

**PERBANDINGAN KINERJA ARSITEKTUR
VGG19-LSTM DAN BLIP DALAM VISUAL
QUESTION ANSWERING (VQA) PADA CITRA MEDIS**

TUGAS AKHIR

Diajukan untuk melengkapi tugas-tugas dan
memenuhi syarat-syarat guna memperoleh gelar Sarjana Komputer

Oleh:

ABDUL HAFIDH
2008107010056



**PROGRAM STUDI INFORMATIKA JURUSAN INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS SYIAH KUALA, BANDA ACEH
JUNI, 2024**

PENGESAHAN

PERBANDINGAN KINERJA ARSITEKTUR VGG19-LSTM DAN BLIP DALAM VISUAL *QUESTION ANSWERING* (VQA) PADA CITRA MEDIS

Oleh:

Nama : Abdul Hafidh
NPM : 2008107010056
Jurusan : Informatika

Menyetujui:

Pembimbing I

Pembimbing II

Alim Misbullah, S.Si., M.S.
NIP. 198806032019031011

Laina Farsiah, S.Si., M.S.
NIP. 198902032022032004

Mengetahui:

Koordinator Program Studi Informatika
Universitas Syiah Kuala,

Alim Misbullah, S.Si., M.S.
NIP. 198806032019031011

ABSTRAK

Visual Question Answering (VQA) merupakan tugas untuk menjawab pertanyaan berdasarkan gambar. Dalam dunia medis, VQA dapat membantu ahli kesehatan untuk mendapatkan informasi dari citra medis. Namun, citra medis memiliki tantangan tersendiri, seperti variasi pertanyaan yang kompleks dan tingkat keabstrakkan yang tinggi, sehingga memerlukan model VQA yang dapat menangani hal tersebut. Penelitian ini melakukan eksperimen dengan dua model, yaitu VGG19-LSTM dan BLIP, dengan *dataset* PathVQA dan VQA-RAD. Model VGG19-LSTM menggabungkan *Convolutional Neural Network* (CNN) dengan *Long Short-Term Memory* (LSTM). BLIP adalah model terunifikasi untuk tugas *vision-language* yang menggunakan *Vision Transformers* (ViT) sebagai *image encoder* dan *transformer* sebagai *text encoder*. Hasil eksperimen menunjukkan keunggulan model BLIP dalam kedua *dataset*. BLIP yang dilatih dengan *dataset* PathVQA dengan konfigurasi *hyperparameter* 15 epochs, *batch size* 8, dan *learning rate* 1×10^{-5} setelah augmentasi pertanyaan mencapai akurasi 83,91%, akurasi pertanyaan *close-ended* 97,43%, dan akurasi pertanyaan *open-ended* 66,15%. Model BLIP yang dilatih *dataset* VQA-RAD dengan konfigurasi *hyperparameter* 45 epochs, *batch size* 8, dan *learning rate* 5×10^{-5} setelah augmentasi pertanyaan mencapai akurasi 82,86%, akurasi pertanyaan *close-ended* 87,85%, dan akurasi pertanyaan *open-ended* 76,82%. Pada tahap inferensi, model BLIP yang dikuantisasi menunjukkan performa yang lebih baik dibandingkan tanpa kuantisasi. Sebelum dilakukan kuantisasi, ukuran model BLIP adalah 1467,73 MB dengan waktu inferensi 0,24 detik pada *dataset* PathVQA, dan 1467,73 MB dengan waktu inferensi 0,21 detik pada *dataset* VQA-RAD. Setelah dilakukan kuantisasi, ukuran model BLIP menjadi 508,21 MB dengan waktu inferensi 0,20 detik pada *dataset* PathVQA, dan 508,20 MB dengan waktu inferensi 0,17 detik pada *dataset* VQA-RAD.

Kata kunci : *Visual Question Answering, Convolutional Neural Network, Long Short-Term Memory, Transformer VGG19-LSTM, BLIP, Citra Medis*

ABSTRACT

Visual Question Answering (VQA) is a task to answer questions based on images. In the medical world, VQA can help health experts to get information from medical images. However, medical images have their own challenges such as complex question variations and high levels of abstraction. Thus, it requires a VQA model that can handle these challenges. This research conducts experiments with two models, VGG19-LSTM and BLIP with PathVQA and VQA-RAD datasets. The VGG19-LSTM model combines Convolutional Neural Network (CNN) with Long Short-Term Memory (LSTM). BLIP is a unified model for vision-language tasks that uses Vision Transformers (ViT) as image encoders and transformers as text encoders. The experimental results show the superiority of the BLIP model in both datasets. BLIP trained with the PathVQA dataset with hyperparameter configuration of 15 epochs, batch size 8, and learning rate 1×10^{-5} after question augmentation achieves an accuracy of 83.91%, close-ended question accuracy of 97.43%, and open-ended question accuracy of 66.15%. After that, the BLIP model trained with the VQA-RAD dataset with hyperparameter configuration of 45 epochs, batch size 8, and learning rate 5×10^{-5} after question augmentation achieves an accuracy of 82.86%, close-ended question accuracy of 87.85%, and open-ended question accuracy of 76.82%. In the inference stage, the quantized BLIP model shows better performance than without quantization. Before quantization, the size of the BLIP model is 1467.73 MB with an inference time of 0.24 seconds on the PathVQA dataset, and 1467.73 MB with an inference time of 0.21 seconds on the VQA-RAD dataset. After quantization, the size of the BLIP model becomes 508.21 MB with an inference time of 0.20 seconds on the PathVQA dataset, and 508.20 MB with an inference time of 0.17 seconds on the VQA-RAD dataset.

