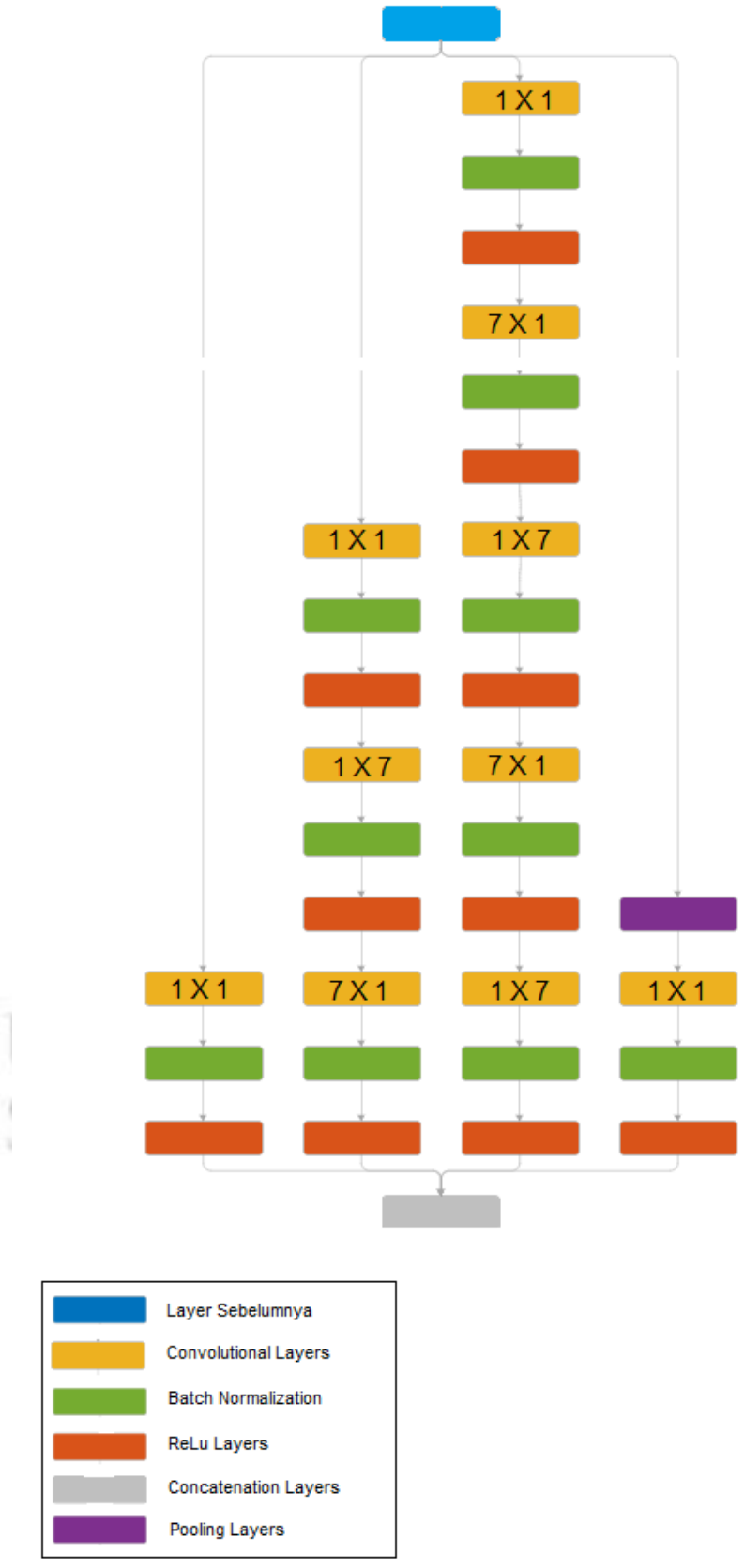
$$E(g^2)_t = \beta E(g^2)_(t-1) + (1-\beta)\delta_t^2 \tag{2.86}$$

$$w_t = w_{(t-1)} - \frac{\alpha}{E(g^2)_t}\delta_t \tag{2.87}$$

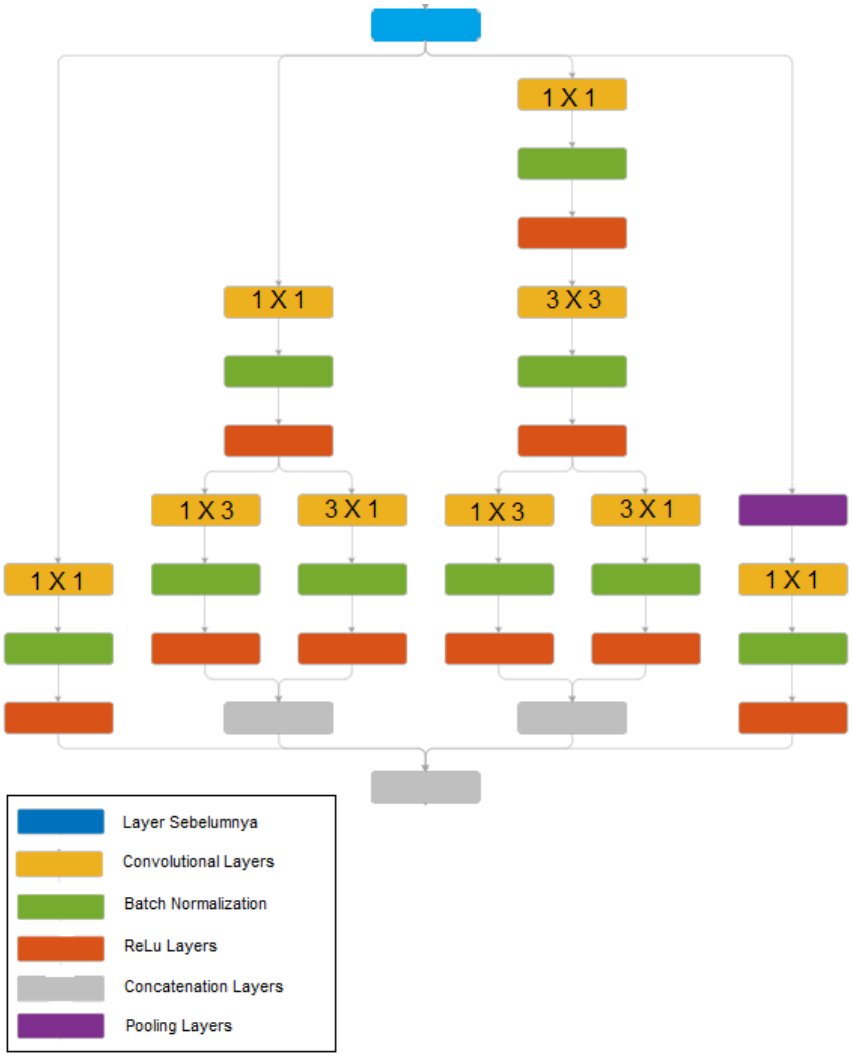
dimana: $E(g^2)_t$ merupakan *moving avarage*, β merupakan nilai paramater *decay rate*, δ_t merupakan perubahan nilai error, w merupakan bobot dan α merupakan learning rate.



Gambar 2.12 model I multi-layers inception V3



Gambar 2.13 model II multi-layers inception V3



Gambar 2.14 model III multi-layers inception V3

Gambar 2.15 Arsitektur inception V3

2.6. Hyperparameters

Hyperparameters merupakan parameter yang memebentuk struktur suatu model. Tidak seperti *learnable parameters*, *hyperparameters* tidak mengalami pembaruan selama proses training. *Hyperparameters* ditetapkan sebelum proses training dan kinerja model dapat berubah secara drastis tergantung nilai dari *hyperpa-*

rameters yang ditetapkan. Untuk mengoptimalkan model, dilakukan ujicoba *hyperparameters*. Ujicoba dilakukan secara manual dengan menggunakan beberapa nilai *hyperparameters* berbeda dan dilanjutkan evaluasi nilai akurasi model. Berikut beberapa *hyperparameters* yang dapat diuji dalam model CNN.

1. *learning rate*

   Learning rate menentukan seberapa besar nilai parameter (bobot) dapat berubah terhadap nilai error selama proses pelatihan. Penggunaan *leraning rate* selama proses pembaruan bobot, dilakukan dengan Persamaan 2.88.

$$\mathbf{W}_{baru} = \mathbf{W} - \alpha \frac{\partial \mathrm{E}}{\partial \mathbf{W}} \tag{2.88}$$

   Nilai *learning rate* yang besar membantu mempercepat proses pelatihan mencapai target error, namun tidak menjamin menghasilkan nilai akurasi yang tinggi. Dalam proses pelatihan data yang lebih kompleks, penggunaan nilai *learning rate* yang kecil sering menghasilkan hasil akurasi yang lebih baik meski memerlukan waktu pelatihan yang lebih lama (Wilson, dkk , 2001).

2. *mini-batch size*

   Secara umum telah diketahui bahwa *mini-batch* dengan ukuran yang besar dapat mengurangi waktu pelatihan model. Namun ukuran *mini-batch* yang lebih kecil memiliki keuntungan karena dapat memberikan model pelatihan yang lebih *general*. Dalam artian ukuran *mini-batch* yang lebih kecil mampu menghasilkan hasil akurasi pengujian model yang lebih baik (Qian, dkk , 2020). Penggunaan *mini-batch* selama proses pembaruan bobot, dilakukan dengan Persamaan 2.89.

$$\mathbf{E}_{total} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{E}_i \tag{2.89}$$

2.7. Evaluasi

Proses evaluasi merupakan penilaian kinerja sistem yang dilakukan untuk mengetahui seberapa baik suatu sistem diterapkan dalam penyelesaian suatu masalah. Evaluasi yang paling sering digunakan dalam klasifikasi penyakit adalah dengan menganalisa nilai *sensitivity*, nilai *specificity*, dan nilai *accuracy*. Terlebih dahulu hasil klasifikasi disajikan dalam bentuk *confusion matrix* seperti pada tabel 2.2.

Tabel 2.2 *Confusion matrix* dengan *Multi Class*

Hasil aktual	Hasil prediksi	
	positif	*negatif*
positif	TP	FP
negatif	FN	TN

1. *True Positif* (TP)

   Merupakan jumlah data positif yang terklasifikasi benar kedalam kelas positif oleh sistem.

2. *False Positif* (FP)

   Merupakan jumlah data negatif yang terklasifikasi salah kedalam kelas positif oleh sistem.

3. *False Negative* (FN)

   Merupakan jumlah data positif yang terklasifikasi salah kedalam kelas negatif oleh sistem.

4. *True Negatif* (TN)

   Merupakan jumlah kelas negatif yang terklasifikasi benar kedalam kelas negatif oleh sistem (Habib dkk., 2015).

Untuk permasalahan kalasifikasi data yang memiliki banyak kelas *multi-class*, *confusion matrix* disajikan seperti dalam tabel 2.3.

Tabel 2.3 *Confusion matrix* dengan *Multi Class*

Actual Class	*Predicted Class*				
	c(1)	c(2)	c(3)	...	c(n)
c(1)	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,n}$
c(2)	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,n}$
c(3)	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{2,n}$
⋮	⋮	⋮	⋮	⋱	⋮
c(n)	$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,n}$

Nilai *true positif*, *false positif*, *false negatif* dan *true negatif* untuk setiap kelas, dapat dihitung dengan Persamaan 2.90 - 2.93.

$$TP_i = x_{1,1} \tag{2.90}$$

$$FN_i = \sum_{j=1}^{n} a_{i,j} \tag{2.91}$$

$$\text{dimana: } a_{i,j} = \begin{cases} x_{i,j}, & \text{jika } i \neq j \\ 0, & \text{lainnya} \end{cases}$$

$$FP_i = \sum_{j=1}^{n} b_{j,i} \tag{2.92}$$

$$\text{dimana: } b_{j,i} = \begin{cases} x_{j,i}, & \text{jika } j \neq i \\ 0, & \text{lainnya} \end{cases}$$

$$TN_i = \sum_{j=1}^{n} \sum_{k=1}^{n} c_{j,k} \tag{2.93}$$

$$\text{dimana: } c_{j,k} = \begin{cases} x_{jk}, & \text{jika } j \neq i \text{dan } k \neq i \\ 0, & \text{lainnya} \end{cases}$$

Selanjutnya dilakukan perhitungan nilai *sensitifity* dan *specificity* secara keseluruhan. Perhitungan ini dikaukan dengan mnghitung rata-rata dari total *sensitifity* dan *specificity* tiap kelas. Selain itu juga dilakukan perhitungan nilai akurasi sistem (Habib dkk. , 2015).

1. *Sensitifity*

   Sensitifity disebut juga sebagai *true positif rate*, merupakan hasil evaluasi tingkat keberhasilan sistem mengenali data positif. Dalam proses klasifikasi penyakit, *sensitifity* digunakan untuk menilai keakuratan sistem dalam mengenali data pasien dengan penyakit tertentu secara tepat pada kelas aktual. *Sensitifity* dihitung menggunakan Persamaan 2.94.

$$Sensitivity = \frac{\sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i}}{n} \times 100\% \tag{2.94}$$

2. *Specificity*

   Specificity disebut juga sebagai *true negative rate*, merupakan hasil evaluasi

tingkat keberhasilan sistem mengenali data negatif. Dalam proses klasifikasi penyakit, *specificity* digunakan untuk menilai keakuratan sistem dalam mengenali data pasien sehat secara tepat.

Dalam kasus klasifikasi penyakit dengan *multiclass*, *true negatif* untuk suatu jenis penyakit ”P” merupakan banyaknya pasien yang secara tepat diklasifikasikan tidak terkena penyakit ”P” dan *specificity* digunakan untuk menilai tingkat keberhasilan tersebut. NIlai *specificity* dihitung menggunakan Persamaan 2.95.

$$Specificity = \frac{\sum_{i=1}^{n} \frac{TN_i}{TN_i + FP_i}}{n} \times 100\% \tag{2.95}$$

3. Akurasi

   Akurasi merupakan tingkat keberhasilan suatu sistem klasifikasi dalam mengenali susatu data secara tepat. Perhitungan nilai akurasi dilakukan mnggunakan Persamaan 2.96.

$$Accuracy = \frac{\sum_{i=1}^{n} TP_i}{N} \times 100\% \tag{2.96}$$

2.8. Pandangan Islam dalam menyikapi penyakit

Sakit merupakan salah satu musibah yang paling sering dialami oleh manusia, baik diakibatkan oleh kurang-nya menjaga kesahatan tubuh, pengaruh cuaca, penyakit bawaan dan penyebab penyakit lainnya. Bahkan hingga kini masih banyak penyakit yang belum diketahui penyebabnya secara pasti. Disisi lain, setiap penyakit pasti ada obatnya sebagaimana sabda Nabi Muhammad SAW:

حَدَّثَنَا هَارُونُ بْنُ مَعْرُوفٍ وَأَبُو الطَّاهِرِ وَأَحْمَدُ بْنُ عِيسَى قَالُوا حَدَّثَنَا ابْنُ وَهْبٍ أَخْبَرَنِي عَمْرٌو وَهُوَ ابْنُ الْحَارِثِ عَنْ عَبْدِ رَبِّهِ بْنِ سَعِيدٍ عَنْ أَبِي الزُّبَيْرِ عَنْ جَابِرٍ عَنْ رَسُولِ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ أَنَّهُ قَالَ لِكُلِّ دَاءٍ دَوَاءٌ فَإِذَا أُصِيبَ دَوَاءُ الدَّاءِ بَرَأَ بِإِذْنِ اللَّهِ عَزَّ وَجَلَّ

Artinya: Telah menceritakan kepada kami Harun bin Maruf dan Abu Ath Thahir serta Ahmad bin Isa mereka berkata; Telah menceritakan kepada kami Ibnu Wahb; Telah mengabarkan kepadaku Amru, yaitu Ibnu al-Harits dari Abdu Rabbih bin Said dari Abu Az Zubair dari Jabir dari Rasulullah shallallahu alaihi wasallam, beliau bersabda: Setiap penyakit ada obatnya. Apabila ditemukan obat yang tepat untuk suatu penyakit, akan sembuhlah penyakit itu dengan izin Allah azza wajalla (HR Muslim).

Berdasarkan hadis tersebut, sebagai orang yang beriman seharusnya kita tetap menjaga ketenangan hati dan pikiran kita karena pasti setiap penyakit yang diderita ada obatnya. Selain itu, orang-orang yang beriman sudah sepantasnya meyakini semua musibah itu terjadi atas izin Allah SWT dan sepatutnya senantiasa bersabar dalam menghadapinya. Allah SWT berfirman dalam surah at-Taghabun, ayat 11:

مَآ أَصَابَ مِن مُّصِيبَةٍ إِلَّا بِإِذْنِ ٱللَّهِ ۗ وَمَن يُؤْمِنۢ بِٱللَّهِ يَهْدِ قَلْبَهُۥ ۚ وَٱللَّهُ بِكُلِّ شَىْءٍ عَلِيمٌ

Artinya: Tidak ada suatu musibah pun yang menimpa seseorang kecuali dengan izin Allah; dan barangsiapa yang beriman kepada Allah niscaya Dia akan memberi petunjuk kepada hatinya. Dan Allah Maha Mengetahui segala sesuatu (QS.at-Taghabun, ayat 11).

Menyikapi setiap penyakit yang diderita, tidaklah dengan cara pasrah tanpa

melakukan apapun. Disisi lain seorang muslim harus senantiasa mencari solusi melalui petunjuk-petunjuk yang diberikan oleh Allah SWT. Sesungguhnya pada segala yang Ia ciptakan di-langit, di-bumi dan terutama pada diri tiap-tiap manusia, terdapat petunjuk tentang cara menyikapi setiap penyakit maupun musibah lainnya yang dihadapi. Sebagaimana firman Allah SWT:

أَلَمْ تَرَ أَنَّ ٱللَّهَ أَنزَلَ مِنَ ٱلسَّمَآءِ مَآءً فَسَلَكَهُۥ يَنَٰبِيعَ فِى ٱلْأَرْضِ ثُمَّ يُخْرِجُ بِهِۦ زَرْعًا مُّخْتَلِفًا أَلْوَٰنُهُۥ ثُمَّ يَهِيجُ فَتَرَىٰهُ مُصْفَرًّا ثُمَّ يَجْعَلُهُۥ حُطَٰمًا ۚ إِنَّ فِى ذَٰلِكَ لَذِكْرَىٰ لِأُو۟لِى ٱلْأَلْبَٰبِ

Artinya: Adakah kamu tidak memperhatikan, bahwa sesungguhnya Allah menurunkan air dari langit, maka diaturnya menjadi sumber-sumber air di bumi kemudian ditumbuhkan-Nya dengan air itu tanaman-tanaman yang bermacam-macam warnanya, lalu menjadi kering, lalu kamu melihatnya kekuning-kuningan, kemudian dijadikan hancur berderai-derai. Sesungguhnya pada yang demikian itu benar-benar terdapat pelajaran bagi orang yang mempunyai akal (QS. Az Zumar: 21).

Oleh karenanya sebagai seorang yang beriman, tidaklah pantas untuk bersikap acuh dalam usaha mencari solusi atas musibah yang dideritanya. Tugas manusia yang dianugrahkan kemampuan berfikir oleh Allah SWT adalah untuk lebih memperhatikan petunjuk tersebut, agar dapat memperoleh solusi yang berguna untuk dirinya sendiri dan orang lain. Rasullullah SAW, bersabda:

مَنْ سَنَّ فِي الْإِسْلَامِ سُنَّةً حَسَنَةً فَلَهُ أَجْرُهَا وَأَجْرُ مَنْ عَمِلَ بِهَا بَعْدَهُ مِنْ غَيْرِ أَنْ يَنْقُصَ مِنْ أُجُورِهِمْ شَيْءٌ وَمَنْ سَنَّ فِي الْإِسْلَامِ سُنَّةً سَيِّئَةً كَانَ عَلَيْهِ وِزْرُهَا وَوِزْرُ مَنْ عَمِلَ بِهَا مِنْ بَعْدِهِ مِنْ غَيْرِ أَنْ يَنْقُصَ مِنْ أَوْزَارِهِمْ شَيْءٌ

Artinya: Rasulullah s.a.w. bersabda, barang siapa menciptakan kebaikan, maka baginya pahalanya dan pahala orang yang mengikuti setelahnya tanpa dikurangi sedikitpun dari pahala tersebut. Dan barang siapa berbuat kejelekan, maka baginya dosanya dan dosa orang yang mengikuti setelahnya tanpa dikurangi sedikitpun.