Keywords : Visual Question Answering, Convolutional Neural Network, Long Short-Term Memory, Transformer VGG19-LSTM, BLIP, Medical Image

KATA PENGANTAR

Segala puji dan syukur kehadiran Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya kepada kita semua, sehingga penulis dapat menyelesaikan penulisan tugas akhir yang berjudul **“Perbandingan Kinerja Arsitektur VGG19-LSTM dan BLIP dalam Visual Question Answering (VQA) pada Citra Medis”** yang telah dapat diselesaikan sesuai rencana. Penulis banyak mendapatkan berbagai pengarahan, bimbingan, dan bantuan dari berbagai pihak. Oleh karena itu, melalui tulisan ini penulis mengucapkan rasa terima kasih kepada:

1. Dewi Ratna Sari, SE.MM, dan Fadhli, SE.Ak, sebagai kedua orang tua penulis, senantiasa memberikan dukungan penuh terhadap aktivitas dan kegiatan yang dilakukan penulis, baik secara moral maupun material, serta menjadi motivasi terbesar bagi penulis untuk menyelesaikan tugas akhir ini.
2. Prof. Dr. Taufik Fuadi Abidin, S.Si., M.Tech, sebagai Dekan Fakultas MIPA Universitas Syiah Kuala.
3. Dr. Nizamuddin, M.Info.Sc., sebagai Ketua Jurusan Informatika Fakultas MIPA Universitas Syiah Kuala.
4. Bapak Alim Misbullah, S.Si., M.S., selaku Dosen Pembimbing I dan Ketua Program Studi Informatika, telah memberikan bimbingan dan arahan yang berharga kepada penulis, membantu penulis dalam menyelesaikan Tugas Akhir ini.
5. Ibu Laina Farsiah, S.Si., M.S., selaku Dosen Pembimbing II, memberikan bimbingan dan arahan yang sangat dibutuhkan oleh penulis, yang akhirnya membantu penulis menyelesaikan tugas akhir ini.
6. Ibu Sri Azizah Nazhifah, S.Kom., M.Sc., selaku dosen pembahas, telah membantu penulis dan kawan-kawan untuk mempersiapkan ruang lingkup tugas akhir, serta memberikan arahan yang diperlukan.
7. Bapak dr. Muhammad Ansari Adista, M.Pd.Ked., Sp.N., selaku dosen pembahas yang telah memberikan masukan dan saran yang sangat berharga bagi penulis dalam menyelesaikan tugas akhir ini.
8. Ibu Dalila Husna Yunardi, B.Sc., M.Sc., selaku dosen wali, memberikan *support* dan pedoman yang diperlukan selama penulis menempuh pendidikan.
9. Muhamad Razan Fawwaz, Amar Suhendra, Khairul Umam Albi, Muhamad Rudy Hidayat, Muhamad Raja Furqan, Teuku Nabil Muhamad Dhuha, Yoan Rifqi Candra, dan Haris Daffa, telah menemani dan memberi dukungan kepada penulis selama empat tahun perkuliahan di jurusan Informatika USK.
10. Anas Naufal Al-Kiram, Muhamad Hanif, Daffa Mudhaffar, Muhamad Ikhsan Fikri, dan Teuku Muhamad Roy Adrian, teman yang selalu mengingatkan, dan membantu penulis untuk selalu berusaha dalam menyelesaikan tugas akhir di semester 8.

11. Sahabat dan teman-teman seperjuangan dari Jurusan Informatika USK 2020 lainnya, telah memberikan dukungan moral yang luar biasa selama penulis menempuh pendidikan di jurusan Informatika USK.
12. Seluruh Dosen dan Staf di Jurusan Informatika Fakultas MIPA, atas ilmu dan bimbingan yang diberikan kepada penulis selama perkuliahan, menjadi bagian tak terpisahkan dalam proses pembelajaran penulis.

Penulis mengakui adanya kekurangan dalam tulisan ini, baik dari aspek materi, metode, maupun bahasa yang digunakan. Oleh karena itu, penulis mengundang kritik dan saran yang konstruktif dari para pembaca untuk meningkatkan kualitas Tugas Akhir ini. Penulis berharap bahwa tulisan ini dapat memberikan manfaat bagi banyak orang dan berkontribusi pada kemajuan ilmu pengetahuan.

DAFTAR ISI

Halaman Judul	i
Halaman Pengesahan	ii
Abstrak	iii
<i>Abstract</i>	iv
Kata Pengantar	v
Daftar Isi	vii
Daftar Tabel	ix
Daftar Gambar.....	x
Daftar Lampiran.....	xi
BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah	2
1.3. Tujuan Penelitian.....	2
1.4. Manfaat Penelitian	3
BAB II TINJAUAN KEPUSTAKAAN	4
2.1. Citra Medis	4
2.2. <i>Deep Learning</i>	5
2.2.1. <i>Artificial Neural Network</i>	6
2.2.2. Fungsi Aktivasi	6
2.2.3. Fungsi <i>loss</i>	8
2.2.4. Fungsi Optimasi	9
2.3. <i>Convolutional Neural Network</i>	10
2.4. <i>Long Short Term Memory</i>	11
2.5. <i>Transfer Learning</i>	11
2.6. Matriks Evaluasi	15
2.6.1. Akurasi.....	16
2.6.2. <i>Bilingual Evaluation Understudy</i>	16
2.7. <i>Visual Question Answering</i>	18
2.7.1. <i>Computer Vision</i>	19
2.7.2. <i>Natural Language Processing</i>	19
2.8. <i>Medical Visual Question Answering Dataset</i>	20
2.8.1. <i>PathVQA Dataset</i>	20
2.8.2. <i>VQA-RAD Dataset</i>	20
2.9. Penelitian Terkait.....	21
BAB III METODOLOGI PENELITIAN	23
3.1. Waktu dan Lokasi Penelitian	23
3.2. Jadwal Pelaksanaan.....	23
3.3. Alat dan Bahan	23
3.3.1. Perangkat Keras.....	23
3.3.2. Perangkat Lunak.....	23
3.4. Metode Penelitian.....	24

3.4.1. Identifikasi Masalah	25
3.4.2. Studi Literatur.....	25
3.4.3. Pengumpulan Data	26
3.4.4. Pemrosesan Data	27
3.4.5. Membangun Model	28
3.4.6. Melatih Model.....	28
3.4.7. Perbandingan Hasil	29
3.4.8. Membangun Sistem Medis Cerdas Berbasis Web	29
BAB IV HASIL DAN PEMBAHASAN	30
4.1. Pengumpulan Data.....	30
4.2. Pengembangan Model	34
4.2.1. VGG19-LSTM	34
4.2.2. BLIP	37
4.3. Augmentasi Data	43
4.4. Analisis Hasil	44
4.5. Pengembangan Sistem Medis Cerdas Berbasis Web	56
BAB V KESIMPULAN DAN SARAN	60
5.1. Kesimpulan	60
5.2. Saran	61
DAFTAR PUSTAKA	62
LAMPIRAN.....	66

DAFTAR TABEL

Tabel 2.1	Deskripsi dari masing-masing nilai BLEU.....	17
Tabel 3.1	Jadwal pelaksanaan penelitian	23
Tabel 3.2	Proses <i>case folding</i>	27
Tabel 3.3	Proses <i>remove punctuation</i>	27
Tabel 3.4	Proses <i>stemming</i>	28
Tabel 3.5	Proses tokenisasi	28
Tabel 4.1	Proses <i>case folding</i> pada sampel dataset PathVQA dan VQA-RAD	31
Tabel 4.2	Proses <i>remove punctuation</i> pada sampel dataset PathVQA dan VQA-RAD	32
Tabel 4.3	Proses <i>stemming</i> pada sampel dataset PathVQA dan VQA-RAD	32
Tabel 4.4	Proses tokenisasi pada sampel dataset PathVQA dan VQA-RAD	33
Tabel 4.5	Contoh pertanyaan yang telah diaugmentasi	44
Tabel 4.6	Konfigurasi <i>hyperparameter</i> pada awal eksperimen	45
Tabel 4.7	Hasil akurasi pelatihan dengan VGG19-LSTM.....	46
Tabel 4.8	Hasil BLEU pada pertanyaan open-ended dengan VGG19-LSTM	47
Tabel 4.9	Hasil akurasi pelatihan dengan BLIP	48
Tabel 4.10	Hasil BLEU pada pertanyaan open-ended dengan BLIP	49
Tabel 4.11	Hasil akurasi setelah dilakukan augmentasi data pertanyaan	50
Tabel 4.12	Hasil nilai BLEU setelah dilakukan augmentasi data pertanyaan	51
Tabel 4.13	Spesifikasi model BLIP sebelum dan sesudah dikuantisasi	58
Tabel 4.14	Daftar API <i>endpoint</i> pada sistem medis cerdas berbasis web	58

DAFTAR GAMBAR

Gambar 2.1	Kategori pendekatan <i>deep learning</i>	5
Gambar 2.2	Perbandingan teknik optimasi Adam dengan teknik optimasi lainnya (Kingma and Ba, 2014).....	9
Gambar 2.3	Arsitektur CNN (Wardani and Leonardi, 2023).....	10
Gambar 2.4	Arsitektur LSTM (Luo et al., 2023).....	11
Gambar 2.5	Konsep penggunaan <i>transfer learning</i> (Mukhlif et al., 2023)	12
Gambar 2.6	Arsitektur VGG19 (Nguyen et al., 2022)	13
Gambar 2.7	Arsitektur BLIP (Li et al., 2022).....	13
Gambar 2.8	<i>Confusion matrix</i> (Kulkarni et al., 2020)	15
Gambar 2.9	Vanilla VQA <i>network</i> model (Srivastava et al., 2021).....	18
Gambar 3.1	Diagram alir penelitian.....	25
Gambar 3.2	Sampel data PathVQA	26
Gambar 3.3	Sampel data VQA-RAD	26
Gambar 4.1	Sampel data dari <i>dataset</i> PathVQA	30
Gambar 4.2	Sampel data dari <i>dataset</i> VQA-RAD.....	30
Gambar 4.3	Struktur arsitektur VGG19-LSTM	35
Gambar 4.4	Perbandingan akurasi model BLIP dengan <i>dataset</i> PathVQA sebelum dan sesudah dilakukan augmentasi data pertanyaan	52
Gambar 4.5	Perbandingan akurasi model BLIP dengan <i>dataset</i> VQA-RAD sebelum dan sesudah dilakukan augmentasi data pertanyaan	53
Gambar 4.6	Perbandingan akurasi model VGG19-LSTM dengan <i>dataset</i> PathVQA sebelum dan sesudah dilakukan augmentasi data pertanyaan.....	54
Gambar 4.7	Perbandingan akurasi model VGG19-LSTM dengan <i>dataset</i> VQA-RAD sebelum dan sesudah dilakukan augmentasi data pertanyaan.....	55
Gambar 4.8	Halaman utama sistem medis cerdas berbasis web	56
Gambar 4.9	Halaman untuk menjawab pertanyaan berdasarkan citra patologi	57
Gambar 4.10	Halaman untuk menjawab pertanyaan berdasarkan citra radiologi	57

DAFTAR LAMPIRAN

Lampiran 1. <i>Layer vision</i> pada model BLIP	66
Lampiran 2. <i>Layer text encoder</i> pada model BLIP	67
Lampiran 3. <i>Layer text decoder</i> pada model BLIP	68
Lampiran 4. Hasil pelatihan dengan VGG19-LSTM untuk pertanyaan <i>close ended</i>	69
Lampiran 5. Hasil pelatihan dengan VGG19-LSTM untuk pertanyaan <i>open ended</i>	70
Lampiran 6. Hasil pelatihan dengan VGG19-LSTM untuk semua pertanyaan ...	71
Lampiran 7. Hasil pelatihan dengan BLIP untuk pertanyaan <i>close ended</i>	72
Lampiran 8. Hasil pelatihan dengan BLIP untuk pertanyaan <i>open ended</i>	73
Lampiran 9. Hasil pelatihan dengan BLIP untuk semua pertanyaan	74
Lampiran 10. Proses demo pada citra patologi di halaman web	76
Lampiran 11. Proses demo pada citra radiologi di halaman web	78
Lampiran 12. Biodata Penulis	80

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Citra medis, seperti *Magnetic Resonance Imaging* (MRI), *X-Ray*, *Ultrasonography* (USG), *Endoscopy*, *Computed Tomography* (CT-Scan), *Nuclear Medicine*, dan lain-lain, menjadi fokus penelitian utama dalam dunia medis (Kusuma and Kusumadewi, 2020). Profesional medis menggunakan berbagai teknik untuk mendekripsi dan menganalisis penyakit pada pasien. Dua cabang ilmu yang berkaitan dengan diagnosa penyakit, yaitu patologi dan radiologi, memainkan peran krusial dalam memahami citra medis (Sorace et al., 2012). Meskipun demikian, kesalahan analisis citra medis oleh tenaga medis terkadang bisa saja terjadi karena sifat manusia yang rentan terhadap kesalahan (Mauli, 2018).

Ketika manusia melakukan pekerjaan yang repetitif, ini dapat menyebabkan kelelahan dan berkurangnya konsentrasi, sehingga meningkatkan kemungkinan kesalahan. Hal ini berbeda dengan *Artificial Intelligence* (AI), AI tidak memiliki perasaan, sehingga AI akan tetap menghasilkan kualitas pekerjaan yang konsisten seiring berjalannya waktu (Fernando and Harsiti, 2019). Pada bidang medis AI tidak diharapkan untuk menggantikan dokter manusia dalam skala besar. Sebaliknya, AI kemungkinan akan memberdayakan praktik kedokteran dengan meningkatkan upaya dokter dan mengatasi masalah seperti kelelahan dokter (Basu et al., 2020). Oleh sebab itu, dibutuhkan sebuah sistem yang dapat membantu tenaga medis dalam menjawab permasalahan yang terdapat pada citra medis. Salah satu solusi yang dapat digunakan adalah dengan membangun sistem *Visual Question Answering* (VQA). Dalam konteks medis VQA dapat memberikan manfaat bagi dokter dan pasien yang mana dokter bisa memperoleh jawaban yang diperoleh dari sistem VQA sebagai bentuk dalam pengambilan keputusan. Sedangkan pasien bisa mengajukan pertanyaan ke sistem VQA dengan gambar medis yang ada pada dirinya untuk mengetahui kondisi kesehatanya (Nguyen et al., 2019).