Disisi lainnya konsekuensi dari sikap acuh akan petunjuk yang diberikan oleh Allah SWT tidak hanya menyebabkan seseorang kembali ditimpa penyakit ataupun musibah yang pernah dihadapinya, namun juga dapat menyebabkan penyesalan di akhirat kelak. Allah SWT berfirman:

وَقَالُوا۟ لَوْ كُنَّا نَسْمَعُ أَوْ نَعْقِلُ مَا كُنَّا فِىٓ أَصْحَٰبِ ٱلسَّعِيرِ

Artinya: Dan mereka berkata, Sekiranya (dahulu) kami mendengarkan atau memikirkan (peringatan itu) tentulah kami tidak termasuk penghuni neraka yang menyala-nyala. (QS. Al-Mulk Ayat 10).

Jalan terbaik untuk mengatasi setiap masalah adalah dengan menuntut ilmu. Dengan ilmu manusia dapat memahami petunjuk-petunjuk yang diberikan Allah SWT. dalam menggapai kebahagiaan hidup di dunia dan akhirat. Rasullullah SAW, bersabda:

مَنْ أَرَادَ الدُّنْيَا فَعَلَيْهِ بِالْعِلْمِ وَمَنْ أَرَادَ الآخِرَةَ فَعَلَيْهِ بِالْعِلْمِ وَمَنْ أَرَادَهُمَا فَعَلَيْهِ بِالْعِلْمِ

Artinya: Barangsiapa menginginkan dunia maka capailah dengan ilmu, Barangsiapa menginginkan akhirat maka capailah dengan ilmu dan Barangsiapa menginginkan keduanya maka capailah dengan ilmu (H.R. Thabrani).

BAB III

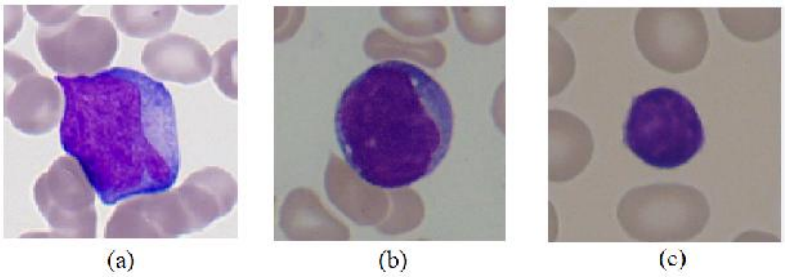
METODE PENELITIAN

3.1. Jenis Penelitian

Penelitian mengenai klasifikasi penyakit leukemia menggunakan convolution neural network dengan arsitektur inception V3 termasuk dalam jenis penelitian terapan. Penelitian ini dikategorikan sebagai jenis penelitian terapan berdasarkan atas tujuan dari penelitian ini, yakni sebagai alternatif deteksi penyakit secara optimal, dalam artian dilakukan secara cepat dan tepat. Hal ini sesuai dengan tujuan dari suatu jenis penelitian terapan, yakni untuk memberikan solusi atas permasalahan tertentu secara praktis.

3.2. Data Penelitian

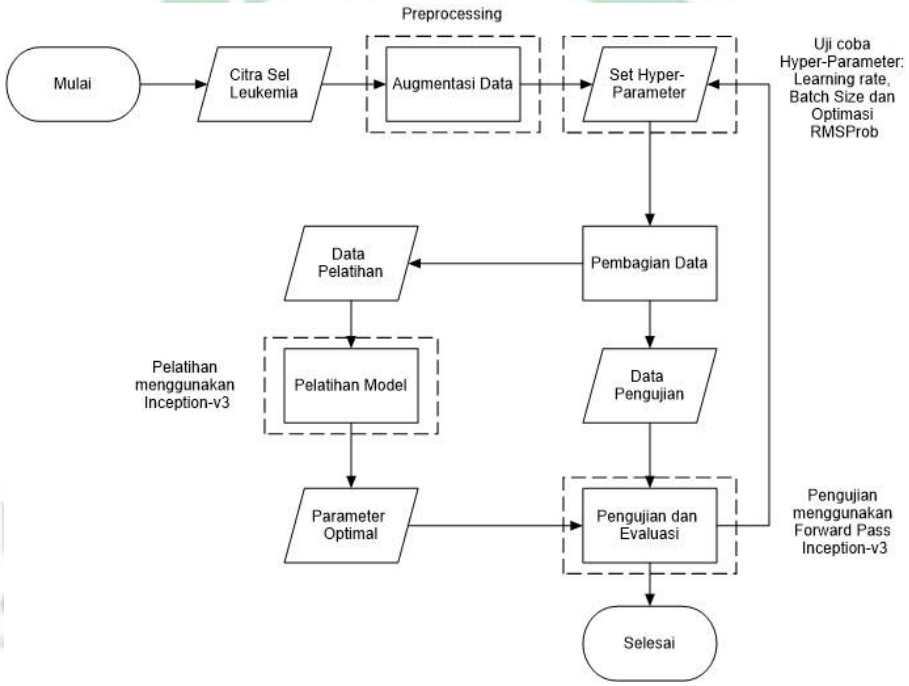
Data yang digunakan dalam penelitian ini merupakan kumpulan citra sel tunggal penyakit leukemia, terdiri atas Acute Myeloblastic Leukemia (AML), Acute Lymphoblastic Leukemia (ALL) dan sel normal yang diambil menggunakan mikroskop optik laboratori. Data ini diperoleh dari data yang dipublish oleh Fabio Scotti dkk (Labati, Donida dkk. , 2011), yang berisi kumpulan citra sel tunggal ALL dan sel normal. serta data yang dipublish oleh Natasha Honomichl (Matek dkk. , 2019), yang berisi kumpulan citra sel tunggal AML. Sebanyak 390 citra digunakan dalam penelitian ini dengan jumlah tiap kelas diseragamkan sebanyak 130 citra.



Gambar 3.1 Citra mikroskopis sel tunggal leukemia. (a) Sel ALL, (b) Sel normal (c) el AML

(Labati, Donida dkk. , 2011)

3.3. Tahapan Penelitian



Gambar 3.2 Diagram alir penelitian

Diagram alir diatas menunjukkan alur proses klasifikasi citra penyakit leukemia menggunakan metode CNN inceptionV3 secara umum. Lebih jelasnya, proses tersebut dilakukan sebegai berikut:

1. Input data

Sebanyak 390 data citra leukemia yang telah dikategorikan dalam tiga kelas ALL, AML dan normal, di-*input* kedalam sistem. Selanjutnya seluruh citra akan diubah ukurann-hya sesuai arsitektur Inception-v3, yakni 299x299 serta dibagi menjadi data pelatihan dan data uji.

2. Augmentasi

   Proses augmentasi data citra termasuk kedalam tahap *pre-processing*, dilakukan untuk memperoleh lebih banyak keragaman data sehingga memperoleh kinerja pelatihan yang lebih baik. Proses augmentasi yang diterapkan dalam penelitian ini yakni: refleksi, translasi dan rotasi.

3. Pembagian data

   Pembagian data untuk pelatihan dan pengujian, dilakukan dengan rasio perbandingan 3:1. Sehingga 75% data digunakan untuk pelatihan dan 25% data digunakan dalam proses pengujian. Pembagian ini mengikuti penelitian sebelumnya yang dilakukan oleh (Huang dkk. , 2020).

4. Pelatihan dan pengujian

   Proses pelatihan bertujuan untuk memperoleh model yang secara umum dapat digunakan untuk beragam data baru dan tidak dikenali. Proses ini dilakukan dengan memperbarui nilai *learnable parameter* tiap layer menggunakan metode *gradient descent*, hingga mendapatkan model optimal. Selanjutnya model optimal hasil pelatihan digunakan pada proses pengujian serta dilakukan evaluasi hasil.

   Dalam penelitian ini proses pelatihan dan pengujian dilakukan denan metode CNN arsitektur Inception-v3. Proses pembelajaran fitur dalam metode CNN dilakukan melalui banyak layers, yakni: *convolution layers*, *Batch Normalization layers*, *Pooling layers* dan *ReLu Layers*. Selanjutnya proses klasifikasi dilakukan

pada *Fully Conected layer*.

5. Uji coba hyper-parameter

   Untuk mendapatkan model terbaik, dilakukan beberapa uji coba *hyper-parameter* sistem. Diantaranya: Inisialisasi *learning rate*, ukuran *mini-batch* dan penggunaan RMSProp Optimization. Pada saat proses pelatihan pertama, nilai *leraning rate* yang digunakan adalah sebesar 0.001 dan ukuran *mini-batch* yang digunakan sebesar 16 (Huang dkk. , 2020).

   i. *Learning rate* dengan nilai yang besar, digunakan untuk mempercepat proses training. Namun nilai *learning rate* yang besar dapat mengakibatkan *convergence*. Nilai learning rate yang kecil digunakan untuk menghindari *convergence* dan menghasilkan *smooth path* pada grafik loss (Wilson, dkk , 2001). Sehingga pada penelitian ini dilakukan uji coba *leraning rate* sebesar 0.1, 0.01, 0.001 dan 0.0001.

   ii. Penggunaan *mini-batch* yang terlalu banyak dapat menyebabkan terjadinya *overfitting*. *Overfitting* adalah kondisi dimana sebuah sistem memiliki hasil pelatihan dengan akurasi tinggi, namun memiliki akurasi pengujian yang rendah (Qian, dkk , 2020). Sehingga dibutuhkan ujicoba ukuran *mini-batch*. Pada penelitian ini dilakukan uji coba ukuran *mini-batch* sebesar 8, 16, 24 dan 32.

   iii. RMSProp Optimization membagi *learning rate* dengan sebuah *exponentially decaying average*. Dalam penelitian (Kingma dan Jimmy , 2014) disebutkan bahwa *learning rate* sebesar 0.001 dan *decay rate* sebesar 0.9 merupakan inisialisasi yang baik digunakan. namun dalam penelitian tersebut juga dilakukan ujicoba nilai *decay rate* sebesar 0.999.

   Dalam penelitian ini digunakan beragam nilai *learning rate*. Sehingga untuk

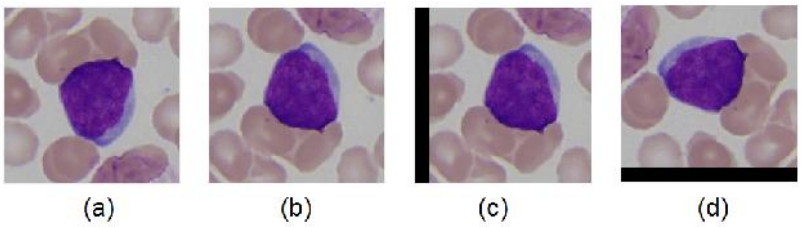
melihat pengaruh penggunaan metode optimasi RMSProp Optimization, dalam penelitian ini dilakukan ujicoba *decay rate* sebesar 0.9, 0.99 dan 0.999.

BAB IV

HASIL DAN PEMBAHASAN

4.1. *Augmentasi*

Proses *augmentasi image* merupakan tahap pertama dari proses klasifikasi citra penyakit leukemia menggunakan CNN. Dalam penelitian ini digunakan metode *random transformation*, sehingga setiap citra memiliki kemungkinan untuk dirubah menjadi sejumlah variasi. Beberapa bentuk transformasi yang digunakan yakni: *reflection* secara horizontal dan vertikal, *translation* secara horizontal dan vertikal dengan range [-30 30], serta *rotation* diantara interval [90 270]. Penggunaan metode ini menghasilkan beragam variasi data baru, sebanyak 3.120 citra.



Gambar 4.1 Hasil augmentasi yang dilakukan mulai dari (a) Gambar asli, (b) refleksi, (c) translasi dan (d) rotasi

Gambar 4.1 menampilkan hasil augmentasi salah satu *input image* dengan transformasi yang dilakukan secara berurutan: *reflection* secara horizontal, *translation* secara vertikal dengan *range* 20 dan *rotation* 90. Berikut merupakan perhitungan proses augmentasi image:

1. Refleksi secara horizontal

Proses refleksi secara horizontal menggunakan Persamaan 2.4, dilakukan seperti berikut:

$$\mathrm{F}^{(0)}(x,y) = \mathrm{I}_{(\mathrm{M}+1)-x,y}$$

$$\begin{aligned}\mathrm{F}^{(0)}_{1,1} &= \mathrm{I}_{(299+1)-1,1}\\ &= \mathrm{I}_{299,1}\end{aligned}$$

$$\begin{aligned}\mathrm{F}^{(0)}_{2,1} &= \mathrm{I}_{(299+1)-2,1}\\ &= \mathrm{I}_{298,1}\end{aligned}$$

$$\vdots$$

$$\begin{aligned}\mathrm{F}^{(0)}_{299,299} &= \mathrm{I}_{(299+1)-299,299}\\ &= \mathrm{I}_{1,299}\end{aligned}$$

2. Translasi secara vertikal dengan range 20

Proses translasi secara vertikal menggunakan Persamaan 2.6, dilakukan seperti berikut:

$$\mathrm{I}(x,y) \Rightarrow \mathrm{I}'(x,y+k)$$

$$\mathrm{I}'(x,y) \Rightarrow \mathrm{I}(x,y-k)$$

$$\mathrm{F}_{x,y}^{(0)} = \begin{cases} \mathrm{I}_{x,y-k} & \text{jika } y > k \\ 0 & \text{selainnya} \end{cases}$$

$$\begin{aligned} \mathrm{F}_{1,1}^{(0)} &= 0 \\ \mathrm{F}_{2,1}^{(0)} &= 0 \\ &\vdots \\ \mathrm{F}_{299,299}^{(0)} &= \mathrm{I}_{299,299-20} \\ &= \mathrm{I}_{299,279} \end{aligned}$$

3. Rotasi sebesar $90°$

Proses rotasi menggunakan Persamaan 2.7, dilakukan seperti berikut:

$$a = \frac{\mathrm{M}_1 + 1}{2} = \frac{299 + 1}{2} = 150$$
$$b = \frac{\mathrm{M}_2 + 1}{2} = \frac{299 + 1}{2} = 150$$

$$\mathrm{I}'(x,y) = \begin{bmatrix} \text{x'} \\ \text{y'} \end{bmatrix} = \begin{bmatrix} cos90° & -sin90° \\ sin90° & cos90° \end{bmatrix} \begin{bmatrix} \text{x-a} \\ \text{y-b} \end{bmatrix} + \begin{bmatrix} \text{a} \\ \text{b} \end{bmatrix}$$
$$= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \text{x-150} \\ \text{y-150} \end{bmatrix} + \begin{bmatrix} \text{a} \\ \text{b} \end{bmatrix}$$

$$\begin{bmatrix} 1' \\ 1' \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1\text{-}150 \\ 1\text{-}150 \end{bmatrix} + \begin{bmatrix} 150 \\ 150 \end{bmatrix}$$

$$= \begin{bmatrix} 299 \\ 1 \end{bmatrix}$$

$$F_{1,1}^{(0)} = I_{299,1}$$

$$\begin{bmatrix} 2' \\ 1' \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2\text{-}150 \\ 1\text{-}150 \end{bmatrix} + \begin{bmatrix} 150 \\ 150 \end{bmatrix}$$

$$= \begin{bmatrix} 299 \\ 2 \end{bmatrix}$$

$$F_{2,1}^{(0)} = I_{299,2}$$

$$\vdots$$

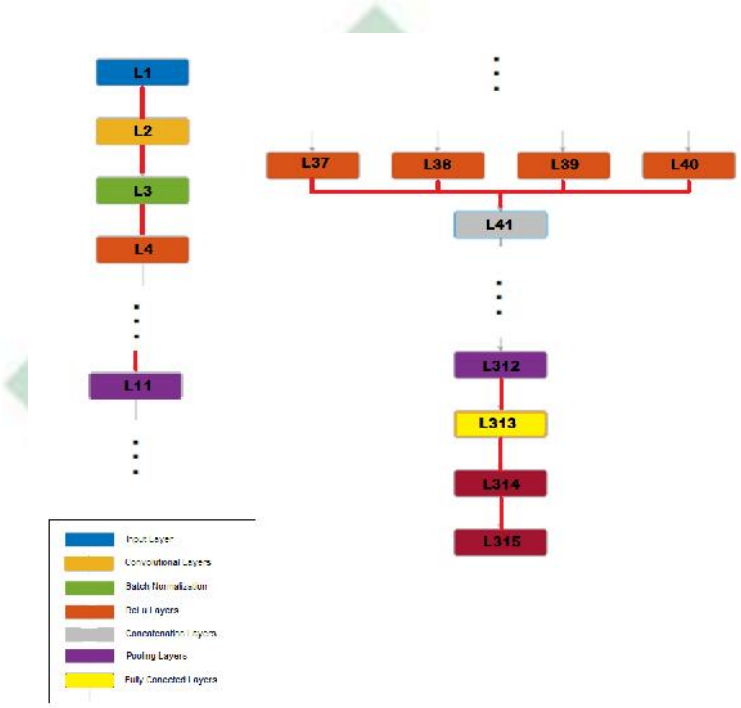
$$\begin{bmatrix} 299' \\ 299' \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 299\text{-}150 \\ 299\text{-}150 \end{bmatrix} + \begin{bmatrix} 150 \\ 150 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 299 \end{bmatrix}$$

$$F_{299,299}^{(0)} = I_{1,299}$$

4.2. Klasifikasi Menggunakan Metode CNN Jenis InceptionV3

Dalam proses klasifikasi ini, dataset dikelompokan sebagai data latih dan data test. Rasio banyaknya data latih dan data test secara berturut-turut adalah 3 : 1. Selanjutnya dilakukan proses pembelajaran fitur dan klasifikasi menggunakan metode Inception-v3, melewati beragam jenis layer.

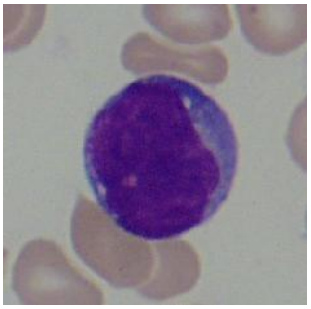


Gambar 4.2 Jalur perhitungan *forward pass* Inception-v3

Proses perhitungan *forward pass* melalui layers yang ditunjukkan dalam Gambar 4.2 dilakukan sebagai berikut:

1. Input layer

   Setelah dilakukan proses augmentasi, Image terlebih dahulu diproses sebagai input ($F^{(n)(0)}$) pada Input layer. Untuk memudahkan proses perhitungan, hanya digunakan sebuah image seperti yang ditampilkan pada Gambar 4.3.



Gambar 4.3 Citra sel tunggal leukemia yang digunakan sebagai input image

Pada layer pertama arsitektur Inception-v3, image dinormalisasi menggunakan Persamaan 2.13.

$$F_{x,y}^{(n)(1)} = (-1) + \frac{F_{x,y}^{(n)(0)}}{255} \times 2$$

$$\begin{aligned} F_{1,1}^{(1)(1)} &= (-1) + \frac{F_{1,1}^{(1)(0)}}{255} \times 2 \\ &= (-1) + \frac{139}{255} \times 2 \\ &= 0.0902 \end{aligned}$$

$$\begin{aligned} F_{2,1}^{(1)(1)} &= (-1) + \frac{F_{2,1}^{(1)(0)}}{255} \times 2 \\ &= (-1) + \frac{142}{255} \times 2 \\ &= 0.1137 \end{aligned}$$

$$\vdots$$

$$\begin{aligned} F_{299,299}^{(3)(1)} &= (-1) + \frac{F_{299,299}^{(3)(0)}}{255} \times 2 \\ &= (-1) + \frac{164}{255} \times 2 \\ &= 0.2863 \end{aligned}$$

Hasil Normalisasi pada input layer ini, seperti yang ditampilkan dalam matrix:

$$\mathrm{F}^{(1)(1)} = \begin{bmatrix} 0.0902 & 0.1059 & 0.1216 & 0.1059 & 0.0824 & \cdots & 0.1294 \\ 0.1137 & 0.1137 & 0.1059 & 0.1059 & 0.0902 & \cdots & 0.1137 \\ 0.0510 & 0.0588 & 0.0667 & 0.0667 & 0.0745 & \cdots & 0.1529 \\ 0.0353 & 0.0353 & 0.0510 & 0.0667 & 0.0745 & \cdots & 0.1608 \\ 0.0588 & 0.0588 & 0.0667 & 0.0745 & 0.0824 & \cdots & 0.1529 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.2157 & 0.2784 & 0.2784 & 0.2627 & 0.2549 & \cdots & 0.2627 \end{bmatrix}$$

$$\mathrm{F}^{(2)(1)} = \begin{bmatrix} -0.0588 & -0.0510 & -0.0353 & -0.0431 & -0.0667 & \cdots & 0.0039 \\ -0.0275 & -0.0431 & -0.0588 & -0.0510 & -0.0588 & \cdots & 0.0039 \\ -0.0431 & -0.0431 & -0.0510 & -0.0588 & -0.0588 & \cdots & 0.0118 \\ -0.0588 & -0.0510 & -0.0510 & -0.0510 & -0.0510 & \cdots & 0.0118 \\ -0.0588 & -0.0588 & -0.0588 & -0.0588 & -0.0588 & \cdots & 0.0039 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.2549 & 0.3098 & 0.3098 & 0.3020 & 0.3098 & \cdots & 0.2941 \end{bmatrix}$$

$$F^{(3)(1)} = \begin{bmatrix} -0.0431 & -0.0353 & -0.0196 & -0.0275 & -0.0510 & \cdots & 0.0118 \\ -0.0275 & -0.0431 & -0.0431 & -0.0353 & -0.0431 & \cdots & -0.0039 \\ -0.0510 & -0.0510 & -0.0431 & -0.0510 & -0.0667 & \cdots & 0.0196 \\ -0.0588 & -0.0588 & -0.0510 & -0.0588 & -0.0667 & \cdots & 0.0275 \\ -0.0510 & -0.0510 & -0.0431 & -0.0431 & -0.0431 & \cdots & 0.0196 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.2235 & 0.2941 & 0.3020 & 0.2706 & 0.2549 & \cdots & 0.2863 \end{bmatrix}$$

2. Convolution layers

Untuk melakukan perhitungan pada convolution layer, terlebih dahulu dilakukan inisialisasi pada beberapa parameter. yakni: nilai-nilai bobot dan bias, banyaknya *feature maps*, nilai stride, nilai padding dan ukuran kernel. Berikut diberikan nilai-nilai *hyper-parameter* dari layer ke-2 arsitektur Inception-v3.