Sistem VQA medis ini juga dapat berperan sebagai asisten yang memiliki pengetahuan yang luas. Sebagai contoh, pendapat tambahan atau opini kedua dari sistem VQA dapat membantu tenaga medis dalam menjawab pertanyaan berdasarkan citra medis yang diberikan dan mengurangi potensi kesalahan dalam diagnosis pada saat bersamaan (Tschandl et al., 2020). Meskipun begitu, penelitian mengenai sistem VQA medis masih terbatas karena menghadapi beberapa tantangan khusus yang harus dihadapi, seperti keanekaragaman pertanyaan yang dapat diajukan, yang memerlukan pemahaman yang mendalam tentang citra medis dan pertanyaan untuk konteks medis,

serta memiliki keterbatasan dalam interpretabilitas yakni kemampuan untuk menanyakan model tentang wilayah gambar yang spesifik (Lin et al., 2023). Akan tetapi, dengan sering dilakukannya penelitian terkait *medical visual question answering*, diharapkan dapat mengatasi tantangan yang ada, sehingga penelitian *medical visual question answering* dapat berkembang dengan baik seiring berjalannya waktu.

Penelitian ini bertujuan untuk mengembangkan sebuah sistem VQA yang mampu memberikan jawaban terhadap pertanyaan yang diajukan berdasarkan citra medis. Sistem ini akan memanfaatkan dua *dataset* yaitu PathVQA yang terkait dengan patologi, dan VQA-RAD yang fokus pada bidang radiologi. Dalam penelitian ini akan menggunakan pendekatan *deep learning* yang memanfaatkan teknik *transfer learning*. Teknik *transfer learning* ini digunakan untuk memanfaatkan model yang sudah dilatih sebelumnya pada *dataset* yang berbeda. Hasil dari penelitian ini diharapkan dapat memberikan gambaran mengenai sistem VQA untuk citra medis dengan *deep learning* dan memberikan kontribusi terkait penelitian *medical visual question answering*.

1.2 RUMUSAN MASALAH

Berdasarkan latar belakang yang telah diuraikan, permasalahan dalam penelitian ini dapat dirumuskan sebagai berikut:

1. Bagaimana membangun model VQA menggunakan citra medis dengan menerapkan teknik *transfer learning*?
2. Bagaimana membandingkan performa model VQA pada arsitektur VGG19-LSTM dan BLIP untuk *dataset* PathVQA dan VQA-RAD?
3. Bagaimana menerapkan model VQA terbaik untuk menjawab pertanyaan berdasarkan citra medis?

1.3 TUJUAN PENELITIAN

Adapun maksud dan tujuan dari penelitian ini adalah sebagai berikut:

1. Membangun model VQA menggunakan citra medis dengan menerapkan teknik *transfer learning*.
2. Membandingkan performa model VQA pada arsitektur VGG19-LSTM dan BLIP, dengan *dataset* PathVQA dan VQA-RAD.
3. Mengimplementasikan model VQA terbaik untuk menjawab pertanyaan berdasarkan citra medis.

1.4 MANFAAT PENELITIAN

Manfaat yang diinginkan dari penelitian ini adalah untuk memberikan gambaran mengenai sistem VQA untuk citra medis dengan *deep learning*. Selanjutnya, memberikan pengetahuan hasil implementasi dengan arsitektur VGG19-LSTM dan BLIP dalam mengembangkan sistem VQA pada citra medis. Lalu yang terakhir, memberikan kontribusi terkait penelitian *medical visual question answering*.

BAB II

TINJAUAN KEPUSTAKAAN

2.1 CITRA MEDIS

Citra adalah gambaran dari suatu objek yang bisa kita lihat. Citra analog tidak bisa disimpan secara langsung dalam komputer. Oleh karena itu, kita perlu mengubah citra analog menjadi citra digital agar bisa diolah oleh komputer. Citra digital adalah citra yang bisa diolah menggunakan perangkat komputer. Alasan utama citra analog tidak bisa diolah oleh komputer adalah karena citra analog tidak memiliki konsep sampling dan kuantisasi. Konsep sampling adalah suatu metode yang mengubah citra analog menjadi *grid* berbentuk M baris dan N kolom, sehingga citra menjadi lebih tersegmentasi. Semakin besar nilai M dan N, semakin halus citra digital yang dihasilkan. Sedangkan konsep kuantisasi adalah suatu metode yang mengubah intensitas dari citra analog menjadi intensitas diskrit. Baik sampling maupun kuantisasi memegang peranan penting dalam mengambil potongan citra menjadi bentuk M baris dan N kolom (proses sampling) serta menentukan nilai intensitas yang terdapat pada setiap titik tersebut (proses kuantisasi). Hasil akhir dari kedua konsep ini adalah citra yang memiliki resolusi sesuai dengan yang kita inginkan (Andono et al., 2018).

Dalam ilmu pengolahan citra, beragam jenis citra ada, dan salah satunya adalah citra *Red Green Blue* (RGB), juga disebut sebagai citra *true color*. Citra ini memiliki matriks data berukuran $m \times n \times 3$, yang menggambarkan warna merah, hijau, dan biru pada setiap piksel. Rentang nilai untuk setiap warna adalah antara 0 (nilai minimum) hingga 255 (nilai maksimum) dalam monitor komputer. Pada bagian ini, skala 0 hingga 255 dipilih berdasarkan representasi delapan digit dalam bilangan biner yang digunakan oleh komputer. Dengan demikian, lebih dari 16 juta warna dapat dihasilkan secara total. Penentuan warna pada setiap piksel didasarkan pada tingkat kecerahan merah, hijau, dan biru (Ardiansyah, 2013).