Nama parameter	Nilai
Ukuran kernel (K)	[3 3]
Banyaknya *feature maps output* ($N^{(\ell)}$)	32
Banyaknya *feature maps input* ($N^{(\ell-1)}$)	3
Stride (s)	2
Padding (p)	0

Nilai-nilai bobot (w) berbentuk 4D dengan ukuran $K_1 \times K_2 \times N^{(\ell-1)} \times N^{(\ell)}$ = 3 x 3 x 3 x 32. Sedangkan bias (b) diinisialisasi sebagai matrix zeros dengan

ukuran 1 x 1 x $N^{(\ell)} = 1 \times 1 \times 32$. Ukuran bobot dan bias telah disesuaikan dengan ukuran kernel, *feature maps Input* dan *feature maps Output*. Berikut inisialisasi nilai-nilai bobot Kernel serta bias:

$$\mathbf{W}^{(1)(1)(2)} = \begin{bmatrix} -0.4591 & -0.7513 & -0.0753 \\ -0.3097 & -0.5500 & -0.0613 \\ 0.1612 & 0.0406 & 0.0836 \end{bmatrix}$$

$$\mathbf{W}^{(2)(1)(2)} = \begin{bmatrix} 0.3164 & 0.3558 & 0.1129 \\ 0.1852 & 0.1909 & 0.0415 \\ 0.0639 & 0.0331 & 0.1144 \end{bmatrix}$$

$$\mathbf{W}^{(3)(1)(2)} = \begin{bmatrix} 0.2217 & 0.5110 & -0.0730 \\ 0.1602 & 0.3693 & -0.0263 \\ -0.2275 & -0.0537 & -0.1941 \end{bmatrix}$$

$$\vdots$$

$$\mathbf{W}^{(3)(32)(2)} = \begin{bmatrix} -0.6191 & -0.1051 & 0.2883 \\ -0.2944 & 0.0072 & 0.2747 \\ 0.0850 & 0.1461 & 0.1482 \end{bmatrix}$$

$$b^{(2)} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

Berdasarkan parameter-parasmeter yang telah diinisialisasi, ukuran *feature maps* yang dihasilkan pada layer ini, dapat dihitung menggunakan Persamaan 2.19.

$$\mathrm{M} = \frac{i - k + 2p}{s} + 1$$
$$\mathrm{M} = \frac{299 - 3 + 2(0)}{2} + 1 = 149$$

Setelah dilakukan proses inisialisasi, selanjutnya dilakukan perhitungan pada convolusi layers menggunakan Persamaan 2.20:

$$\mathrm{F}_{x,y}^{(n)(\ell)} = \sum_{m=1}^{\mathrm{N}^{\ell-1}} \sum_{a=1}^{\mathrm{K}_1} \sum_{b=1}^{\mathrm{K}_2} w_{a,b}^{(m)(n)} \, \mathrm{F}_{i-a+1,j-b+1}^{(m)(\ell-1)} + b^{(n)}$$

1. perhitungan *maps* pertama $(n = 1)$, koordinat spatial $(x, y) = (1, 1)$:

F

0.0902	0.1059	0.1216	0.1059	...
0.1137	0.1137	0.1059	0.1059	...
0.051	0.0588	0.0667	0.0667	...
0.0353	0.0353	0.051	0.0667	...
0.0588	0.0588	0.0667	0.0745	...
0.0902	0.0902	0.0824	0.0824	...
0.098	0.098	0.098	0.098	...
...	...	...	...	...

n = 1

-0.0588	-0.051	-0.0353	-0.0431	...
-0.0275	-0.0431	-0.0588	-0.051	...
-0.0431	-0.0431	-0.051	-0.0588	...
-0.0588	-0.051	-0.051	-0.051	...
-0.0588	-0.0588	-0.0588	-0.0588	...
-0.0667	-0.0667	-0.0667	-0.0667	...
-0.0667	-0.0667	-0.0667	-0.0588	...
...	...	...	...	...

n = 2

-0.0431	-0.0353	-0.0196	-0.0275	...
-0.0275	-0.0431	-0.0431	-0.0353	...
-0.051	-0.051	-0.0431	-0.051	...
-0.0588	-0.0588	-0.051	-0.0588	...
-0.051	-0.051	-0.0431	-0.0431	...
-0.0353	-0.0353	-0.0353	-0.0275	...
-0.0275	-0.0275	-0.0275	-0.0196	...
...	...	...	...	...

n = 3

w

$*$

-0.4591	-0.7513	-0.0753
-0.3097	-0.55	-0.0613
0.1612	0.0406	0.0836

$*$

0.3164	0.3558	0.1129
0.1852	0.1909	0.0415
0.0639	0.0331	0.1144

$*$

0.2217	0.511	-0.073
0.1602	0.3693	-0.0263
-0.2275	-0.0537	-0.1941

$$\mathrm{F}_{1,1}^{(1)(2)} = \sum_{m=1}^{\mathrm{N}^3} \sum_{a=1}^{3} \sum_{b=1}^{3} w_{a,b}^{(1)(2)} \, \mathrm{F}_{2-a,2-b}^{(m)(1)} + b^{(1)}$$

$$= ((-0.4591) \times 0.0902 + (-0.3097) \times 0.1137 + 0.1612 \times 0.0510 +$$

$$(-0.7513) \times 0.1059 + (-0.5500) \times 0.1137 + 0.0406 \times 0.0588 + (-0.0753) \times$$

$$0.1216(-0.0613) \times 0.10590.0836 \times 0.0667) +$$

$$(0.3164 \times (-0.0588) + 0.1852 \times (-0.0275) + 0.0639 \times (-0.0431) +$$

$$0.3558 \times (-0.0510) + 0.1909 \times (-0.0431) + 0.0331 \times (-0.0431) +$$

$$0.1129 \times (-0.0353)0.0415 \times (-0.0588)0.1144 \times (-0.0510)) +$$

$$(0.2217 \times (-0.0431) + 0.1602 \times (-0.0275) + (-0.2275) \times (-0.0510) +$$

$$0.5110 \times (-0.0353) + 0.3693 \times (-0.0431) + (-0.0537) \times (-0.0510) +$$

$(-0.0730)\times(-0.0196)(-0.0263)\times(-0.0431)(-0.1941)\times(-0.0431))+$

0

$= -0.3074$

2. perhitungan *maps* pertama $(n = 1)$, koordinat spatial $(x, y) = (2, 1)$:

F

0.0902	0.1059	0.1216	0.1059	...
0.1137	0.1137	0.1059	0.1059	...
0.051	0.0588	0.0667	0.0667	...
0.0353	0.0353	0.051	0.0667	...
0.0588	0.0588	0.0667	0.0745	...
0.0902	0.0902	0.0824	0.0824	...
0.098	0.098	0.098	0.098	...
...	...	...	...	...

n = 1

-0.0588	-0.051	-0.0353	-0.0431	...
-0.0275	-0.0431	-0.0588	-0.051	...
-0.0431	-0.0431	-0.051	-0.0588	...
-0.0588	-0.051	-0.051	-0.051	...
-0.0588	-0.0588	-0.0588	-0.0588	...
-0.0667	-0.0667	-0.0667	-0.0667	...
-0.0667	-0.0667	-0.0667	-0.0588	...
...	...	...	...	...

n = 2

-0.0431	-0.0353	-0.0196	-0.0275	...
-0.0275	-0.0431	-0.0431	-0.0353	...
-0.051	-0.051	-0.0431	-0.051	...
-0.0588	-0.0588	-0.051	-0.0588	...
-0.051	-0.051	-0.0431	-0.0431	...
-0.0353	-0.0353	-0.0353	-0.0275	...
-0.0275	-0.0275	-0.0275	-0.0196	...
...	...	...	...	...

n = 3

w

-0.4591	-0.7513	-0.0753
-0.3097	-0.55	-0.0613
0.1612	0.0406	0.0836

$*$

0.3164	0.3558	0.1129
0.1852	0.1909	0.0415
0.0639	0.0331	0.1144

$*$

0.2217	0.511	-0.073
0.1602	0.3693	-0.0263
-0.2275	-0.0537	-0.1941

$*$

$$\mathrm{F}_{2,1}^{(1)(2)} = \sum_{m=1}^{\mathrm{N}^3} \sum_{a=1}^{3} \sum_{b=1}^{3} w_{a,b}^{(1)(2)} \, \mathrm{F}_{4-a,2-b}^{(m)(1)} + b^{(1)}$$

$= ((-0.4591) \times 0.0510 + (-0.3097) \times 0.0353 + 0.1612 \times 0.0588 +$

$(-0.7513)\times0.0588+(-0.5500)\times0.0353+0.0406\times0.0588+(-0.0753)\times$

$0.0667(-0.0613) \times 0.05100.0836 \times 0.0667) +$

$$(0.3164 \times (-0.0431) + 0.1852 \times (-0.0588) + 0.0639 \times (-0.0588) +$$
$$0.3558 \times (-0.0431) + 0.1909 \times (-0.0510) + 0.0331 \times (-0.0588) +$$
$$0.1129 \times (-0.0510)0.0415 \times (-0.0510)0.1144 \times (-0.0588)) +$$
$$(0.2217 \times (-0.0510) + 0.1602 \times (-0.0588) + (-0.2275) \times (-0.0510) +$$
$$0.5110 \times (-0.0510) + 0.3693 \times (-0.0588) + (-0.0537) \times (-0.0510) +$$
$$(-0.0730) \times (-0.0431)(-0.0263) \times (-0.0510)(-0.1941) \times (-0.0431)) +$$
$$0$$
$$= -0.1999$$

$$\vdots$$

3. perhitungan *maps* pertama $(n = 1)$, koordinat spatial $(x, y) = (149, 149)$:

F

0.0902	0.1059	0.1216	0.1059	...
0.1137	0.1137	0.1059	0.1059	...
0.051	0.0588	0.0667	0.0667	...
0.0353	0.0353	0.051	0.0667	...
0.0588	0.0588	0.0667	0.0745	...
0.0902	0.0902	0.0824	0.0824	...
0.098	0.098	0.098	0.098	...
...	...	...	...	...

n = 1

-0.0588	-0.051	-0.0353	-0.0431	...
-0.0275	-0.0431	-0.0588	-0.051	...
-0.0431	-0.0431	-0.051	-0.0588	...
-0.0588	-0.051	-0.051	-0.051	...
-0.0588	-0.0588	-0.0588	-0.0588	...
-0.0667	-0.0667	-0.0667	-0.0667	...
-0.0667	-0.0667	-0.0667	-0.0588	...
...	...	...	...	...

n = 2

-0.0431	-0.0353	-0.0196	-0.0275	...
-0.0275	-0.0431	-0.0431	-0.0353	...
-0.051	-0.051	-0.0431	-0.051	...
-0.0588	-0.0588	-0.051	-0.0588	...
-0.051	-0.051	-0.0431	-0.0431	...
-0.0353	-0.0353	-0.0353	-0.0275	...
-0.0275	-0.0275	-0.0275	-0.0196	...
...	...	...	...	...

n = 3

w

*	-0.4591	-0.7513	-0.0753
	-0.3097	-0.55	-0.0613
	0.1612	0.0406	0.0836

*	0.3164	0.3558	0.1129
	0.1852	0.1909	0.0415
	0.0639	0.0331	0.1144

*	0.2217	0.511	-0.073
	0.1602	0.3693	-0.0263
	-0.2275	-0.0537	-0.1941

$$\mathrm{F}_{149,149}^{(1)(2)} = \sum_{m=1}^{\mathrm{N}^3} \sum_{a=1}^{3} \sum_{b=1}^{3} w_{a,b}^{(1)(2)} \, \mathrm{F}_{298-a,298-b}^{(m)(1)} + b^{(1)}$$

$$= ((-0.4591) \times 0.0824 + (-0.3097) \times 0.0824 + 0.1612 \times 0.0902 +$$

$$(-0.7513) \times 0.0902 + (-0.5500) \times 0.0824 + 0.0406 \times 0.0824 + (-0.0753) \times$$

$$0.0980(-0.0613) \times 0.09020.0836 \times 0.0902) +$$

$$(0.3164 \times (-0.0588) + 0.1852 \times (-0.0667) + 0.0639 \times (-0.0588) +$$

$$0.3558 \times (-0.0588) + 0.1909 \times (-0.0667) + 0.0331 \times (-0.0667) +$$

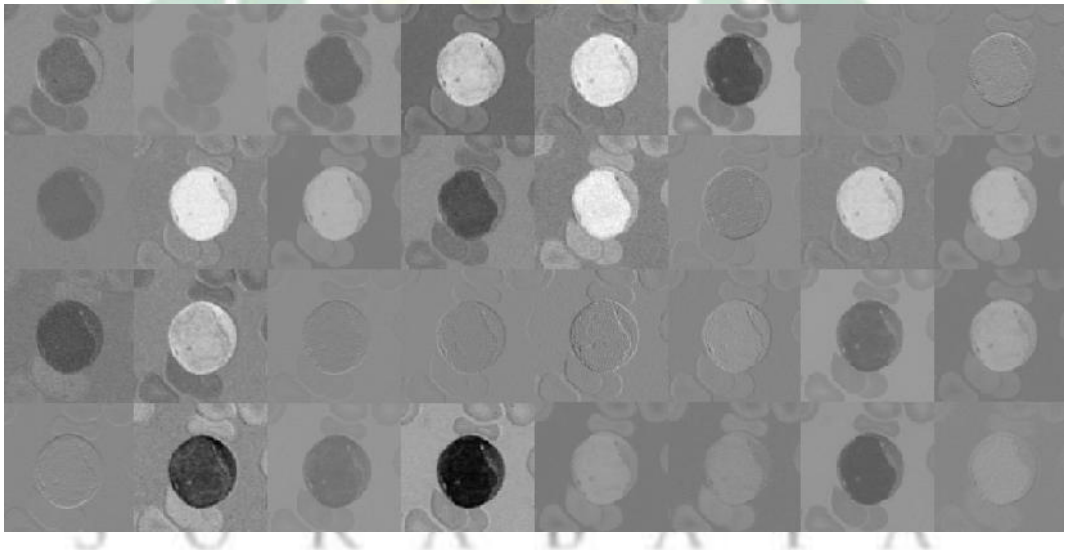
$$0.1129 \times (-0.0431)0.0415 \times (-0.0510)0.1144 \times (-0.0588)) +$$

$$(0.2217 \times (-0.0431) + 0.1602 \times (-0.0275) + (-0.2275) \times (-0.0118) +$$

$$0.5110 \times (-0.0431) + 0.3693 \times (-0.0353) + (-0.0537) \times (-0.0275) +$$

$$(-0.0730)\times(-0.0588)(-0.0263)\times(-0.0510)(-0.1941)\times(-0.0431))+$$
$$0$$
$$= -0.2790$$

Perhitungan ini dilakukan hingga memperoleh *feature maps* terakhir ($F^{(32)(2)}$), sesuai dengan nilai banyaknya *feature maps output* yang ditetapkan (N = 32). Proses convolusi pada layer ini menghasilkan *feature maps* dangan ukuran $149 \times 149 \times 32$. Pada Gambar 4.4 diperlihatkan tampilan seluruh *feature maps* yang dihasilkan pada layer ini.



Gambar 4.4 Tampilan *feature maps output* pada layer convolution ke-1

3. *Batch Normalization layers*

   Feature maps yang diperoleh dari *convolution layer*, selanjutnya dinormalisasi menggunakan Persamaan. Terlebih dahulu dilakukan inisialisasi parameter-parameter *Batch Normalization layers*:

Nama parameter	Nilai
(μ)	-0.1563
(σ^2)	0.3316
ϵ	0.001

Selanjutnya nilai β dan γ diinisialisasi sebagai berikut:

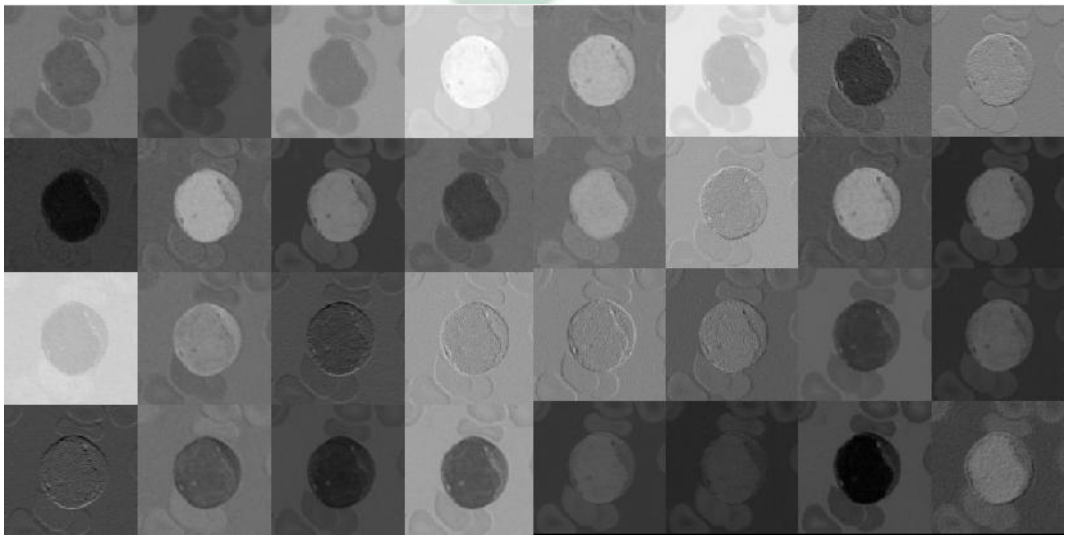
$$\beta = \begin{bmatrix} 0.3064 & -1.4081 & 1.2066 & \cdots & -0.6390 \end{bmatrix}$$

$$\gamma = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

Nilai dari *Dimensions of Scale* (γ) dan *offset* (β) bervariasi untuk setiap *feature maps*. Berikut merupakan perhitungan normalisasi untuk *feature maps* $I^{(1)}$.

$$
\begin{aligned}
\mathrm{F}_{x,y}^{(n)(\ell)} &= \gamma^{(n)} \frac{\mathrm{F}_{x,y}^{(n)(\ell-1)} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta^{(n)} \\
\mathrm{F}_{1,1}^{(1)(3)} &= (1) \frac{(-9.6213) - (-0.1563)}{\sqrt{(0.3316) + (0.001)}} + (0.3064) \\
&= -16.1055 \\
\mathrm{F}_{2,1}^{(1)(3)} &= (1) \frac{(-4.3068) - (-0.1563)}{\sqrt{(0.3316) + (0.001)}} + (0.3064) \\
&= -6.8904 \\
&\vdots \\
\mathrm{F}_{149,149}^{(1)(3)} &= (1) \frac{(36.9831) - (-0.1563)}{\sqrt{(0.3316) + (0.001)}} + (0.3064) \\
&= 64.7046
\end{aligned}
$$

Seluruh *feature maps* hasil normalisasi, ditampilkan dalam bentuk image seperti yang terdapat pada Gambar 4.5.