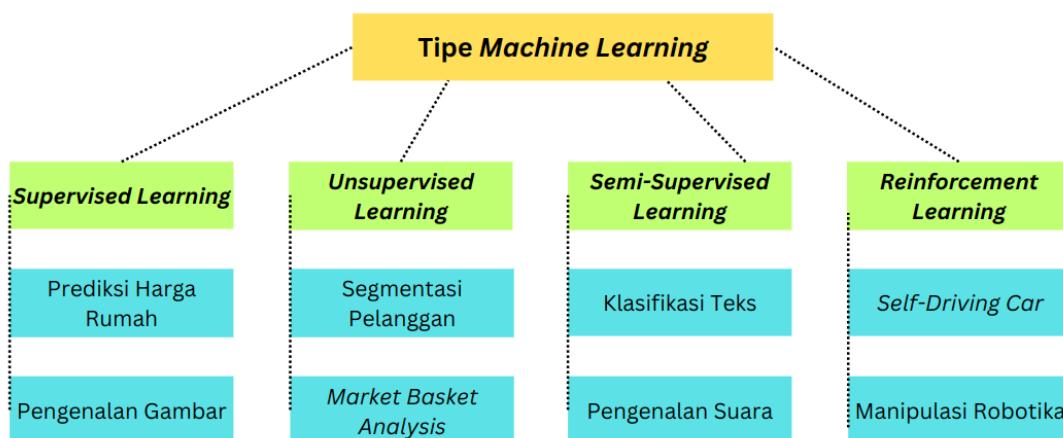
Di bidang kesehatan, pencitraan medis memegang peranan penting dalam berbagai bidang klinis seperti prosedur medis yang digunakan untuk deteksi dini, pemantauan, diagnosis, dan evaluasi pengobatan berbagai kondisi medis (Puttagunta and Ravi, 2021). Citra medis merujuk pada gambar dua dimensi yang menggambarkan struktur internal tubuh manusia, dimanfaatkan oleh profesional kesehatan untuk diagnosis penyakit. Pengolahan citra medis memiliki aplikasi luas, termasuk deteksi tumor atau kanker pada organ reproduksi wanita, identifikasi patologi pada organ-organ seperti paru-paru, hati, dan tulang, segmentasi struktur tulang dari jaringan otot, klasifikasi gigi, serta analisis gambar dari mikroskop. Berbagai teknologi digunakan dalam pencitraan medis ini, antara lain *Magnetic Resonance Imaging* (MRI), *X-Ray*,

Ultrasonography (USG), endoskopi, *Computed Tomography* (CT-Scan), dan *Nuclear Medicine* (Kusuma and Ellyana, 2018).

2.2 DEEP LEARNING

Deep learning adalah cabang ilmu *machine learning* yang berbasis pada *Artificial Neural Network* (ANN) sebagai pengembangannya. Metode dalam *machine learning* biasanya hanya mengandalkan alat seperti *Central Processing Unit* (CPU) dan *Random Access Memory* (RAM) dalam menentukan kecepatan komputasi. Sedangkan metode *deep learning* selain menggunakan CPU dan RAM, metode ini juga menggunakan *Graphics Processing Unit* (GPU) sehingga proses komputasi data yang besar dapat dilakukan dengan cepat (Ilahiyah and Nilogiri, 2018).

Deep learning memiliki pendekatan yang dapat dikategorikan seperti *supervised learning*, *unsupervised learning*, *semi-supervised learning*, dan *reinforcement learning* (Alom et al., 2019). Kategori pendekatan *deep learning* dapat dilihat pada Gambar 2.1.



Gambar 2.1. Kategori pendekatan *deep learning*

Pendekatan ini melibatkan berbagai teknik dan algoritma untuk menyelesaikan berbagai jenis masalah. Dalam mengimplementasikan model *deep learning*, beberapa *hyperparameter* yang sangat penting untuk diperhatikan adalah *learning rate*, *batch size*, jumlah *epochs*, dan arsitektur jaringan itu sendiri. *Learning rate* adalah ukuran yang menentukan seberapa besar langkah yang diambil model saat memperbarui bobot selama pelatihan. *Batch size* mengacu pada jumlah sampel data yang diproses sebelum memperbarui parameter model, sementara jumlah *epochs* menunjukkan berapa kali seluruh *dataset* akan dilalui selama pelatihan. Pemilihan *hyperparameter* sering kali dilakukan dengan fleksibilitas berdasarkan eksperimen dan kebutuhan spesifik dari proyek yang sedang dikerjakan. Tidak ada aturan tetap mengenai nilai-nilai spesifik yang harus dipilih untuk *hyperparameter*. Seorang peneliti dapat mencoba berbagai nilai

untuk menemukan kombinasi yang optimal. Metode seperti *grid search* menunjukkan bahwa nilai-nilai *hyperparameter* dapat dipilih secara bebas dalam rentang yang telah ditentukan (Bergstra and Bengio, 2012).

Grid search adalah metode pencarian *hyperparameter* yang melibatkan pencarian melalui ruang *hyperparameter* dengan mencoba setiap kombinasi yang mungkin dari nilai *hyperparameter* yang telah ditentukan sebelumnya. Dengan *grid search*, seorang peneliti membuat sebuah *grid* dari *hyperparameter* yang berbeda dan mengevaluasi model dengan setiap kombinasi yang ada di *grid* tersebut (Bergstra and Bengio, 2012). Pendekatan ini memastikan bahwa semua kombinasi yang mungkin diuji, meskipun bisa sangat memakan waktu dan sumber daya.

Pada bidang medis *deep learning* digunakan untuk memproses citra medis seperti X-rays, CT, scan MRI (*Magnetic Resonance Imaging*) dan lain-lain untuk mendiagnosa kondisi kesehatan (Kelleher, 2019).

2.2.1 Artificial Neural Network

Artificial Neural Network (ANN) atau jaringan saraf tiruan adalah jaringan saraf yang memproses informasi dengan cara yang mirip dengan otak manusia (Kristiyanti and Saputra, 2023). *Neural network* terdiri dari elemen pemrosesan sederhana yang disebut *node* yang saling terhubung, mirip dengan cara kerja neuron dalam otak manusia. Kemampuan untuk melakukan pemrosesan dalam jaringan ini disimpan dalam koneksi antara *node*, yang biasanya disebut sebagai *weight*. Nilai-nilai *weight* ini diperoleh melalui proses pembelajaran atau adaptasi yang berdasarkan pada pola data yang dipelajari oleh ANN (Gurney, 1997).

2.2.2 Fungsi Aktivasi

Fungsi aktivasi dalam konteks jaringan saraf dapat diibaratkan dengan cara tubuh manusia merespon rangsangan dari lingkungan. Ketika seseorang menerima rangsangan eksternal, tubuhnya secara otomatis meresponsnya. Sebagai contoh, ketika tangan kita digigit, tubuh kita akan merespons dengan menolak atau melepaskan gigitan tersebut. Respons tubuh ini akan semakin intens jika rangsangan yang diterima semakin kuat. Dalam konteks algoritma jaringan saraf, respons tubuh ini analoginya digantikan oleh nilai bobot dan tingkat aktivasi yang tinggi. Pada persamaan 2.1 menunjukkan persamaan dari jaringan saraf sebelum menggunakan fungsi aktivasi.

$$y = \sum_{i=1}^n x_i w_i + b \quad (2.1)$$

- y adalah nilai keluaran dari jaringan saraf.

- x_i adalah nilai *input* dari jaringan saraf.
- w_i adalah bobot dari setiap nilai *input*.
- b adalah bias dari jaringan saraf.

Ketika menerapkan fungsi aktivasi pada persamaan 2.1, maka persamaan akan berubah menjadi seperti yang ditunjukkan pada persamaan 2.2.

$$z = \text{Act}\left(\sum_{i=1}^n x_i w_i + b\right) \quad (2.2)$$

- z adalah nilai keluaran dari jaringan saraf setelah menggunakan fungsi aktivasi.
- Act adalah fungsi aktivasi.
- x_i adalah nilai *input* dari jaringan saraf.
- w_i adalah bobot dari setiap nilai *input*.
- b adalah bias dari jaringan saraf.

Persamaan 2.2 ini menunjukkan bahwa nilai keluaran z akan bergantung pada fungsi aktivasi terhadap nilai prediksi yang dihasilkan dari perkalian nilai *input* dan bobot (Kristiyanti and Saputra, 2023).

Fungsi aktivasi adalah komponen penting dalam jaringan saraf tiruan yang mengatur bagaimana nilai hasil penjumlahan terbobot dari *input* data diubah menjadi *output* yang dikeluarkan oleh neuron dalam jaringan saraf. Fungsi aktivasi ini digunakan untuk menentukan apakah suatu neuron akan meneruskan nilai kalkulasinya ke neuron berikutnya, berdasarkan suatu nilai ambang tertentu. Fungsi ini juga sering disebut sebagai fungsi transfer karena memiliki kemampuan untuk mengubah data yang dihasilkan dari hasil penjumlahan terbobot pada suatu lapisan, yang kemudian akan diteruskan ke lapisan selanjutnya. Fungsi aktivasi bisa berupa fungsi linear atau fungsi non-linear tergantung pada tugas yang ingin diselesaikan, dan fungsi aktivasi ini dapat digunakan dalam berbagai hal seperti pengenalan objek dan klasifikasi (Nwankpa et al., 2018).

Fungsi aktivasi perlu memiliki sifat diskriminatif, yang merupakan aspek yang penting karena memungkinkan penggunaan proses propagasi balik kesalahan dalam pelatihan jaringan. Salah satu fungsi aktivasi yang umum digunakan dalam konteks CNN adalah fungsi ReLU (*Rectified Linear Unit*). Fungsi ini merupakan fungsi aktivasi yang mengubah seluruh isi nilai *input* menjadi angka positif (Alzubaidi et al., 2021). Persamaan fungsi aktivasi ini dapat dilihat pada persamaan 2.3.

$$f(x)_{\text{ReLU}} = \max(0, x) \quad (2.3)$$

Salah satu fungsi aktivasi yang digunakan dalam berbagai model mutakhir seperti GPT-3, BERT, dan sebagian besar model Transformer lainnya adalah *Gaussian Error Linear Unit* (GELU). Fungsi ini merupakan fungsi aktivasi yang menimbang *input* berdasarkan persentilnya, daripada mengelompokkan *input* berdasarkan tandanya seperti ReLU. Oleh karena itu, GELU dapat dikatakan lebih mulus dibandingkan ReLU. Hal ini memungkinkan GELU untuk lebih mudah memperkirakan fungsi yang rumit daripada ReLU atau ELU (Hendrycks and Gimpel, 2016). Persamaan fungsi aktivasi ini dapat dilihat pada persamaan 2.4.

$$\text{GELU}(x) = xP(X \leq x) = x\Phi(x) = x \cdot \frac{1}{2}[1 + \text{erf}(x/\sqrt{2})] \quad (2.4)$$

- x adalah nilai *input* dari fungsi aktivasi.
- Φ adalah *Cumulative Distribution Function* (CDF) dari x .
- erf adalah fungsi kesalahan dari x .

Salah satu fungsi aktivasi yang dapat digunakan untuk mengklasifikasi lebih dari dua kelas adalah fungsi aktivasi *softmax*. Persamaan 2.5 menunjukkan persamaan dari fungsi aktivasi *softmax*.

$$f_j(Z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (2.5)$$

Pada persamaan 2.5, Notasi f_j menunjukkan hasil fungsi pada setiap elemen ke- j pada vektor keluaran kelas. Argumen z merupakan hipotesis yang diberikan oleh model pelatihan agar dapat diklasifikasi oleh fungsi *softmax* (Ilahiyyah and Nilogiri, 2018).

2.2.3 Fungsi loss

Fungsi *loss* adalah salah satu fungsi pada *Artificial Neural Network* (ANN) untuk melakukan perhitungan nilai *error* atau kesalahan dari hasil prediksi dari suatu *output* ANN (Zhang, 2016).

Fungsi *loss* yang menghukum kesalahan probabilitas *false negative* daripada *false positive* adalah *categorical cross entropy* (Ho and Wookey, 2019). Persamaan 2.6 menunjukkan persamaan dari *categorical cross-entropy*.

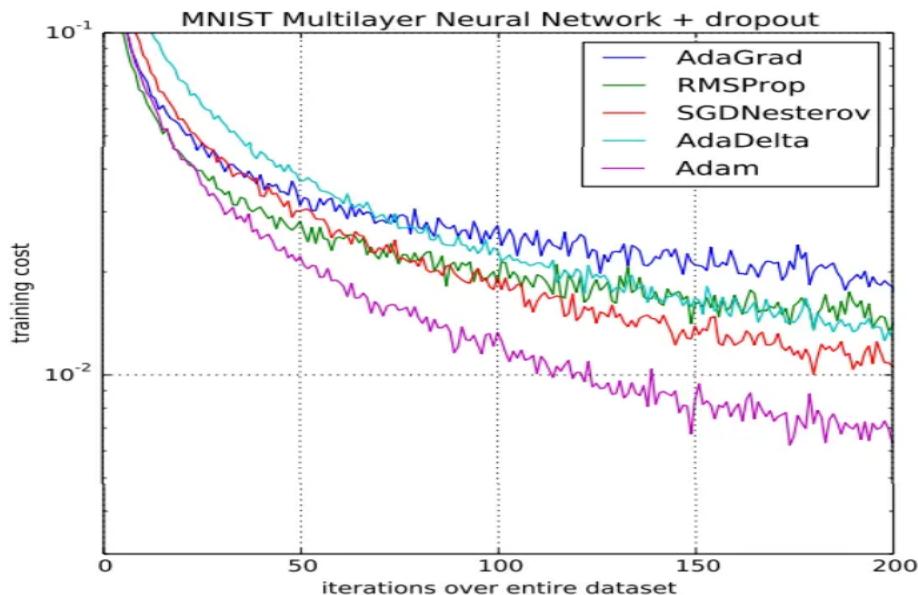
$$J_{cce} = -\frac{1}{M} \sum_{k=1}^K \sum_{m=1}^M y_m^k \log_{(\theta)}(x_m, k) \quad (2.6)$$