Gambar 4.5 Tampilan *feature maps* hasil normalisasi

4. ReLU layers

Dalam ReLU layers, nilai-nilai *feature* akan diubah sesuai Persamaan fungsi aktifasinya. Dalam Persamaan 2.62, seluruh *feature* yang bernilai negatif diubah menjadi 0. Berikut diperlihatkan aktifasi ReLu pada layer ke-4 arsitektur inception V3. Terlebih dahulu ditampilkan salahsatu *feature maps* yang dihasilkan layer sebelumnya:

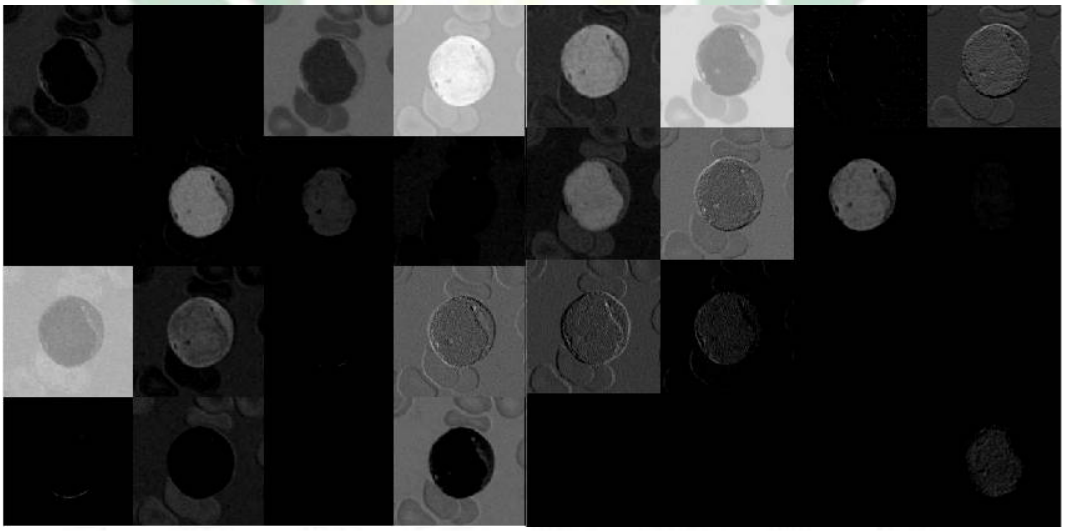
$$\mathrm{F}^{(1)(3)}_{97:102,97:102} = \begin{bmatrix} -0.2092 & -0.2529 & -0.2574 & -0.1949 & -0.0849 & 0.3425 \\ -0.2215 & -0.1964 & -0.2016 & -0.1575 & -0.0506 & 0.4129 \\ -0.1412 & -0.1030 & -0.0651 & -0.0183 & 0.0702 & 0.4613 \\ 0.0161 & 0.0006 & -0.1283 & -0.1148 & 0.1871 & 0.6072 \\ -0.0013 & -0.0427 & -0.0848 & -0.0062 & 0.3624 & 0.7637 \\ -0.0149 & -0.0989 & -0.1376 & -0.0186 & 0.3366 & 0.7811 \end{bmatrix}$$

Feature maps yang dihasilkan dari layer ke-3 diatas, selanjutnya diproses dalam *ReLu layer* menggunakan Persamaan 2.62.

$$\mathrm{F}^{(4)}_{x,y} = \max\left(0, \mathrm{F}^{(3)}_{x,y}\right)$$

$$F^{(1)(4)}_{97:102,97:102} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0.3425 \\ 0 & 0 & 0 & 0 & 0 & 0.4129 \\ 0 & 0 & 0 & 0 & 0.0702 & 0.4613 \\ 0.0161 & 0.0006 & 0 & 0 & 0.1871 & 0.6072 \\ 0 & 0 & 0 & 0 & 0.3624 & 0.7637 \\ 0 & 0 & 0 & 0 & 0.3366 & 0.7811 \end{bmatrix}$$

Seluruh *feature maps* yang dihasilkan pada layer ini, ditampilkan pada Gambar 4.6.



Gambar 4.6 Tampilan *feature maps* hasil aktifasi ReLu

5. *Pooling Layers*

Pada *layers* ini, ukuran *feature maps* direduksi menggunakan teknik *pooling*. Terdapat dua metode *pooling* yang digunakan, yakni *Maximum pooling* dan *Average pooling*. Berikut perhitungan pada layer ke-5 menggunakkan metode *Maximum pooling* dengan ukuran *cluster* 3×3. Hasil *Maximum pooling*

seperti yang ditampilkan pada Gambar 4.7.

$$
\begin{aligned}
&\mathrm{F}_{1,1}^{(1)(5)} \\
&\qquad = \mathrm{Max} \begin{bmatrix} 0.6122 & 0.5529 & 0.5375 \\ 0.5150 & 0.3171 & 0.3073 \\ 0.6495 & 0.3643 & 0.1968 \end{bmatrix} \\
&\qquad = 0.6495
\end{aligned}
$$

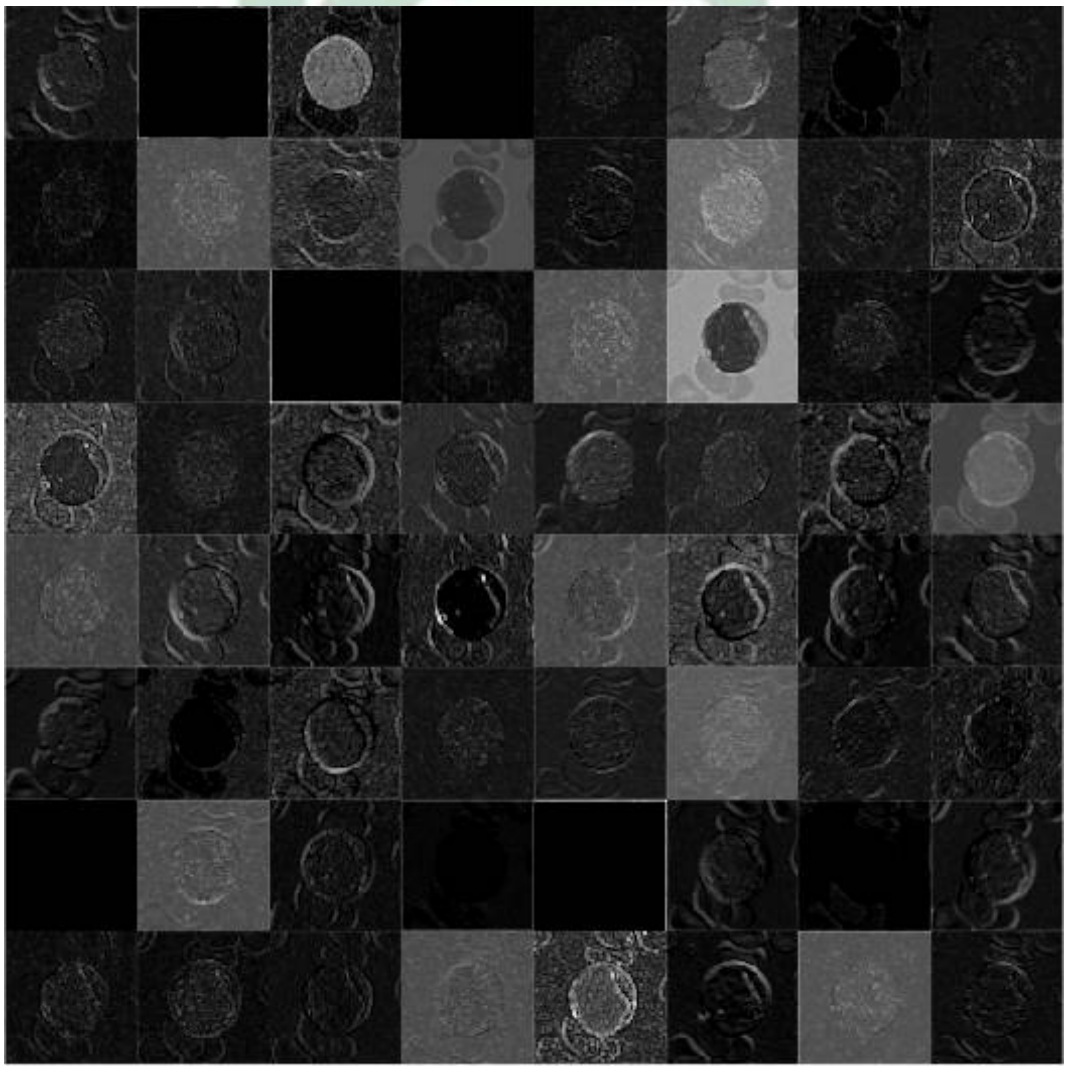
$$
\begin{aligned}
&\mathrm{F}_{2,1}^{(1)(5)} \\
&\qquad = \mathrm{Max} \begin{bmatrix} 0.6495 & 0.3643 & 0.1968 \\ 0.5501 & 0.3571 & 0.2720 \\ 0.4379 & 0.2531 & 0.3854 \end{bmatrix} \\
&\qquad = 0.6495
\end{aligned}
$$

$$\vdots$$

$$F_{73,73}^{(1)(5)}$$

$$= \text{Max} \begin{bmatrix} 0.4701 & 0.3507 & 0.4897 \\ 0.3709 & 0.3725 & 0.4999 \\ 0.4064 & 0.4554 & 0.4924 \end{bmatrix}$$

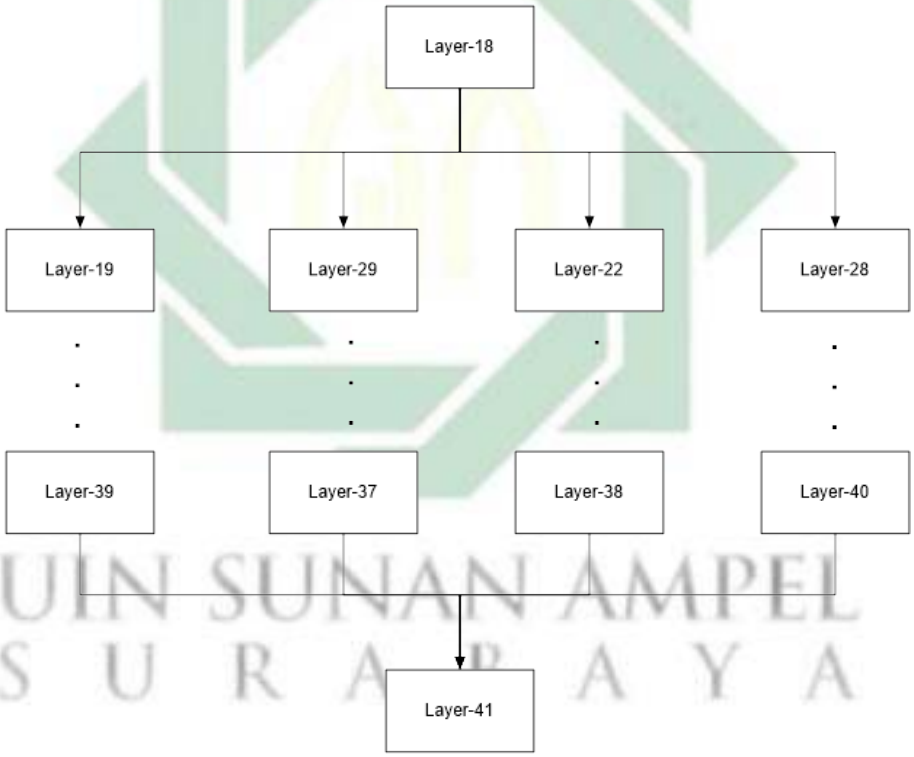
$$= 0.4999$$



Gambar 4.7 Tampilan *feature maps* hasil Pooling

6. *Concatenation layers*

Perbedaan utama arsitektur *Inception* dengan arsitektur CNN lainnya adalah terdapat penggunaan secara terpisah *layer-layer* yang berbeda, pada tingkatan kedalaman yang sama. *Deepht concatenation layers* digunakan untuk menyatukan *feature maps* yang dihasilkan tiap layers tersebut. Penerapan *Deepht concatenation layers* pada arsitektur InceptionV3, seperti yang dapat dilihat pada Gambar 4.8.



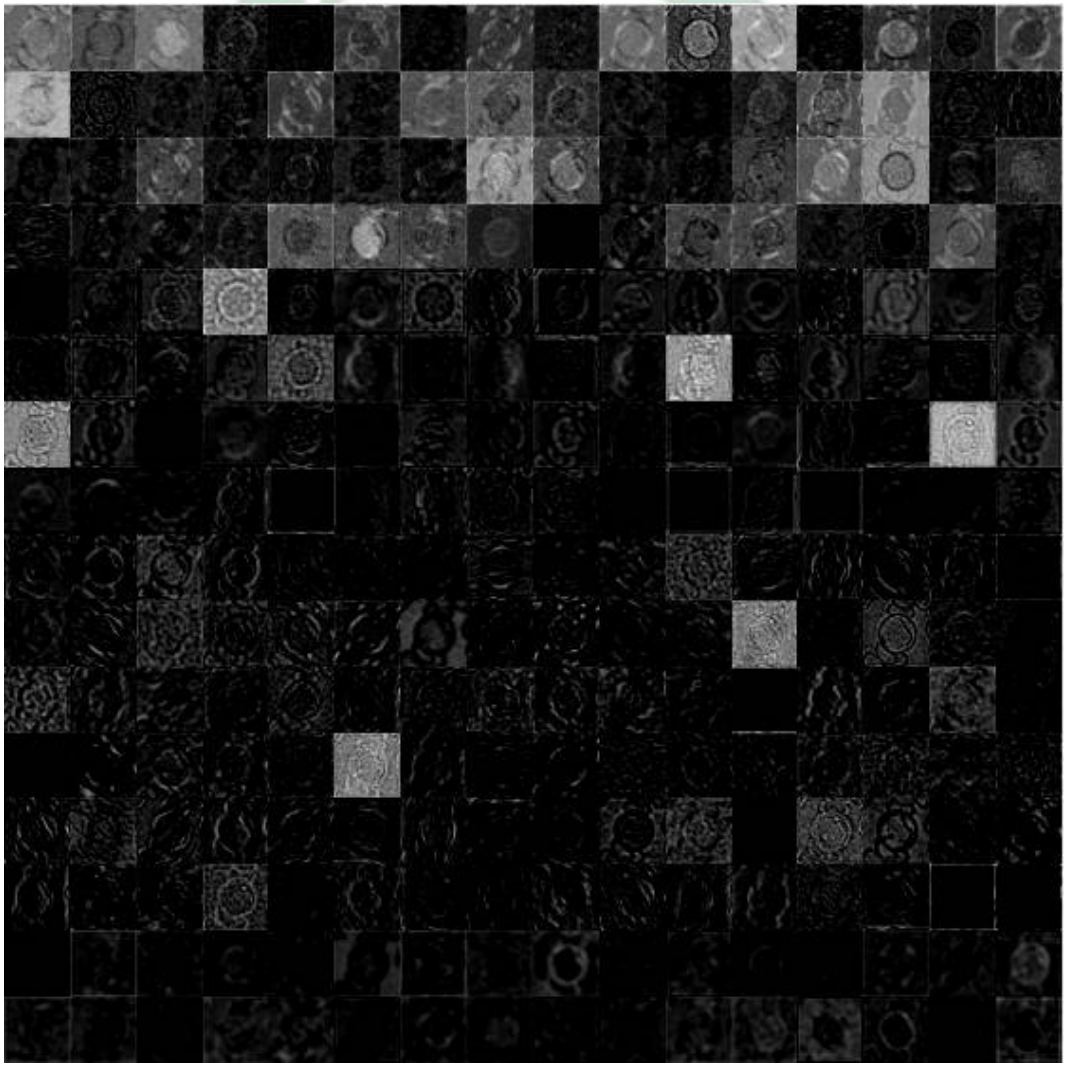
Gambar 4.8 *Layers* pada tingkat kedalaman yang sama

Feature maps yang dihasilkan pada layer-18, diproses peda 4 layers secara terpisah dengan rincian dalam tabel 4.1. *Feature maps* yang dihasilkan dari empat layer tersebut, kemudian diproses lagi pada layar-layer selanjutnya hingga disatukan dalam layer-41 (*concatenation layers*). Proses *concatena-*

tion pada layer-41 menghasilkan *feature maps* dengan ukuran 35 x 35 x 256. Hasil tersebut seperti pada Gambar 4.9.

Tabel 4.1 Ukuran kernel dalam convolution layers

Nama Layer	Ukuran *Feature Maps Output*
Layer-37	35 x 35 x 64
Layer-38	35 x 35 x 64
Layer-39	35 x 35 x 96
Layer-40	35 x 35 x 32



Gambar 4.9 *feature maps* hasil *concatenation*

7. *Fully Connected Layer*

Proses pembelajaran *features* pada *Fully Connected Layer* hanya dapat dilakukan pada 1D array (array satu dimensi). Oleh sebab itu, pada umumnya *feature maps* pada layer sebelumnya dikenakan operasi *flatten*. Proses flaten dilakukan untuk mengubah *feature maps* berbentuk 2D (dua dimensi) menjadi 1D array. Pada arsitektur inceptionV3, *Global average pooling* digunakan sebagai pengganti *flatten layer*. *Features* hasil proses *Global average pooling* berbentuk 1D array, dengan tiap elemennya merupakan *mean* dari setiap *feature maps* pada layer sebelumnya.

Dalam arsitektur Inception-v3, *Global average pooling* digunakan pada layer ke-312. Untuk melakukan perhitungan pada layer ini, terlebih dulu diberikan *Feature maps* yang dihasilkan layer-311, yaitu:

.

$$F^{(1)(311)} = \begin{bmatrix} 0.9350 & 0.4032 & 0 & 0 & 0.3421 & 0 & 0.0972 & 0 \\ 0.3335 & 0 & 0 & 0.6978 & 1.3413 & 1.3303 & 0 & 0 \\ 0 & 0 & 0.0541 & 0.4292 & 0.3050 & 0.0834 & 0.4134 & 0 \\ 0.1926 & 0 & 0 & 0 & 0.1196 & 0.4635 & 0.5536 & 0.2656 \\ 0 & 0 & 0 & 0 & 0.5921 & 1.3326 & 1.2269 & 0.6204 \\ 0 & 0 & 0.3072 & 0.8502 & 1.1824 & 0.1245 & 0.6538 & 0.8678 \\ 0 & 0 & 0 & 0.3811 & 0 & 0 & 0.1298 & 0 \\ 0 & 0.1140 & 0.9417 & 0.2942 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$F^{(2)(311)} = \begin{bmatrix} 0 & 1.0673 & 0.4235 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5088 & 0.7579 & 0.0235 & 0 & 0 & 0 & 0 \\ 0.6158 & 0.3703 & 0.1714 & 0 & 0 & 0 & 0 & 0 \\ 0.7975 & 0 & 0.4334 & 0.2264 & 0 & 0 & 0 & 0 \\ 0.2317 & 0.0911 & 0.1477 & 0 & 0 & 0 & 0.9735 & 0 \\ 0.1961 & 0.1393 & 0 & 0 & 0 & 0.4678 & 0.6561 & 0.1110 \\ 0 & 0 & 0 & 0 & 0.0378 & 1.1250 & 0.8600 & 0.1525 \\ 0 & 0 & 0 & 0 & 0 & 0.7447 & 0 & 0 \end{bmatrix}$$

$$\vdots$$

$$F^{(2048)(311)} = \begin{bmatrix} 0.5738 & 1.0654 & 1.1951 & 1.1314 & 0.8486 & 0.7819 & 0.7566 & 0.5479 \\ 1.0607 & 1.6743 & 1.6540 & 1.5028 & 1.2450 & 1.2371 & 1.1818 & 0.8260 \\ 1.2965 & 1.8481 & 1.5228 & 1.1598 & 0.7436 & 0.8216 & 1.0020 & 0.8666 \\ 0.9357 & 1.2684 & 0.9005 & 0.5631 & 0.1503 & 0.2952 & 0.7649 & 0.8450 \\ 0.7410 & 1.0658 & 0.7046 & 0.2054 & 0 & 0.0525 & 0.8198 & 1.1022 \\ 0.8030 & 1.1298 & 0.9034 & 0.3232 & 0 & 0.5776 & 1.1268 & 1.1993 \\ 1.0015 & 1.4091 & 1.5235 & 1.1289 & 1.0882 & 1.5874 & 1.6844 & 1.3175 \\ 0.7093 & 1.0028 & 1.2605 & 1.1152 & 1.2439 & 1.3749 & 1.2043 & 0.7821 \end{bmatrix}$$

Feature maps yang dihasilkan layer-311 dan berdimensi $8 \times 8 \times 2048$ diatas, selanjutnya diproses menggunakan metode *Global average pooling*.

$$
\begin{aligned}
\mathrm{F}^{(n)(312)} &= \frac{\sum_{i=1}^{\mathrm{M}_1}\sum_{j=1}^{\mathrm{M}_2}\mathrm{F}_{i,j}^{(n)(311)}}{\mathrm{M}_1 \times \mathrm{M}_2} \\
\mathrm{F}^{(1)(312)} &= \frac{\sum_{i=1}^{8}\sum_{j=1}^{8}\mathrm{F}_{i,j}^{(1)(311)}}{8 \times 8} \\
\mathrm{F}^{(1)(312)} &= \frac{\mathrm{F}_{1,1}^{(1)(311)} + \mathrm{F}_{2,1}^{(1)(311)} + \cdots + \mathrm{F}_{8,8}^{(1)(311)}}{16} \\
&= \frac{0.9350 + 0.3335 + ... + 0}{16} \\
&= 0.2809 \\
\mathrm{F}^{(2)(312)} &= \frac{\sum_{i=1}^{8}\sum_{j=1}^{8}\mathrm{F}_{i,j}^{(2)(311)}}{8 \times 8} \\
\mathrm{F}^{(2)(312)} &= \frac{\mathrm{F}_{1,1}^{(2)(311)} + \mathrm{F}_{2,1}^{(2)(311)} + \cdots + \mathrm{F}_{8,8}^{(2)(311)}}{16} \\
&= \frac{0 + 0 + ... + 0}{16} \\
&= 0.1770 \\
&\vdots \\
\mathrm{F}^{(2048)(312)} &= \frac{\sum_{i=1}^{8}\sum_{j=1}^{8}\mathrm{F}_{i,j}^{(2048)(311)}}{8 \times 8} \\
\mathrm{F}^{(2048)(312)} &= \frac{\mathrm{F}_{1,1}^{(2048)(311)} + \mathrm{F}_{2,1}^{(2048)(311)} + \cdots + \mathrm{F}_{8,8}^{(2048)(311)}}{16} \\
&= \frac{0.5738 + 1.0607 + ... + 0.7821}{16} \\
&= 0.9758
\end{aligned}
$$

Langkah selanjutnya adalah perhitungan *fully conected layer*. Terlebih dahu-

lu dilakukan inisialisasi bobot dan bias. Ukuran matrix bobot dan bias menyesuaikan ukuran banyaknya *features* yang di-input, yakni sebanyak 2048 dan jumlah *class* data, yakni sebanyak 3: kelas-1 (ALL), kelas-2 (AML) dan kelas-3 (normal).

$$w = \begin{bmatrix} -0.0086 & -0.0398 & \cdots & -0.0181 \\ 0.0284 & -0.0138 & \cdots & -0.0194 \\ 0.0229 & 0.0439 & \cdots & 0.0516 \end{bmatrix}$$

$$b = \begin{bmatrix} -0.2239 \\ -0.4474 \\ -0.0314 \end{bmatrix}$$

$$\mathrm{F}^{(n)(313)} = \sum_{i=1}^{M} \mathrm{F}_i^{(312)} w_i^{(n)} + b^{(n)}$$

$$\begin{aligned} \mathrm{F}^{(1)(313)} &= (-0.0086 \times 0.2809 + -0.0398 \times 0.1770 + \cdots + -0.0181 \times 0.9758) + -0.2239 \\ &= -0.6821 \end{aligned}$$

$$\begin{aligned} \mathrm{F}^{(2)(313)} &= (0.0284 \times 0.2809 + -0.0138 \times 0.1770 + \cdots + -0.0194 \times 0.9758) + -0.4474 \\ &= 1.4318 \end{aligned}$$

$$\begin{aligned} \mathrm{F}^{(3)(313)} &= (0.0229 \times 0.2809 + 0.0439 \times 0.1770 + \cdots + 0.0516 \times 0.9758) + -0.0314 \\ &= 0.5397 \end{aligned}$$

Selanjutnya, *image* yang di-input, diklasifikasi pada kelas (n) berdasarkan ni-

lai tertinggi fungsi probabilitas *softmax*. Nilai probabilitas ini dihitung menggunakan Persamaan.

$$
\begin{aligned}
softmax^{(n)} &= \frac{e^{F^{(n)}}}{\sum_{i=1}^{M} e^{F^{(i)}}} \\
\mathrm{F}^{(n)(314)} &= \frac{e^{F^{(n)(314)}}}{\sum_{i=1}^{M} e^{F^{(i)}}} \\
\mathrm{F}^{(1)(314)} &= \frac{e^{-0.6821}}{e^{-0.6821} + e^{1.4318} + e^{0.5397}} \\
&= 0.0789 \\
\mathrm{F}^{(2)(314)} &= \frac{e^{1.4318}}{e^{-0.6821} + e^{1.4318} + e^{0.5397}} \\
&= 0.6534 \\
\mathrm{F}^{(3)(314)} &= \frac{e^{0.5397}}{e^{-0.6821} + e^{1.4318} + e^{0.5397}} \\
&= 0.2677
\end{aligned}
$$

Berdasarkan nilai probabilitas softmax, *image* diklasifikasi kedalam kelas-2 (AML).

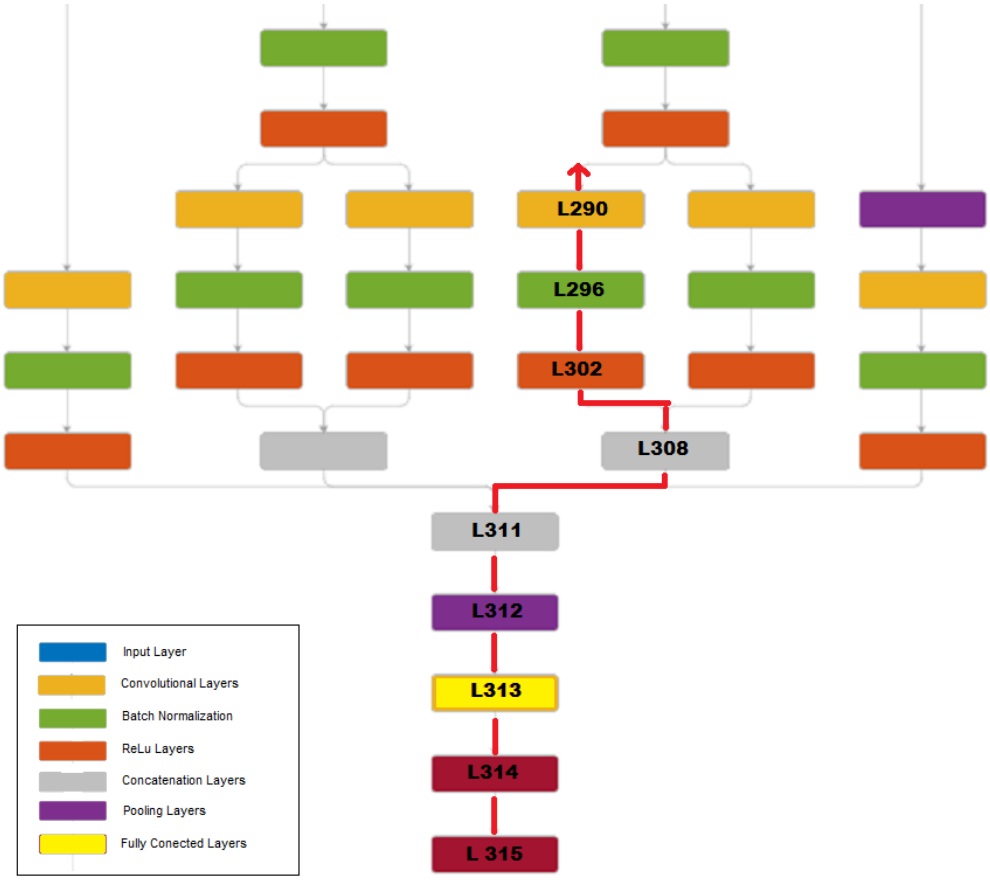
Perhitungan nilai loss dilakukan dengan Persamaan 2.66. Terlebihdahulu diberikan nilai target:

$$
T = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}
$$

$$
\begin{aligned}
\mathrm{E} &= -\sum_{n=1}^{\mathrm{N}} t_n ln(p_n) \\
&= -(T_1 ln(p_1) + T_2 ln(p_2) + T_3 ln(p_3)) \\
&= -(0 \times ln(0.0789) + 1 \times ln(0.6534) + 0 \times ln(0.2677)) \\
&= -(0 + -0.4256 + 0) \\
&= 0.4256
\end{aligned}
$$

8. *Backward pass*

   Proses *backward pass* dilakukan untuk memperbarui nila-nilai *learnable parameter* dalam sistem. Terdapat tiga layers yang berisikan *learnable parameter*, yakni: *convolution layers*, *batch normalization layers* dan *fully conected layers*. Gambar 4.10 menampilkan ilustari jalur proses perhitungan *backward pass*, untuk memperbarui *learnable parameter* dalam layer ke-313, layer-296 dan layer-290 pada arsitektur inception-v3.



Gambar 4.10 Jalur perhitungan *bacward pass* Inception-v3

a. Perhitungan matrix *gradient loss* $\delta^{(313)}$

Proses *backward pass* dimulai dengan menghitung nilai $\delta^{(313)}$. Perhitungan dilakukan dengan menggunakan Persamaan 2.67.

$$\delta_i^{(313)} = p_i - t_i$$

$$\delta_1^{(313)} = p_1 - t_1$$
$$= 0.0789 - 0 = 0.0789$$
$$\delta_2^{(313)} = p_2 - t_2$$
$$= 0.6534 - 1 = -0.3466$$
$$\delta_3^{(313)} = p_3 - t_3$$
$$= 0.2677 - 0 = 0.2677$$

b. Pembaruan bobot $\mathbf{W}^{(313)}$ dan bias $b^{(313)}$

Sebelum perhitungan dilakukan, diberikan bobot dan bias lama, serta *feature maps* yang dihasil layer sebelumnya ($\mathbf{F}^{(312)}$). Seluruh matrix yang diberikan ditampilkan dalam Gambar 4.11.

$\mathbf{F}^{(312)}$

0.2809	0.177	0.093	0.0086	0.0507	0.1324	1.3002	...	0.2148
n = 1	n = 2	n = 3	n = 4	n = 5	n = 6	n = 7		n = 384

$\mathbf{W}^{(313)}$ $b^{(313)}$

-0.0086	-0.0398	-0.0185	0.0125	-0.0013	...	-0.0084	-0.2239
0.0284	-0.0138	0.0174	0.0177	-0.0178	...	-0.009	-0.4474
0.0229	0.0439	0.0014	-0.0357	-0.0251	...	0.0194	-0.0314
n = 1	n = 2	n = 3	n = 4	n = 5		n = 384	

Gambar 4.11 Matrix bobot dan bias serta *feature maps input*

Pembaruan bobot dilakukan dengan menggunakan Persamaan 2.78.

$$\mathbf{W}_i^{(n)(313)} = \mathbf{W}_i^{(n)(313)} - \alpha \mathrm{F}_n^{(312)} \delta_i^{(313)}$$

$$
\begin{aligned}
\mathbf{W}_1^{(1)(313)} &= \mathbf{W}_1^{(1)(313)} - \alpha \mathbf{F}_1^{(312)} \delta_1^{(313)} \\
&= -0.0086 - 0.1 \times 0.2809 \times 0.0789 \\
&= -0.0108 \\
\mathbf{W}_2^{(1)(313)} &= \mathbf{W}_{2,1} - \alpha \mathbf{F}_2^{(312)} \delta_1^{(313)} \\
&= 0.0398 - 0.1 \times 0.1770 \times 0.0789 \\
&= -0.0412 \\
&\vdots \\
\mathbf{W}_3^{(384)(313)} &= \mathbf{W}_{2048,3} - \alpha \mathbf{F}_{2048}^{(312)} \delta_3^{(313)} \\
&= 0.0516 - 0.1 \times 0.9758 \times 0.2677 \\
&= 0.0255
\end{aligned}
$$

Pembaruan bias dilakukan dengan menggunakan Persamaan 2.81.

$$
\begin{aligned}
b_j^{(313)} &= b_j - \alpha\delta_j^{(313)} \\
b_1^{(313)} &= b_1 - \alpha\delta_1^{(313)} \\
&= -0.2239 - 0.1 \times 0.0789 \\
&= -0.2318 \\
b_2^{(313)} &= b_2 - \alpha\delta_2^{(313)} \\
&= -0.4474 - 0.1 \times -0.3466 \\
&= -0.4127 \\
b_3^{(313)} &= b_3 - \alpha\delta_3^{(313)} \\
&= -0.0314 - 0.1 \times 0.0789 \\
&= -0.0393
\end{aligned}
$$

c. Perhitungan matrix $\delta^{(312)}$

Gradient loss pada layer ke-312 dapat dihitung dengan menggunakan Persamaan 2.85

$$\delta^{(n)(\ell-1)} = \sum_i \delta^{(n)(\ell)} \mathbf{w}_i^{(n)(\ell)}$$

$$\delta^{(1)(312)} = \sum_i \delta^{(1)(313)} \mathbf{W}_i^{(1)(313)}$$
$$= 0.0789 \times -0.0086 + -0.3466 \times 0.0284 + 0.2677 \times 0.0229$$
$$= -0.0044$$
$$\delta^{(2)(312)} = \sum_i \delta^{(2)(313)} \mathbf{W}_i^{(2)(313)}$$
$$= 0.0789 \times -0.0398 + -0.3466 \times -0.0138 + 0.2677 \times 0.0439$$
$$= 0.0134$$
$$\vdots$$
$$\delta^{(384)(312)} = \sum_i \delta^{(384)(313)} \mathbf{W}_i^{(384)(313)}$$
$$= 0.0789 \times -0.0084 + -0.3466 \times -0.0090 + 0.2677 \times 0.0194$$
$$= 0.0076$$

d. Perhitungan matrix $\delta^{(302)}$

Selanjutnya matrix $\delta^{(312)}$ ditransformasikan dengan Persamaan 2.56 dan dikalikan dengan $\mathbf{F'}^{(312)}$. Sehinggga menghasilkan matrix $\delta^{(302)}$ pada Gambar 4.12.

$$\delta^{(\ell)} = reshape(\delta^{(\ell+1)})\mathbf{F'}^{(\ell+1)}$$
$$\delta^{(302)} = reshape(\delta^{(312)})\frac{1}{M_1 M_2}$$
$$= reshape(\delta^{(312)})\frac{1}{16}$$

$\delta^{(302)} \times 10^{-3}$

-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728
-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728	-0.2728

n = 1

0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376

n = 2

⋮

0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779

n = 384

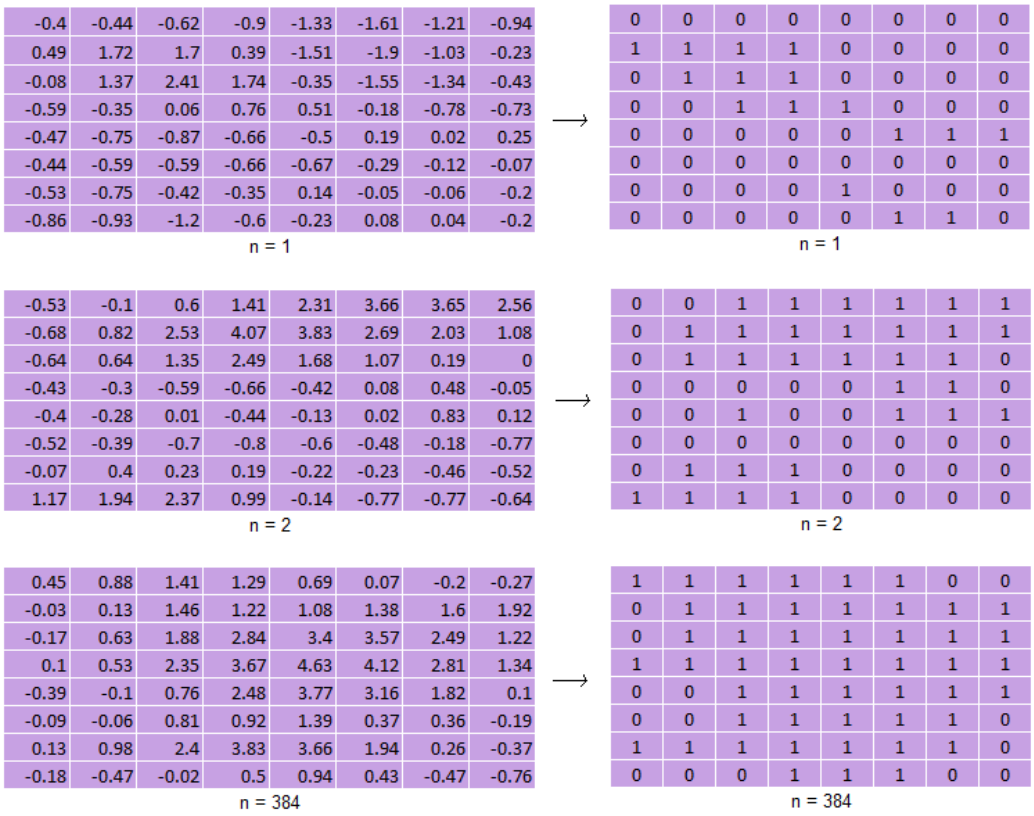
Gambar 4.12 Matrix *gradient loss* layer ke-302

e. Perhitungan matrix $\delta^{(296)}$

Untuk mendapatkan matrix $\delta^{(296)}$, sebelumnya dilakukan perhitungan matrix $\mathbf{F'}^{(302)}$ perhitungan dilakukan dengan menngunakan Persamaan 2.63.

$$\mathbf{F'}^{(n)(302)}_{x,y} = \begin{cases} 1, & \text{jika } \mathbf{F}^{(n)(296)}_{x,y} \geq 0 \\ 0, & \text{lainnya} \end{cases}$$

Hasil perhitungan matrix $\mathbf{F'}^{(302)}$ ditampilkan dalam Gambar 4.13.



Gambar 4.13 Matrix perhitungan turunan pertama fungsi ReLu layers ke-302

Selanjutnya dilakukan perhitungan matrix $\delta^{(296)}$:

$$\delta^{(296)} = \delta^{(302)}\mathbf{F'}^{(302)}$$

matrix $\delta^{(296)}$ atau matrix $\frac{\partial \mathrm{E}}{\partial \widehat{x}}$ yang dihasilkan ditunjukkan dalam Gambar 4.14.

$\delta^{(296)} \times 10^{-3}$

0	0	0	0	0	0	0	0
-0.2728	-0.2728	-0.2728	-0.2728	0	0	0	0
0	-0.2728	-0.2728	-0.2728	0	0	0	0
0	0	-0.2728	-0.2728	-0.2728	0	0	0
0	0	0	0	0	-0.2728	-0.2728	-0.2728
0	0	0	0	0	0	0	0
0	0	0	0	-0.2728	0	0	0
0	0	0	0	0	-0.2728	-0.2728	0

n = 1

0	0	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376
0	0.8376	0.8376	0.8376	0.8376	0.8376	0.8376	0
0	0	0	0	0	0.8376	0.8376	0
0	0	0.8376	0	0	0.8376	0.8376	0.8376
0	0	0	0	0	0	0	0
0	0.8376	0.8376	0.8376	0	0	0	0
0.8376	0.8376	0.8376	0.8376	0	0	0	0

n = 2

⋮

0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0	0
0	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0	0	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779
0	0	0.4779	0.4779	0.4779	0.4779	0.4779	0
0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0.4779	0
0	0	0	0.4779	0.4779	0.4779	0	0

n = 384

Gambar 4.14 Matrix *gradient loss* pada layer ke-296

f. Pembaruan bobot *scale* (γ) dan *shift* (β)

Sebelum perhitungan dilakukan, diberikan *scale* dan *shift* lama, serta Matrix $\widehat{x}$ pada layer ke-296. Seluruh matrix yang diberikan ditampilkan dalam Gambar 4.15.