- M adalah jumlah sampel data yang digunakan untuk pelatihan.
- K adalah jumlah kelas yang ada pada data.
- y_m^k adalah nilai target dari sampel data ke- m pada kelas ke- k .
- x adalah *input* untuk contoh pelatihan ke- m .
- H_θ adalah bobot model *neural network* θ .

2.2.4 Fungsi Optimasi

Fungsi optimasi atau *optimization function* dapat diartikan sebagai suatu fungsi yang berperan sebagai *black box*, dimana fungsi ini menerima kesalahan sebagai input dan menghasilkan nilai bobot yang optimal untuk suatu model ANN (Li and Malik, 2017).

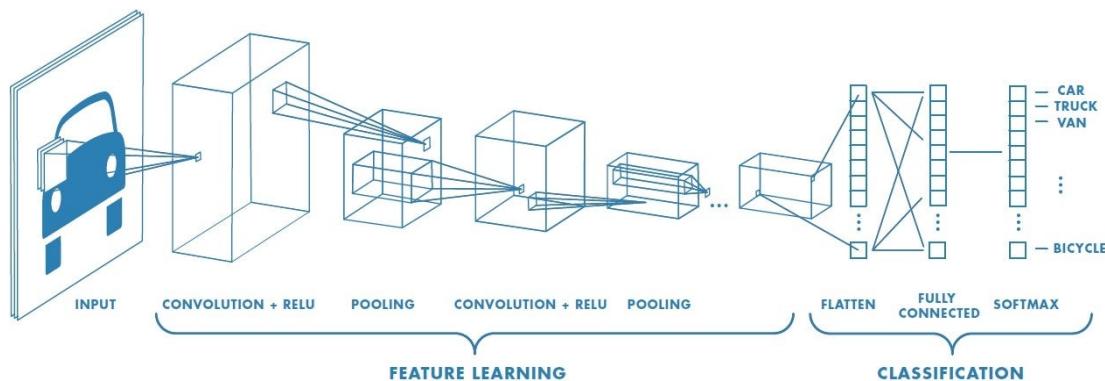
Salah satu dari fungsi optimasi yang dapat digunakan dalam pengembangan model *deep learning* adalah Adam (*Adaptive Moment Estimation*). Adam adalah teknik optimasi untuk *gradient descent*. Metode ini sangat efisien saat bekerja dengan masalah yang melibatkan banyak data atau parameter. Algoritma ini membutuhkan memori yang lebih sedikit dan efisien. Secara intuitif, Adam merupakan gabungan antara algoritma *stochastic gradient descent momentum* dan *RMSProp*. Secara eksperimen Adam adalah teknik optimasi terbaik dengan *training cost* yang rendah menurut (Kingma and Ba, 2014). Gambar 2.2 menunjukkan perbandingan teknik optimasi Adam dengan teknik optimasi lainnya.



Gambar 2.2. Perbandingan teknik optimasi Adam dengan teknik optimasi lainnya
(Kingma and Ba, 2014)

2.3 CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Network (CNN) adalah turunan dari algoritma *neural network* yang dikhususkan untuk memproses data yang berupa gambar. CNN adalah algoritma yang meniru proses pengolahan visual yang terjadi pada manusia. Seperti mata manusia yang berfungsi sebagai alat *input*, CNN menggunakan lapisan konvolusi yang terdiri dari miliaran neuron untuk memproses informasi visual dan menghasilkan prediksi terhadap objek yang diamati (Kristiyanti and Saputra, 2023). Algoritma CNN dirancang dengan neuron yang berfungsi mirip dengan cara area penglihatan pada otak manusia dan hewan bekerja, seperti yang dijelaskan oleh (Henningsen-Schomers and Pulvermüller, 2022). Arsitektur ini terdiri dari sejumlah lapisan, yang biasanya disebut sebagai blok-blok multi-bangunan dan dapat dilihat pada Gambar 2.3.



Gambar 2.3. Arsitektur CNN (Wardani and Leonardi, 2023)

Pada Gambar 2.3 dapat dilihat bahwa konvolusi merupakan langkah awal dalam pengolahan gambar yang digunakan untuk mengekstraksi fitur penting dari gambar *input*. Dalam konvolusi, hubungan antar piksel dipertahankan dengan cara mengoperasikan kotak kecil pada masukan untuk memahami fitur-fitur gambar. Konvolusi melibatkan operasi matematika linear yang mencakup perkalian antara matriks gambar dan filter (*kernel*) yang merupakan matriks bobot dua dimensi.

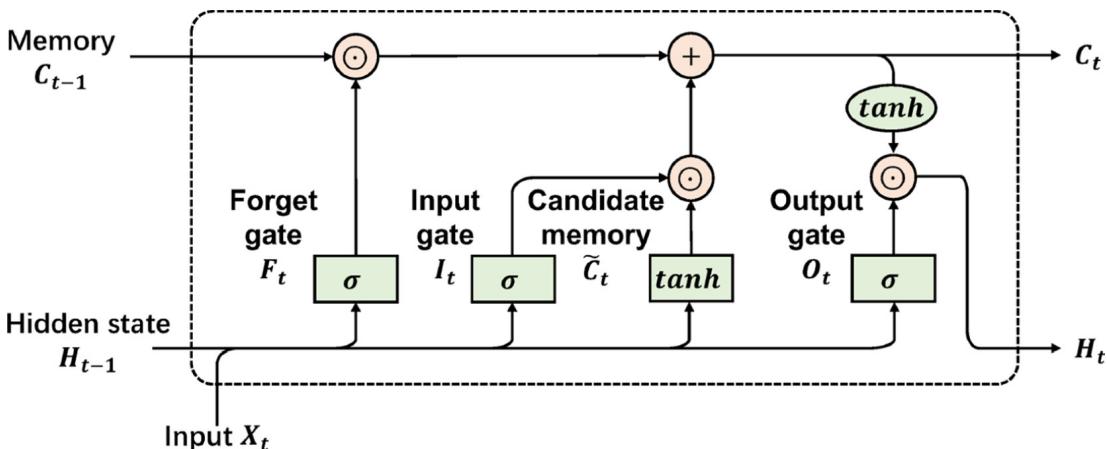
Fungsi dari lapisan *pooling* adalah untuk secara bertahap mengurangi ukuran representasi spasial gambar, sehingga mengurangi jumlah parameter yang dibutuhkan dalam jaringan. Lapisan *pooling* biasanya ditempatkan di antara lapisan-lapisan konvolusi. Lapisan ini beroperasi secara independen pada setiap peta fitur (Monedero et al., 2021).