$\widehat{x}$

-0.1405	-0.1409	-0.1426	-0.1452	-0.1493	-0.152	-0.1482	-0.1456
-0.1321	-0.1205	-0.1206	-0.133	-0.151	-0.1547	-0.1464	-0.1389
-0.1375	-0.1238	-0.114	-0.1203	-0.14	-0.1514	-0.1494	-0.1408
-0.1424	-0.1401	-0.1362	-0.1296	-0.132	-0.1384	-0.1441	-0.1437
-0.1412	-0.1439	-0.145	-0.143	-0.1415	-0.135	-0.1365	-0.1343
-0.1409	-0.1423	-0.1424	-0.143	-0.1431	-0.1395	-0.1379	-0.1374
-0.1417	-0.1438	-0.1407	-0.1401	-0.1354	-0.1372	-0.1373	-0.1386
-0.1449	-0.1455	-0.1481	-0.1424	-0.139	-0.136	-0.1363	-0.1386

n = 1

-0.1531	-0.1489	-0.142	-0.134	-0.1252	-0.1119	-0.112	-0.1228
-0.1546	-0.1398	-0.123	-0.1078	-0.1102	-0.1215	-0.1279	-0.1372
-0.1542	-0.1416	-0.1346	-0.1234	-0.1314	-0.1373	-0.146	-0.1479
-0.1521	-0.1508	-0.1537	-0.1543	-0.152	-0.1471	-0.1432	-0.1484
-0.1518	-0.1507	-0.1477	-0.1522	-0.1492	-0.1477	-0.1397	-0.1467
-0.153	-0.1518	-0.1547	-0.1558	-0.1538	-0.1526	-0.1497	-0.1554
-0.1486	-0.1439	-0.1456	-0.146	-0.1501	-0.1501	-0.1524	-0.153
-0.1363	-0.1288	-0.1246	-0.1382	-0.1493	-0.1554	-0.1554	-0.1541

n = 2

⋮

-0.1112	-0.1072	-0.1023	-0.1034	-0.109	-0.1147	-0.1172	-0.1178
-0.1156	-0.1142	-0.1019	-0.1041	-0.1054	-0.1026	-0.1006	-0.0976
-0.1169	-0.1095	-0.098	-0.0891	-0.0839	-0.0824	-0.0924	-0.1041
-0.1144	-0.1104	-0.0936	-0.0815	-0.0726	-0.0773	-0.0894	-0.1029
-0.1189	-0.1162	-0.1083	-0.0925	-0.0805	-0.0862	-0.0986	-0.1144
-0.1162	-0.1159	-0.1078	-0.1068	-0.1025	-0.112	-0.1121	-0.1171
-0.1141	-0.1063	-0.0932	-0.08	-0.0816	-0.0974	-0.1129	-0.1188
-0.117	-0.1196	-0.1156	-0.1107	-0.1067	-0.1113	-0.1196	-0.1224

n = 384

γ

1	1	1	1	1	1	...	1
n = 1	n = 2	n = 3	n = 4	n = 5	n = 6		n =384

β

-0.138	-0.1493	-0.0957	-0.1602	-0.1279	-0.0529	...	-0.1164
n = 1	n = 2	n = 3	n = 4	n = 5	n = 6		n =384

Gambar 4.15 Matrix *scale* dan *shift* dan *feature maps* hasil normalisasi

Pembaruan *scale* dilakukan dengan menggunakan Persamaan 2.43.

$$\gamma^{(n)(296)} = \gamma^{(n)(296)} - \alpha \sum_{x=1}^{8} \sum_{y=1}^{8} \delta_{x,y}^{(n)(296)} \widehat{x}_{x,y}$$

$$\gamma^{(1)(296)} = \gamma^{(1)(296)} - \alpha \sum_{x=1}^{8}\sum_{y=1}^{8} \delta_{x,y}^{(1)(296)} \widehat{x}_{x,y}$$

$$= 1.0000 - 0.1((-0.00 \times 10^{-3}) \times -0.14 +$$

$$(-0.27 \times 10^{-3}) \times -0.13 + \cdots + (-0.00 \times 10^{-3}) \times -0.14)$$

$$= 1.0000 - 0.1(0.0006)$$

$$= 0.9999$$

$$\gamma^{(2)(296)} = \gamma^{(2)(296)} - \alpha \sum_{x=1}^{8}\sum_{y=1}^{8} \delta_{x,y}^{(2)(296)} \widehat{x}_{x,y}$$

$$= 1.0000 - 0.1((0.00 \times 10^{-3}) \times -0.15 +$$

$$(0.00 \times 10^{-3}) \times -0.15 + \cdots + (0.00 \times 10^{-3}) \times -0.14)$$

$$= 1.0000 - 0.1(-0.0036)$$

$$= 1.0004$$

$$\vdots$$

$$\gamma^{(384)(296)} = \gamma^{(384)(296)} - \alpha \sum_{x=1}^{8}\sum_{y=1}^{8} \delta_{x,y}^{(384)(296)} \widehat{x}_{x,y}$$

$$= 1.0000 - 0.1((0.48 \times 10^{-3}) \times -0.11 +$$

$$(0.00 \times 10^{-3}) \times -0.12 + \cdots + (0.00 \times 10^{-3}) \times -0.12)$$

$$= 1.0000 - 0.1(-0.0023)$$

$$= 1.0002$$

Pembaruan bias dilakukan dengan menggunakan Persamaan 2.46.

$$\beta^{(n)(296)} = \beta^{(n)(296)} - \alpha \sum_{x=1}^{8} \sum_{y=1}^{8} \delta_{x,y}^{(n)(296)}$$

$$\begin{aligned}
\beta^{(1)(296)} &= \beta^{(1)(296)} - \alpha \sum_{x=1}^{8} \sum_{y=1}^{8} \delta_{x,y}^{(1)(296)} \\
&= -0.1380 - 0.1((-0.00 \times 10^{-3}) + (-0.27 \times 10^{-3}) + \cdots + (-0.00 \times 10^{-3})) \\
&= -0.1380 - 0.1(-0.0044) \\
&= -0.1376 \\
\beta^{(2)(296)} &= \beta^{(2)(296)} - \alpha \sum_{x=1}^{8} \sum_{y=1}^{8} \delta_{x,y}^{(2)(296)} \\
&= -0.1493 - 0.1((0.00 \times 10^{-3}) + (0.00 \times 10^{-3}) + \cdots + (0.00 \times 10^{-3})) \\
&= -0.1493 - 0.1(0.0268) \\
&= -0.1520 \\
&\vdots \\
\beta^{(384)(296)} &= \beta^{(384)(296)} - \alpha \sum_{x=1}^{8} \sum_{y=1}^{8} \delta_{x,y}^{(384)(296)} \\
&= -0.1164 - 0.1((0.48 \times 10^{-3}) + (0.00 \times 10^{-3}) + \cdots + (0.00 \times 10^{-3})) \\
&= -0.1164 - 0.1(0.0234) \\
&= -0.1188
\end{aligned}$$

g. Perhitungan matrix $\delta^{(290)}$

Untuk mendapatkan matrix $\delta^{(290)}$, sebelumnya dilakukan perhitungan $\frac{\partial \mathrm{E}}{\partial \sigma^2}$

dan $\frac{\partial E}{\partial \mu}$ dengan menggunakan persamaan 2.48 dan 2.50. Sehingga terlebih dahulu diberikan matrix $\mathbf{F}^{(290)}$ dan beberapa parameter pada layer ke-296, seperti yang ditampilkan dalam Gambar 4.16.

$\mathbf{F}^{(290)}$

-0.0018	-0.0059	-0.0234	-0.0504	-0.0927	-0.1202	-0.0815	-0.0549
0.0839	0.2037	0.2023	0.0746	-0.1105	-0.1482	-0.0631	0.0145
0.0293	0.1699	0.2709	0.2062	0.0029	-0.1144	-0.0937	-0.0048
-0.0212	0.0024	0.042	0.11	0.0859	0.0191	-0.0396	-0.0345
-0.0088	-0.0366	-0.0479	-0.0277	-0.0122	0.0549	0.0386	0.0612
-0.006	-0.0204	-0.0211	-0.0278	-0.0286	0.0085	0.0247	0.0298
-0.0149	-0.0364	-0.0039	0.0022	0.0506	0.0315	0.0308	0.0173
-0.0469	-0.0537	-0.0804	-0.0216	0.0138	0.0441	0.0408	0.0176

n = 1

-0.0192	0.0233	0.0926	0.1735	0.2626	0.3964	0.3957	0.2869
-0.0341	0.1151	0.2841	0.4372	0.413	0.2998	0.2347	0.1407
-0.0302	0.0969	0.1675	0.2803	0.2	0.14	0.0524	0.0331
-0.0096	0.0039	-0.0248	-0.0317	-0.0086	0.0413	0.0808	0.0285
-0.0065	0.0053	0.0348	-0.0104	0.0204	0.0353	0.1158	0.0456
-0.0184	-0.0057	-0.0355	-0.0462	-0.0259	-0.0141	0.0154	-0.0425
0.0262	0.0736	0.0566	0.0524	0.0112	0.0111	-0.012	-0.0178
0.1498	0.2262	0.2683	0.1314	0.0192	-0.0429	-0.0429	-0.0295

n = 2

⋮

0.07	0.1115	0.1622	0.1509	0.093	0.0335	0.0075	0.001
0.0244	0.0389	0.1671	0.1436	0.1307	0.1593	0.1802	0.2112
0.0105	0.0877	0.2076	0.3003	0.354	0.3699	0.2658	0.1441
0.0362	0.0781	0.253	0.3794	0.4716	0.4227	0.2971	0.1559
-0.0103	0.0177	0.0999	0.2649	0.3893	0.3302	0.2014	0.0364
0.0182	0.0213	0.1051	0.1157	0.1603	0.062	0.061	0.0089
0.0395	0.1214	0.2574	0.3947	0.3782	0.213	0.052	-0.0089
0.0095	-0.018	0.0246	0.0748	0.1167	0.0685	-0.0179	-0.0462

n = 384

μ

0.0232	0.0187	0.0185	0.0152	0.0204	0.0179	...	0.0157
n = 1	n = 2	n = 3	n = 4	n = 5	n = 6		n =384

σ^2

0.0085	0.0088	0.0084	0.0086	0.0083	0.0084	...	0.0082
n = 1	n = 2	n = 3	n = 4	n = 5	n = 6		n =384

Gambar 4.16 *Feature maps* layer ke-290, serta nilai mean dan varian layer k3 296

Perhitungan $\frac{\partial \mathrm{E}}{\partial \sigma^2}$:

$$\frac{\partial \mathrm{E}}{\partial \sigma^2}^{(n)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}}^{(n)} \left(\mathbf{F}_{x,y}^{(n)(290)} - \mu\right) \frac{-1}{2} \left(\sigma^2 + \epsilon\right)^{\frac{-3}{2}}$$

$$\frac{\partial \mathrm{E}}{\partial \sigma^2}^{(1)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}}^{(1)} \left(\mathbf{F}_{x,y}^{(1)(290)} - 0.0232\right) \frac{-1}{2} \left(0.0085 + 0.001\right)^{\frac{-3}{2}}$$

$$= 0.2031$$

$$\frac{\partial \mathrm{E}}{\partial \sigma^2}^{(2)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}}^{(2)} \left(\mathbf{F}_{x,y}^{(2)(290)} - 0.0187\right) \frac{-1}{2} \left(0.0088 + 0.001\right)^{\frac{-3}{2}}$$

$$= -2.2278$$

$$\vdots$$

$$\frac{\partial \mathrm{E}}{\partial \sigma^2}^{(384)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}}^{(384)} \left(\mathbf{F}_{x,y}^{(384)(290)} - 0.0157\right) \frac{-1}{2} \left(0.0082 + 0.001\right)^{\frac{-3}{2}}$$

$$= -2.2208$$

Perhitungan $\frac{\partial \mathrm{E}}{\partial \mu}$:

$$\frac{\partial \mathrm{E}}{\partial \mu}^{(n)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}}^{(n)} \frac{-1}{\sqrt{\sigma^2+\epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma^2} \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{(-2)\left(\mathbf{F}^{(n)(290)} - \mu\right)}{M_1 M_2}$$

$$\frac{\partial \mathrm{E}}{\partial \mu}^{(1)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}}^{(1)} \frac{-1}{\sqrt{0.0085+0.001}} + (0.2031) \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{(-2)\left(\mathbf{F}^{(1)(290)} - 0.0232\right)}{16}$$

$$= -28.4745$$

$$\frac{\partial \mathrm{E}}{\partial \mu}^{(2)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}}^{(2)} \frac{-1}{\sqrt{0.0088+0.001}} + (-2.2278) \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{(-2)\left(\mathbf{F}^{(2)(290)} - 0.0187\right)}{16}$$

$$= -28.7897$$

$$\vdots$$

$$\frac{\partial \mathrm{E}}{\partial \mu}^{(384)} = \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{\partial \mathrm{E}}{\partial \widehat{x}}^{(384)} \frac{-1}{\sqrt{0.0084+0.001}} + (-2.2208) \sum_{x=1}^{8}\sum_{y=1}^{8} \frac{(-2)\left(\mathbf{F}^{(384)(290)} - 0.018\right.}{16}$$

$$= -28.5148$$

Selanjutnya perhitungan $\delta_{x,y}^{(290)}$ dapat dilakukan dengan menggunakan Persamaan 2.53.

$$\delta_{x,y}^{(n)(290)} = \frac{\partial \mathrm{E}}{\partial \widehat{x}_{x,y}^{(n)}} \frac{-1}{\sqrt{\sigma_{(n)}^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma_{(n)}^2}(-2)\frac{\left(\mathbf{F}_{x,y}^{(n)(290)} - \mu^{(n)}\right)}{M_1 M_2} + \frac{\partial \mathrm{E}}{\partial \mu^{(n)}}\frac{1}{M_1 M_2}$$

$$\delta_{1,1}^{(1)(290)} = \frac{\partial \mathrm{E}}{\partial \widehat{x}_{1,1}^{(1)}} \frac{-1}{\sqrt{\sigma_{(1)}^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma_{(1)}^2}(-2)\frac{\left(\mathbf{F}_{1,1}^{(1)(290)} - \mu^{(1)}\right)}{M_1 M_2} + \frac{\partial \mathrm{E}}{\partial \mu^{(1)}}\frac{1}{M_1 M_2}$$

$$\delta_{1,1}^{(1)(290)} = (0)\frac{-1}{\sqrt{0.0085 + 0.001}} + (0.2031)(-2)\frac{(-0.0018 - 0.0232)}{16} + (-28.4745)\frac{1}{16}$$

$$= -1.7803$$

$$\delta_{2,1}^{(1)(290)} = \frac{\partial \mathrm{E}}{\partial \widehat{x}_{2,1}^{(1)}} \frac{-1}{\sqrt{\sigma_{(1)}^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma_{(1)}^2}(-2)\frac{\left(\mathbf{F}_{2,1}^{(1)(290)} - \mu^{(1)}\right)}{M_1 M_2} + \frac{\partial \mathrm{E}}{\partial \mu^{(1)}}\frac{1}{M_1 M_2}$$

$$\delta_{2,1}^{(1)(290)} = (-0.27 \times 10^{-3})\frac{-1}{\sqrt{0.0085 + 0.001}} + (0.2031)(-2)\frac{(0.0839 - 0.0232)}{16} + (-28.4745)\frac{1}{16}$$

$$= -1.7809$$

$$\vdots$$

$$\delta_{8,8}^{(384)(290)} = \frac{\partial \mathrm{E}}{\partial \widehat{x}_{8,8}^{(384)}} \frac{-1}{\sqrt{\sigma_{(384)}^2 + \epsilon}} + \frac{\partial \mathrm{E}}{\partial \sigma_{(384)}^2}(-2)\frac{\left(\mathbf{F}_{8,8}^{(384)(290)} - \mu^{(384)}\right)}{M_1 M_2} + \frac{\partial \mathrm{E}}{\partial \mu^{(384)}}\frac{1}{M_1 M_2}$$

$$\delta_{1,1}^{(1)(290)} = (0)\frac{-1}{\sqrt{0.0082 + 0.001}} + (-2.2208)(-2)\frac{(-0.0462 - 0.0157)}{16} + (-28.7630)\frac{1}{16}$$

$$= -1.7805$$

h. Pembaruan bobot $\mathbf{W}^{(290)}$ dan bias $b^{(290)}$

Sebelum perhitungan dilakukan, diberikan bobot dan bias lama yang ditampilkan dalam Gambar 4.17.

$\mathbf{W}^{(290)}$

	m = 1		
n = 1	-0.0170	-0.0109	-0.0143
n = 2	-0.0024	0.0000	-0.0036
n = 3	0.0132	0.0129	0.0130
n = 384	-0.0093	-0.0060	-0.0096

	m = 2		
n = 1	-0.0159	-0.0125	-0.0167
n = 2	0.0156	0.0123	0.0150
n = 3	-0.0066	-0.0038	-0.0058
n = 384	0.0076	0.0042	0.0038

	m = 384		
n = 1	-0.0034	-0.0015	-0.0047
n = 2	0.0046	0.0028	0.0054
n = 3	-0.0024	-0.0016	-0.0034
n = 384	0.0010	0.0033	0.0007

$b^{(290)}$

n = 1	0
n = 2	0
n = 3	0
n = 384	0

Gambar 4.17 Matrix bobot dan bias layer ke-290

Selainitu juga diberikan *feature maps* pada layer sebelumnya ($\mathbf{F}^{(288)}$). *feature maps* ini telah diriotasikan sebesar $180°$ dan diberikan zeros padding dikedua sisi, seperti yang tampak pada Gambar 4.18.

n = 1

0	0	0	0.292	0	0	0	0	0	0
0	0	0.208	0.715	0.058	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

n = 2

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

n = 384

0	0	0	0.263	0.077	0	0	0	0	0
0	0	0	0.044	0	0	0	0	0	0
0	0	0	0	0.069	0	0	0	0	0
0	0	0	0	0	0.087	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Gambar 4.18 *Feature maps* pada layer ke-288

Pembaruan bobot dilakukan dengan menggunakan Persamaan 2.23.

$$\mathbf{W}_{x,y}^{(m)(n)(\ell)} = \mathbf{W}_{x,y}^{(m)(n)(\ell)} - \alpha \sum_a \sum_b \delta_{a,b}^{(n)(\ell)} \mathrm{F}_{x-a+1,y-b+1}^{(m)(\ell-1)}$$

$$\mathbf{W}_{1,1}^{(1)(1)(290)} = \mathbf{W}_{1,1}^{(1)(1)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(1)(290)} \mathrm{F}_{1-a+1,1-b+1}^{(1)(288)}$$

$$= -0.0170 - 0.1(-1.7803 \times 0 + -1.7809 \times 0 + -1.7795 \times 0 +$$

$$\cdots + -1.7798 \times 0)$$

$$= -0.0170 - 0.1(-2.2678)$$

$$= 0.2098$$

$$\mathbf{W}_{1,2}^{(1)(1)(290)} = \mathbf{W}_{1,2}^{(1)(1)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(1)(290)} \mathrm{F}_{1-a+1,2-b+1}^{(1)(288)}$$

$$= -0.0170 - 0.1(-1.7803 \times 0 + -1.7809 \times 0 + -1.7795 \times 0 +$$

$$\cdots + -1.7798 \times 0)$$

$$= -0.0170 - 0.1(-2.2651)$$

$$= 0.2156$$

$$\mathbf{W}_{1,3}^{(1)(1)(290)} = \mathbf{W}_{1,3}^{(1)(1)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(1)(290)} \mathrm{F}_{1-a+1,3-b+1}^{(1)(288)}$$

$$= -0.0170 - 0.1(-1.7803 \times 0 + -1.7809 \times 0.2080 + -1.7795 \times 0 +$$

$$\cdots + -1.7798 \times 0)$$

$$= -0.0170 - 0.1(-2.2654)$$

$$= 0.2122$$

Pembaruan bias dilakukan dengan menggunakan Persamaan 2.26.

$$
\begin{aligned}
b^{(n)(\ell)} &= b^{(n)(\ell)} - \alpha \sum_a \sum_b \delta_{a,b}^{(n)(\ell)} \\
b^{(1)(290)} &= b^{(1)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(1)(290)} \\
&= 0 - 0.1(-1.7803 + -1.7809 + -1.7795 + \cdots + -1.7798) \\
&= 0 - 0.1(-113.9695) \\
&= 11.3970 \\
b^{(2)(290)} &= b^{(2)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(2)(290)} \\
&= 0 - 0.1(-1.7888 + -1.7847 + -1.7858 + \cdots + -1.7859) \\
&= 0 - 0.1(-116.0794) \\
&= 11.6079 \\
&\vdots \\
b^{(384)(290)} &= b^{(384)(290)} - \alpha \sum_a \sum_b \delta_{a,b}^{(384)(290)} \\
&= 0 - 0.1(-1.8078 + -1.8001 + -1.7963 + \cdots + -1.7805) \\
&= 0 - 0.1(-117.0452) \\
&= 11.7045
\end{aligned}
$$

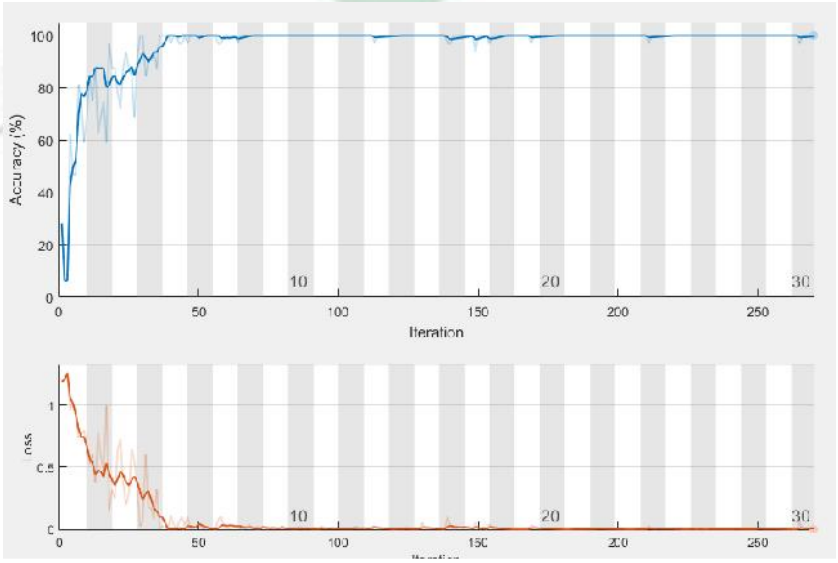
Seluruh proses diatas, terus dilakukan hingga diperoleh *learnable parameter* baru dari seluruh layers.

4.3. Analisis Hasil Klasifikasi Penyakit Leukemia menggunakan Inception V3

Metode InceptionV3 yang digunakan dalam penelitian ini telah berhasil dalam proses klasifikasi citra sel tunggal leukemia. Pada mulanya Dataset ditrain menggunakan metode InceptionV3 dengan hyper parameter yang telah ditetapkan dalam tabel 4.3. Sebelumnya dilakukan pembagian data secara acak dengan *training* sebesar 75% dan 25% sebagai data *testing*. Terlihat pada Gambar 4.19 merupakan grafik proses *training* klasifikasi penyakit Leukemia menggunakan Inception V3.

Tabel 4.2 Hyper parameter proses training

Hyper parameter	Nilai
Max Epochs	30
Mini Batch Size	32
Learning Rate	0.001



Gambar 4.19 Hasil Training klasifikasi penyakit Leukemia menggunakan Inception V3

Grafik diatas menunjukkan bahwa proses *training* berjalan dengan baik. Didapatkan hasil akurasi sebesar 100% dengan 525 iterasi dalam kurun waktu 133 menit 41 detik. Setelah melakukan *training*, sebanyak 25% data testing digunakan pada proses *testing*. Hasil kalsifikasi berupa *confusion matrix* seperti pada Gambar 4.4. Selanjutnya dihitung nilai akurasi menggunakan Persamaan 2.96, nilai sensitifitas menggunakan Persamaan 2.94, dan nilai spesitifitas menggunakan Persamaan 2.95.

Tabel 4.3 Hyper parameter proses training

Hyper parameter	Nilai
Max Epochs	30
Mini Batch Size	32
Learning Rate	0.0001

Tabel 4.4 Confusion Matrix

	AML	ALL	Normal
AML	28	0	1
ALL	0	32	0
Normal	4	0	31

$$
\begin{aligned}
Sensitivity &= \frac{\sum \frac{TP}{TP+FN}}{n} \times 100\% \\
&= \frac{\frac{28}{28+4} + \frac{32}{32+0} + \frac{31}{31+1}}{3} \times 100\% \\
&= \frac{0.8750 + 1 + 0.9688}{3} \times 100\% \\
&= 94.7917\% \\
Specificity &= \frac{\sum \frac{TN}{TN+FP}}{n} \times 100\% \\
&= \frac{\frac{63}{63+1} + \frac{64}{64+0} + \frac{60}{60+4}}{3} \times 100\% \\
&= \frac{0.9844 + 1 + 0.9375}{3} \times 100\% \\
&= 97.3958\% \\
Accuracy &= \frac{TP_{all}}{n_{all}} \times 100\% \\
&= \frac{28 + 32 + 31}{96} \times 100\% \\
&= 94.7917\%
\end{aligned}
$$

Untuk meningkatkan kinerja model, dilakukan beberapa percobaan hyper parameter, yakni minibatch size dan learning rate. Selain itu juga dilakukan perbandingan hasil antara pengujian dengan menggunakan augmentasi dan pengujian tanpa menggunakan augmentasi. Selanjutnya untuk mengoptimalkan proses pembelajaran model, digunakan suatu metode optimasi RMSProb. Nilai maximum epoch yang digunakan adalah 10. Penurunan nilai maximum epoch disesuaikan dengan hasil training sebelumnya, seperti pada grafik, yang menunjukkan bahwa akurasi training mulai stabil dimulai dari epoch ke-6. Hal ini dapat mengoptimalkan model dengan mempersingkat waktu training.

Hasil ujicoba ditampilkan dalam tabel 4.5 - 4.9. Selain itu, untuk mempermudah proses analisis hasil ujicoba metode optimasi, minibatch size dan nilai learning rate, ditampilkan grafik rata-rata tiap uji coba tersebut seperti pada Gambar 4.20 - 4.21.

Tabel 4.5 Hasil evaluasi ujicoba klasifikasi Tanpa menggunakan Augmentasi

Batchsize	Learning rate	Sensitifitas (%)	Spesifisitas (%)	Akurasi (%)
8	0.1	42.71	71.35	42.71
	0.01	98.95	99.48	98.96
	0.001	98.95	99.48	98.96
	0.0001	97.91	98.95	97.91
16	0.1	58.33	79.17	58.33
	0.01	92.71	96.35	92.71
	0.001	98.95	99.47	98.96
	0.0001	95.83	97.91	95.83
32	0.1	55.2	77.6	55.21
	0.01	97.91	98.95	97.92
	0.001	98.95	99.47	98.96
	0.0001	94.79	97.39	94.79
64	0.1	65.62	82.81	65.62
	0.01	97.91	98.95	97.91
	0.001	98.95	99.47	98.96
	0.0001	91.66	95.83	91.66

Tabel 4.6 Hasil evaluasi ujicoba klasifikasi menggunakan Augmentasi

Batchsize	Learning rate	Sensitifitas (%)	Sensitifitas (%)	Akurasi (%)
8	0.1	62.50	81.25	62.50
	0.01	86.46	93.23	86.46
	0.001	98.96	99.48	98.96
	0.0001	98.96	99.48	98.96
16	0.1	45.83	72.92	45.83
	0.01	97.92	98.96	97.92
	0.001	98.96	99.48	98.96
	0.0001	94.79	97.40	94.79
32	0.1	35.42	67.71	35.42
	0.01	97.92	98.96	97.92
	0.001	98.96	99.48	98.96
	0.0001	94.79	97.40	94.79
64	0.1	66.67	83.33	66.67
	0.01	97.92	98.96	97.92
	0.001	97.92	98.96	97.92
	0.0001	87.50	93.75	87.50

Akurasi tertinggi proses klasifikasi dengan menggunakan augmentasi dan tanpa menggunakan augmentasi sama, yakni 98.96%. Sedangkan Rata-rata ujicoba akurasi klasifikasi dengan menggunakan augmentasi dan tanpa menggunakan augmentasi adalah sebesar 85.09% dan 86.58%. Hal ini menunjukkan bahwa tidak terdapat pengaruh yang besar dalam pengggunaan metode augmentasi. Namun jika dilihat dari rata-rata akurasi, proses klasifikasi tanpa menggunakan augmentasi lebih baik dibanding menggunakan augmentasi.

Proses ujicoba dilanjutkan menggunakan optimasi RMSProp. Selama proses ujicoba data yang digunakan merupakan data yang telah dilakukan augmentasi. Meski rata-rata ujicoba akurasi klasifikasi tanpa augmentasi lebih tinggi, namun penggunaan augmentasi mampu memberikan generalisasi yang lebih baik. Hal ini diharapkan mampu meningkatkan performa selama ujicoba optimasi RMSProp.

Tabel 4.7 Hasil evaluasi ujicoba klasifikasi menggunakan metode Train RMSProb decay rate 0.9

Batchsize	Learning rate	Sensitifitas (%)	Spesifisitas (%)	Akurasi (%)
8	0.1	100	48.44	65.63
	0.01	93.75	93.75	93.75
	0.001	100	100	100
	0.0001	90.63	98.44	95.83
16	0.1	66.67	83.33	66.67
	0.01	100	98.44	97.92
	0.001	93.75	100	97.92
	0.0001	68.75	100	89.58
32	0.1	84.37	92.18	84.37
	0.01	100	98.44	98.96
	0.001	96.88	100	98.96
	0.0001	71.88	98.44	88.54
64	0.1	100	48.44	65.63
	0.01	100	98.44	98.96
	0.001	78.13	100	92.71
	0.0001	62.5	92.19	82.29

Tabel 4.8 Hasil evaluasi ujicoba klasifikasi menggunakan metode Train RMSProb decay rate 0.99

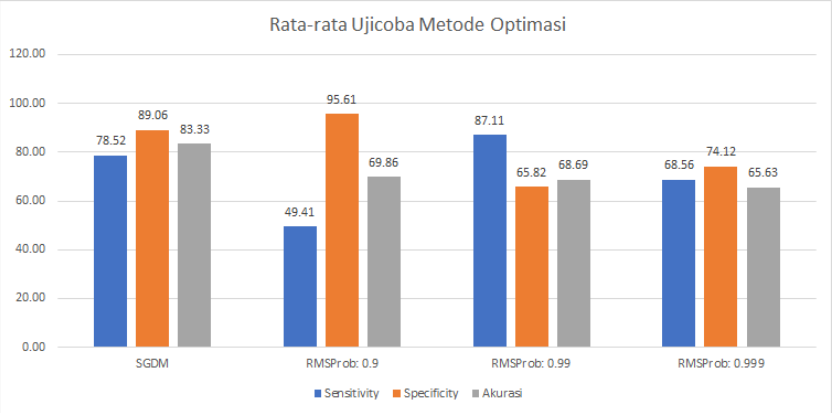
Batchsize	Learning rate	Sensitifitas (%)	Spesifisitas (%)	Akurasi (%)
8	0.1	100	48.44	65.63
	0.01	65.62	82.81	65.63
	0.001	100	50	66.67
	0.0001	100	98.44	98.96
16	0.1	100	46.88	64.58
	0.01	33.33	66.67	33.33
	0.001	100	48.44	65.63
	0.0001	100	100	100
32	0.1	100	81.25	87.5
	0.01	100	46.88	64.58
	0.001	64.58	82.29	66.67
	0.0001	96.88	98.44	97.92
64	0.1	100	46.88	64.58
	0.01	100	84.38	56.25
	0.001	100	84.38	56.25
	0.0001	96.88	98.44	97.92

Tabel 4.9 Hasil evaluasi ujicoba klasifikasi menggunakan metode Train RMSProb decay rate 0.999

Batchsize	Learning rate	Sensitifitas (%)	Spesifisitas (%)	Akurasi (%)
8	0.1	78.13	85.94	83.33
	0.01	65.62	82.81	65.62
	0.001	100	29.69	53.13
	0.0001	100	100	98.96
16	0.1	100	48.44	65.63
	0.01	65.62	82.81	65.62
	0.001	100	66.67	71.67
	0.0001	100	100	100
32	0.1	100	67.19	77.08
	0.01	66.67	82.29	66.67
	0.001	100	65.63	77.08
	0.0001	100	93.75	95.83
64	0.1	100	s84.38	65.63
	0.01	100	48.44	56.25
	0.001	96.88	100	98.96
	0.0001	100	95.31	96.88

Hasil evalusai uji coba klasifikasi penyakit leukemia menggunakan Inception V3, seperti yang ditunjukkan dalam tabel 4.5 - 4.9 memiliki nilai akurasi terbaik sebesar 100%, sensitifity sebesar 100%, dan spesitifity sebesar 100%. Hasil terbaik tersebut, dihasilkan dalam beberapa kombinasi hyper parameter, yakni dengan menggunakan model train: RMSProp 0.9 dengan ukuran *mini-batch* 8 dan nilai *learning rate* 0.001 serta model train RMSProp 0.99 dan RMSProb 0.999 de-

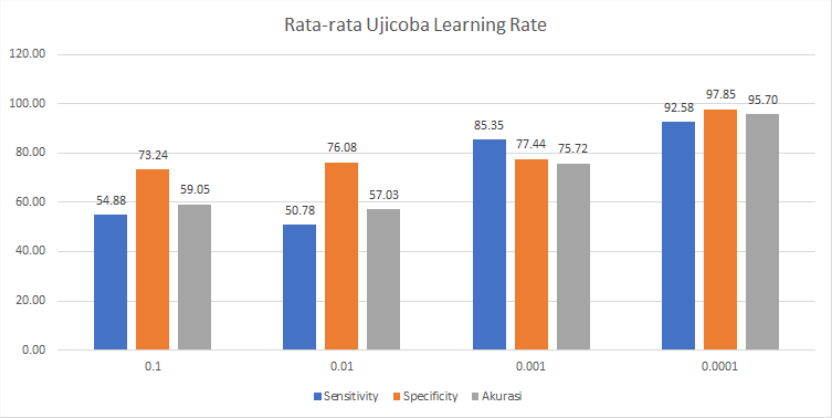
ngan ukuran *mini-batch* 16 dan nilai *learning rate* 0.0001.



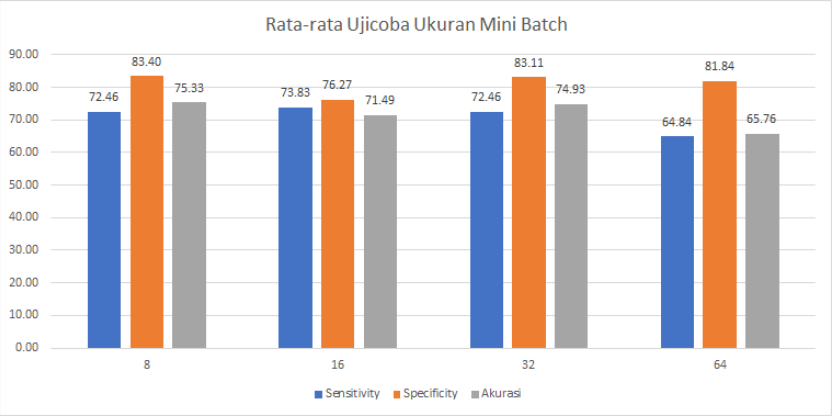
Gambar 4.20 Grafik Rata-rata Ujicoba Metode Optimasi

Gambar 4.20 menampilkan grafik rata-rata ujicoba metode optimasi. Optimasi menggunakan RMSProb tampak mampu meningkatkan rata-rata nilai spesifisity saat menggunakan nilai decay rate 0.9. Selainitu, penggunaan optimasi RMSProb juga tampak mampu meningkatkan rata-rata nilai sensitifity saat menggunakan nilai decay rate 0.99. Namun secara keseluruhan penggunaan RMSProb tidak memberikan pengaruh yang signifikan pada peningkatan kinerja model InceptionV3 dalam proses klasifikasi. hal ini dapat dilihat dari rata-rata akurasi penggunaan train SGDM yang lebih tinggi dari penggunaan Optimasi RMSProb.

Dari grafik 4.21 terlihat bahwasannya nilai learning rate memiliki pengaruh yang signifikan pada kinerja sistem. Semakin kecil nilai learning rate, kinerja sistem tampak semakin meningkat. Disisi lain, ukuran *mini-batch* tidak terlalu memberikan pengaruh pada akurasi sistem, seperti yang terlihat dari hasil ujicoba dalam grafik 4.22.



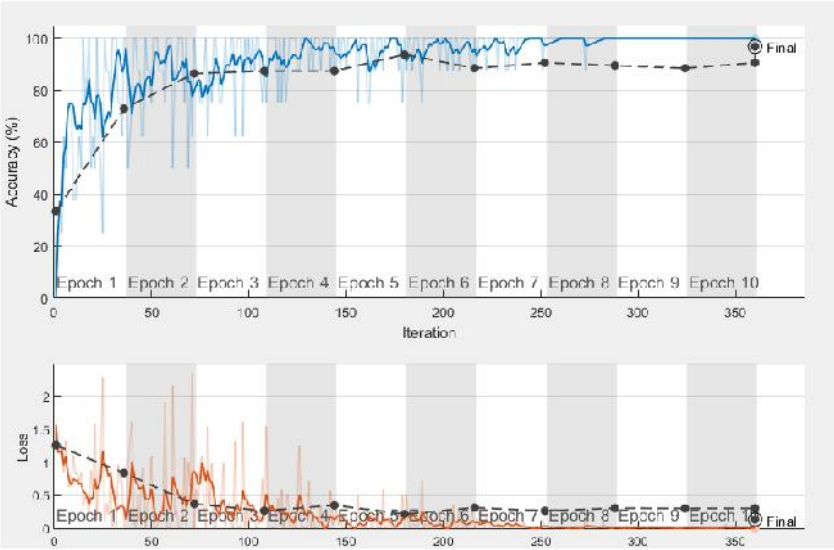
Gambar 4.21 Grafik Rata-rata Ujicoba Learning Rate



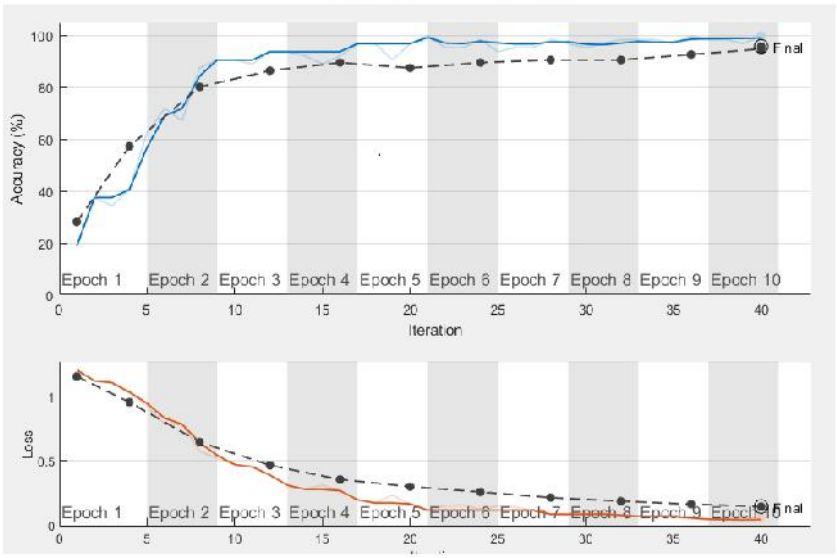
Gambar 4.22 Grafik Rata-rata Ujicoba Mini Batch Size

Penggunaan *batch normalization layers* pada arsitektur Inception-v3 mungkin memberikan pengaruh pada keadaan tersebut. Dimana *batch normalization layers* bekerja lebih optimal pada pengguanaan ukuran *mini-batch* yang besar, berbanding terbalik dengan kecenderungan akurasi sistem yang lebih baik pada ukuran *mini-batch* yang kecil. Meski tidak memiliki pengaruh signifikan pada akurasi sistem, grafik 4.23 menunjukkan bahwa selama proses pelatihan ukuran *mini-batch* memiliki pengaruh yang signifikan pada *convergence*. Hal ini dapat dilihat dari fluktuasi pada pelatihan dengan *mini-batch* besar, lebih kecil dibanding *mini-batch* dengan ukuran kecil. Penggunaan ukuran *mini-batch* yang besar juga mampu mengurangi waktu pelatihan seperti yang ditampilkan dalam grafik 4.24.

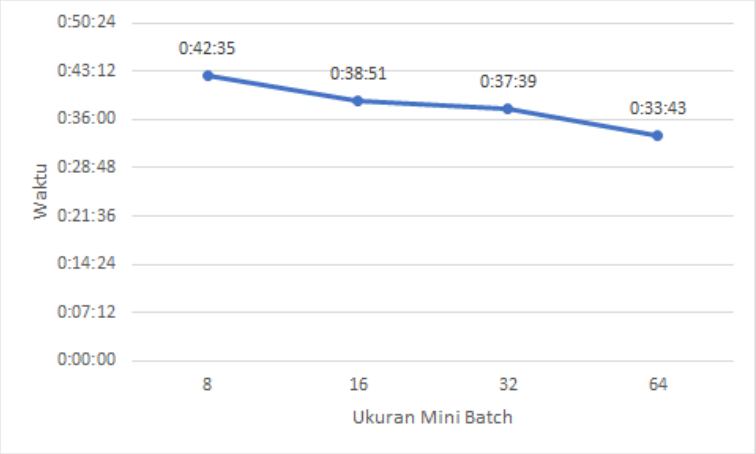
ukuran mini batch 8



ukuran batch size 16



Gambar 4.23 Grafik Perbandingan akurasi dan loss sistem selama pelatihan



Gambar 4.24 Grafik waktu pelatihan sistem

4.4. Integrasi Keilmuan Sains dan Teknologi dalam Al-Qur'an dan Hadits

Dalam Al-Quran, manusia senantiasa diperintahkan untuk memperhatikan segala ciptaan ALlah SWT. termasuk didalamnya adalah diri manusia. Sebagaimana firman Allah SWT. dalam surah Ar-Rum ayat 8.

أَوَلَمْ يَتَفَكَّرُوا فِي أَنفُسِهِم ۗ مَا خَلَقَ اللَّهُ السَّمَاوَاتِ وَالْأَرْضَ وَمَا بَيْنَهُمَا إِلَّا بِالْحَقِّ وَأَجَلٍ مُّسَمًّى ۗ وَإِنَّ كَثِيرًا مِّنَ النَّاسِ بِلِقَاءِ رَبِّهِمْ لَكَافِرُونَ

Arinya: Dan mengapa mereka tidak memikirkan tentang (kejadian) diri mereka? Allah tidak menciptakan langit dan bumi dan apa yang ada di antara keduanya melainkan dengan (tujuan) yang benar dan dalam waktu yang ditentukan. Dan sesungguhnya kebanyakan di antara manusia benar-benar mengingkari pertemuan dengan Tuhannya.

Terdapat banyak hal dalam sistem tubuh manusia yang dapat dipelajari untuk diterapkan dalam penyelesaian berbagi permasalahan dalam kehidupan. Salah satu contohnya adalah penerapan metode *Convolutional Neural Network*. CNN se-

bagai salah satu jenis penerapan metode Jaringan Syaraf Tiruan (JST), merupakan metode komputasi yang dimodelkan berdasarkan sistem syaraf manusia.

CNN juga merupakan salah satu jenis metode *mechine learning*, yang digunakan dalam mempelajari citra digital. Suatu mesin klasifikasi tentulah tidak langsung menghasilkan akurasi yang tinggi. Sehingga digunakan metode *gradient descent* untuk mengevaluasi nilai-nilai kesalahan mesin dan meningkatkan performanya secara otomatis. Manusia sebagai makhluk yang diberkahi dengan kecerdasan seharusnya tidak menyerah dalam menghadapi kegagalan. Karena melalui pengalaman tersebut seseorang mampu meningkatkan kemampuan serta kebijaksanaannya, sebagaimana sabda Rasulullah SAW:

عن ابي يعلى شداد بن اوس رضي الله عنه عن النبي صلى الله عليه و سلم قال : ألكيس من دان نفسه و عمل لما بعد الموت والعاجز من اتبع نفسه هواها وتمني على الله. (رواه الترمذي(وقال : هذا حديث صحيح.

Artinya: Dari Abu Yala yaitu Saddad ibnu Aus r.a. dari Nabi saw. Beliau bersabda : Orang yang cerdas ialah orang yang mampu mengintrospeksi dirinya dan suka beramal untuk kehidupannya setelah mati. Sedangkan orang lemah ialah orang yang selalu mengikuti hawa nafsunya dan berharap kepada Allah dengan harapan kosong. (H.R. At-Tirmidzi dan beliau berkata, Hadits Hasan.

Metode CNN telah berhasil digunakan dalam diagnosis citra sel penyakit. Setiap pixel citra akan diproses melalui barbagai jaringan sistem untuk mendapatkan informasi penting yang bahkan sulit diperhatikan hanya dengan menggunakan mata manusia. Sebagaimana sebuah citra, terdapat banyak informasi penting dari setiap ciptaan Allah SWT. baik dilangit maupun dibumi. Hanya saja banyak dian-

tara manusia yang tidak dapat memperoleh pelajaran. Sebagaimana firman Allah SWT dalam surah Saba ayat 9.

أَفَلَمْ يَرَوْا إِلَىٰ مَا بَيْنَ أَيْدِيهِمْ وَمَا خَلْفَهُم مِّنَ السَّمَاءِ وَالْأَرْضِ ۚ إِن نَّشَأْ نَخْسِفْ بِهِمُ الْأَرْضَ أَوْ نُسْقِطْ عَلَيْهِمْ كِسَفًا مِّنَ السَّمَاءِ ۚ إِنَّ فِي ذَٰلِكَ لَآيَةً لِّكُلِّ عَبْدٍ مُّنِيبٍ

Artinya: Maka apakah mereka tidak memperhatikan langit dan bumi yang ada di hadapan dan di belakang mereka? Jika Kami menghendaki, niscaya Kami benamkan mereka di bumi atau Kami jatuhkan kepada mereka kepingan-kepingan dari langit. Sungguh, pada yang demikian itu benar-benar terdapat tanda (kekuasaan Allah) bagi setiap hamba yang kembali (kepada-Nya).

Berdasarkan ayat ini, jika manusia mampu mempelajari setiap informasi yang diberikan Allah SWT. Kemudian mengantisipasi segala hal yang mungkin terjadi dan segara melakukan perbaikan, tentulah manusia akan dijauhkan dari segala bencana yang merupakan azab Allah SWT. Selaras dengan ayat diatas, metode CNN diterapkan guna mempermudah diagnosis penyakit secara valid. Sehingga seseorang mampu menangani penyakitnya dengan baik dan terhindar dari efek buruk yang timbul jika penyakit tersebut diabaikan.

Hasil penelitian ini berguna untuk mempermudah proses diagnosis penyakit leukemia secara dini. Penyakit leukemia yang merupakan suatu penyakit mematikan memiliki tingkat kesembuhan yang tinggi melalui pengobatan secara dini. Keberhasilan pengobatan berbagai penyakit mematikan merupakan bukti dari kebenaran hadits Nabi, yakni:

حَدَّثَنَا مُحَمَّدُ بْنُ الْمُثَنَّى حَدَّثَنَا أَبُو أَحْمَدَ الزُّبَيْرِيُّ حَدَّثَنَا عُمَرُ بْنُ سَعِيدِ بْنِ أَبِي حُسَيْنٍ قَالَ حَدَّثَنِي عَطَاءُ بْنُ أَبِي رَبَاحٍ عَنْ أَبِي هُرَيْرَةَ رَضِيَ اللَّهُ عَنْهُ عَنْ النَّبِيِّ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ مَا أَنْزَلَ اللَّهُ دَاءً إِلَّا أَنْزَلَ لَهُ شِفَاءً

Artinya: Telah menceritakan kepada kami Muhammad bin al-Mutsanna telah menceritakan kepada kami Abu Ahmad Az Zubairi telah menceritakan kepada kami ’Umar bin Sa’id bin Abu Husain dia berkata; telah menceritakan kepadaku ’Atha‘bin Abu Rabah dari Abu Hurairah radliallahu ’anhu dari Nabi shallallahu ’alaihi wasallam beliau bersabda: ”Allah tidak akan menurunkan penyakit melainkan menurunkan obatnya juga.” (HR Bukhari).

BAB V

PENUTUP

5.1. Kesimpulan

Kesimpulan yang dapat diambil dari hasil penelitian klasifikasi penyakit leukemia menggunakan CNN dengan arsitektur InceptionV3 adalah:

1. Metode CNN dengan arsitektur Inception-v3 mampu melakukan klasifikasi citra penyakit leukemia dengan hasil akurasi sebesar 100%, sensitifity 100% dan spesitifity 100%. Hasil terbaik tersebut, dihasilkan dalam beberapa kombinasi *hyperparameter*, yakni dengan menggunakan model pelatihan: RMSProp 0.9 dengan ukuran *mini-batch* 8 dan nilai *learning rate* 0.001 serta model pelatihan: RMSProp 0.99 dan RMSProb 0.999 dengan ukuran *mini-batch* 16 dan nilai *learning rate* 0.0001.

2. Akurasi tertinggi proses ujicoba klasifikasi dengan menggunakan augmentasi dan tanpa menggunakan augmentasi sama, yakni 98.96%. Meskipun penggguaan augmentasi dapat meningkatkan variasi image dan menghasilkan model yang mampu mengenali objek baru secara lebih general. Namun dalam penelitian ini diketahui bahwa tidak terdapat pengaruh yang besar dalam penggggunaan metode augmentasi selama proses ujicoba. Hal ini dapat dilihat dari rata-rata ujicoba akurasi klasifikasi dengan menggunakan augmentasi dan tanpa menggunakan augmentasi adalah sebesar 85.09% dan 86.58%. Hasil akurasi tersebut juga menunjukkan bahwa terdapat pengaruh yang signifikan

dalam pengggunaan metode augmentasi terhadap kinerja sistem. Juga terlihat bahwa akurasi proses ujicoba tanpa menggunakan augmentasi lebih baik dibanding menggunakan augmentasi.

3. Ukuran *mini-batch* tidak terlalu memberikan pengaruh pada akurasi sistem. Namun selama proses pelatihan ukuran *mini-batch* memiliki pengaruh yang signifikan pada *convergence*. Hal ini dapat dilihat dari fluktuasi pada pelatihan dengan ukuran *mini-batch* besar, lebih kecil dibanding penggunaan *mini-batch* dengan ukuran kecil. Selain itu penggunaan ukuran *mini-batch* yang besar juga mampu megnurangi waktu pelatihan.

5.2. Saran

Penelitian mengenai klasifikasi penyakit leukemia menggunakan CNN dengan arsitektur InceptionV3 ini, tentu masih banyak terdapat kekurangan, sehingga sangat diperlukan saran yang bersifat membangun untuk menjadikan sistem klasifikasi ini menjadi semakin baik. Beberapa hal yang disarankan untuk penelitian mendatang yaitu:

1. Melakukan beragam metode optimasi terhadap arsitektur InceptionV3, serta ujicoba hyperparameter lainnya. Contohnya seperti pengaruh penggunaan *batch normalization layer* pada arsitektur Inception-v3. Arsitektur InceptionV3 juga dapat dimodifikasi atau digabungkan dengan metode klasifikasi lain, seperti metode SVM dan lain-lain.

2. Menggunakan data citra penyakit dengan lebih banyak kelas. Disarankan juga untuk menggunakan dataset dengan jumlah yang lebih besar, sehingga mampu menguji kinerja sistem dengan lebih baik.

DAFTAR PUSTAKA

American Cancer Society. (2021). *"Leukemia"*. https://cancerstatisticscenter.cancer.org. diakses pada 31 Maret 2021.

Annapurani, K., and Ravilla, D. (2019). *"CNN based image classification model*. International Journal of Innovative Technology and Exploring Engineering, 8(11 Special Issue), 11061114. https://doi.org/10.35940/ijitee.K1225.09811S19

Arrofiqoh, E. N., dan Harintaka, H. (2018). Implementasi Metode Convolutional Neural Network Untuk Klasifikasi Tanaman Pada Citra Resolusi Tinggi. Geomatika, 24(2), 61. https://doi.org/10.24895/jig.2018.24-2.810

Boue, L. (2018). Deep learning for pedestrians: backpropagation in CNNs. SAP Labs.

Dia Rofinda, Z. (2012) *"Kelainan Hemostasis pada Leukemia"*, Jurnal Kesehatan Andalas, 1(2), pp. 6874. doi: 10.25077/jka.v1i2.40.

Donna M. Bozzone (2009) *"Leukemia"*. Available at: http://library1.nida.ac.th/termpaper6/sd/2554/19755.pdf.

F. Schilling (2016). *"The Effect of Batch Normalization on Deep Convolutional Neural Networks"*, Dissertation.

Firmino, M. et al. (2016) *"Computer-aided detection (CADe) and diagnosis (CADx) system for lung cancer with likelihood of malignancy"*, BioMedical Engineering Online. BioMed Central, 15(1), pp. 117. doi: 10.1186/s12938-015-0120-7.

Gonzalez, Rafael C., Richard E. Woods, and Barry R. Masters. 2009. *"Digital Image Processing, Third Edition"*. Journal of Biomedical Optics 14 (2): 029901. https://doi.org/10.1117/1.3115362.

Habib Hanga, A., Hussain, I., Habib, A., Alalyani, M., Hussain, I., and Almutheibi, M. S. (2015). Brief review on Sensitivity, Specificity and Predictivities Brief review on Sensitivity, Specificity and Predictivities 1. Article in IOSR Journal of Dental and Medical Sciences, 14, 6468. https://doi.org/10.9790/0853-14456468

Hasma, Yunita Aulia, and Widya Silfianti. 2018. Implementasi Deep Learning Menggunakan Framework Tensorflow Dengan Metode Faster Regional Convolutional Neural Network Untuk Pendeteksian Jerawat. Jurnal Ilmiah Teknologi Dan Rekayasa 23 (2): 89102. https://doi.org/10.35760/tr.2018.v23i2.2459.

He, Jian, Xuemei Pu, Menglong Li, Chuan Li, and Yanzhi Guo. 2020. Deep Convolutional Neural Networks for Predicting Leukemia-Related Transcription Factor Binding Sites from DNA Sequence Data. Chemometrics and Intelligent Laboratory Systems 199 (December 2019): 103976. https://doi.org/10.1016/j.chemolab.2020.103976.

Huang, Furong, Peiwen Guang, Fucui Li, Xuewen Liu, Weimin Zhang, and Wendong Huang. 2020. AML, ALL, and CML Classification and Diagnosis Based on Bone Marrow Cell Morphology Combined with Convolutional Neural Network: A STARD Compliant Diagnosis Research. Medicine 99 (45): e23154. https://doi.org/10.1097/MD.0000000000023154.

Ioffe, S., and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. http://arxiv.org/abs/1502.03167

Kingma, D. P., and Ba, J. (2014). Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980

L. H. S. Vogado, R. M. S. Veras, F. H. D. Araujo, R. R. V. Silva, and K. R. T. Aires, *"Leukemia diagnosis in blood slides using transfer learning in CNNs and SVM for classification"* Eng. Appl. Artif. Intell., vol. 72, no. October 2017, pp. 415422, 2018, doi: 10.1016/j.engappai.2018.04.024.

Labati, Donida, Ruggero, Vincenzo Piuri, and Fabio Scotti. 2011. ALL-IDB: THE ACUTE LYMPHOBLASTIC LEUKEMIA IMAGE DATABASE FOR IMAGE PROCESSING Ruggero. Ieee International Conference On Image Processing, 208992.

Li, Yanfen, Hanxiang Wang, L. Minh Dang, Abolghasem Sadeghi-Niaraki, and Hyeonjoon Moon. 2020. Crop Pest Recognition in Natural Scenes Using Convolutional Neural Networks. Computers and Electronics in Agriculture 169 (December 2019): 105174. https://doi.org/10.1016/j.compag.2019.105174.

Matek, C., Schwarz, S., Marr, C., and Spiekermann, K. (2019). *"A Single-cell Morphological Dataset of Leukocytes from AML Patients and Non-malignant Controls [Data set]"*. The Cancer Imaging Archive. https://doi.org/10.7937/tcia.2019.36f5o9ld

Nugraha, I. B. A. and Adnyana, W. L. (2017) *"Seorang Penderita Acute Myeloid Leukemia (AML) M4 yang Mengalami Tumor Lysis Syndrome"*, Jurnal Penyakit Dalam Udayana, 1(1), pp. 816. doi: 10.36216/jpd.v1i1.8.

Pakan, P. D. (2018) *"Klasifikasi Kanker Leukimia Menggunakan Microarray Ekspresi Gen"*, 4(2), pp. 7883.

Qian, X., and Klabjan, D. (2020). "The Impact of the Mini-batch Size on the Variance of Gradients in Stochastic Gradient Descent". http://arxiv.org/abs/2004.13146

Rafael C. Gonzalez dan Richard E. Woods (2018). Digital Image Proccessing. New York: Pearson.

Rahmawati, A. and Mutiah, R. (2014) *"Potensi Ekstrak Daun Widuri (Calotropis gigantea)"*.

Rama, A., A. Kumaravel, and C. Nalini. 2019. Preprocessing Medical Images for Classification Using Deep Learning Techniques. International Journal of Innovative Technology and Exploring Engineering 8 (9 Special Issue 3): 71116. https://doi.org/10.35940/ijitee.I3147.0789S319.

Rehman, Amjad, Naveed Abbas, Tanzila Saba, Syed Ijaz ur Rahman, Zahid Mehmood, and Hoshang Kolivand. 2018. Classification of Acute Lymphoblastic Leukemia Using Deep Learning. Microscopy Research and Technique 81 (11): 131017. https://doi.org/10.1002/jemt.23139.

Rendra, M., Yaswir, R. and Hanif, A. M. (2013) *"Gambaran Laboratorium Leukemia Kronik di Bagian Penyakit Dalam RSUP Dr. M. Djamil Padang"*, Jurnal Kesehatan Andalas, 2(3), p. 141. doi: 10.25077/jka.v2i3.153.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. http://arxiv.org/abs/1609.04747

Saifullah, S. (2020). Segmentasi Citra Menggunakan Metode Watershed Transform Berdasarkan Image Enhancement Dalam Mendeteksi Embrio Telur. Systemic: Information System and Informatics Journal, 5(2), 5360. https://doi.org/10.29080/systemic.v5i2.798

Sanjaya, Joseph, and Mewati Ayub. 2020. Augmentasi Data Pengenalan Citra Mobil Menggunakan Pendekatan Random Crop, Rotate, Dan Mixup. Jurnal Teknik Informatika Dan Sistem Informasi 6 (2): 31123. https://doi.org/10.28932/jutisi.v6i2.2688.

Sergey Ioffe dan Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift.

Singh, D., and Singh, B. (2020). Investigating the impact of data normalization on classification performance. Applied Soft Computing, 97. https://doi.org/10.1016/j.asoc.2019.105524

Smarter Health Pte Ltd. 2017. *Leukemia Limfositik Kronis*. Online at https://www.smarterhealth.id/penyakit/leukemia-limfositik-kronis/, accessed 10 Juni 2021.

Swapna, M, Yogesh Kumar Sharma, and B M G Prasadh. 2020. CNN Architectures: Alex Net, Le Net, VGG, Google Net, Res Net. International Journal of Recent Technology and Engineering 8 (6): 95360. https://doi.org/10.35940/ijrte.f9532.038620.

Szegedy, Christian, Vincent Vanhoucke, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going Deeper with Convolutions. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 07-12-June-2015: 19. https://doi.org/10.1109/CVPR.2015.7298594.

World Health Organization 2020. ”Estimated number of new cases in 2020, worldwide, both sexes, all ages”. viewed 11 Juni 2021, https://gco.iarc.fr/

The Leukemia and Lymphoma Society (2018) *”Updated Data on Blood Cancers”*, p. 32.

Wilson, D. R., and Martinez, T. R. (2001). ”The need for small learning rates on large problems. Proceedings of the International Joint Conference on Neural Networks”, 1, 115119. https://doi.org/10.1109/ijcnn.2001.939002

Yadav, Shilpi, Vineet Chandan, Aaisha Makkar, Lakshit Sama, Greater Noida, and Computer Science. 2020. Recognition of Pandemic Virus Covid-19 Using Inception-V3 29 (12): 167579.

Yadav, S. S., dan Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis. Journal of Big Data, 6(1). https://doi.org/10.1186/s40537-019-0276-2

Zhang, Z. (2016). Derivation of Backpropagation in Convolutional Neural Network (CNN). University of Tennessee, Knoxvill, TN

Zhou, X. (2018). Understanding the Convolutional Neural Networks with Gradient Descent and Backpropagation. Journal of Physics: Conference Series, 1004(1). https://doi.org/10.1088/1742-6596/1004/1/012028

Zufar, M. dan Setiyono, B. (2016). Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time (Vol. 5, Issue 2).

...