CNN merupakan terobosan besar dalam pengenalan gambar. Mereka belajar langsung dari data gambar, menggunakan pola untuk mengklasifikasikan gambar dan menghilangkan kebutuhan ekstraksi fitur manual. Saat ini, CNN merupakan topik yang menarik dalam *machine learning*, dan telah menunjukkan kinerja yang sangat baik dalam klasifikasi (Khan et al., 2020).

2.4 LONG SHORT TERM MEMORY

Long Short Term Memory (LSTM) adalah tipe dari *Recurrent Neural Network* (RNN) yang diciptakan untuk menangani data yang bersifat *sequential* seperti data *time series*, *speech*, dan *text*. LSTM ini dikembangkan untuk mengatasi masalah *vanishing gradient* yang ada dalam RNN tradisional, yang membuatnya sulit bagi jaringan untuk mempelajari ketergantungan jangka panjang (Brownlee, 2017).

Menurut (Alom et al., 2019) LSTM adalah model jaringan saraf yang menggunakan *cell state* untuk menyimpan informasi dari *input* sebelumnya. *Cell state* memiliki tiga gerbang: *input gate* (i_t), *forget gate* (f_t), dan *output gate* (o_t) yang bisa dilihat pada Gambar 2.4.



Gambar 2.4. Arsitektur LSTM (Luo et al., 2023)

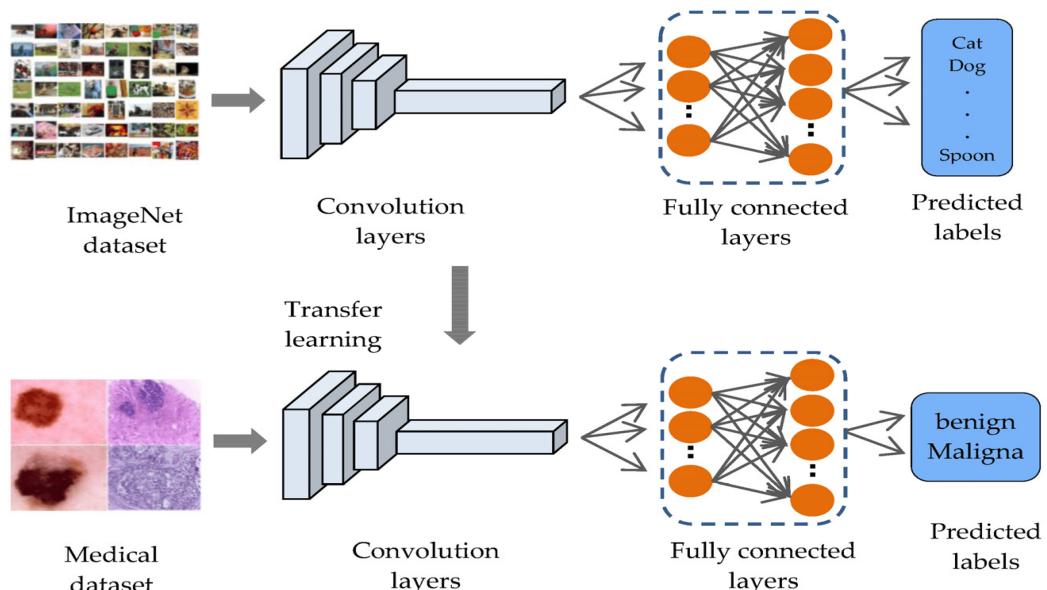
Input gate (i_t) digunakan untuk mengontrol pengaruh data yang masuk saat ini terhadap bobot unit tersebut, *forget gate* (f_t) bertujuan untuk mengendalikan pengaruh riwayat informasi pada bobot unit saat ini, *output gate* (o_t) bertujuan dalam mengendalikan eksport nilai keadaan unit memori (Huang et al., 2021).

2.5 TRANSFER LEARNING

Transfer learning digunakan untuk meningkatkan pembelajaran dari satu domain dengan mentransfer informasi dari domain terkait. Kita dapat mengambil pengetahuan dunia nyata non-teknis untuk memahami mengapa *transfer learning* memungkinkan. Pertimbangkan contoh dua orang yang ingin belajar bermain piano. Satu orang tidak memiliki pengalaman sebelumnya dalam bermain musik, dan orang lain memiliki pengetahuan musik yang luas melalui bermain gitar. Orang dengan latar belakang musik yang luas akan dapat belajar piano dengan lebih efisien dengan mentransfer pengetahuan musik yang sudah dipelajari sebelumnya ke tugas belajar bermain piano. Satu orang dapat mengambil informasi dari tugas yang sudah dipelajari sebelumnya dan

menggunakannya secara bermanfaat untuk belajar tugas yang terkait (Pan and Yang, 2009).

Transfer learning pada *Artificial Neural Network* (ANN) bisa dibayangkan seperti seseorang yang belajar menjadi lebih mudah, cepat, dan akurat dalam memahami tugas dan konsep baru jika mereka sudah memiliki pengalaman belajar yang serupa dengan konsep baru yang ingin dipelajari. Ini mirip dengan bagaimana seorang individu dapat lebih mudah memahami fisika setelah belajar matematika atau bagaimana seseorang dapat lebih lancar mengendarai truk setelah menguasai kemampuan mengemudi mobil. Pada dasarnya *transfer learning* terjadi ketika pemahaman tentang suatu konteks dipengaruhi oleh pemahaman sebelumnya tentang konteks yang mirip (Cireşan et al., 2012). Intinya adalah bahwa *transfer learning* memungkinkan kita untuk menggunakan pengetahuan yang sudah ada untuk memecahkan masalah baru seperti yang ditunjukkan pada Gambar 2.5.

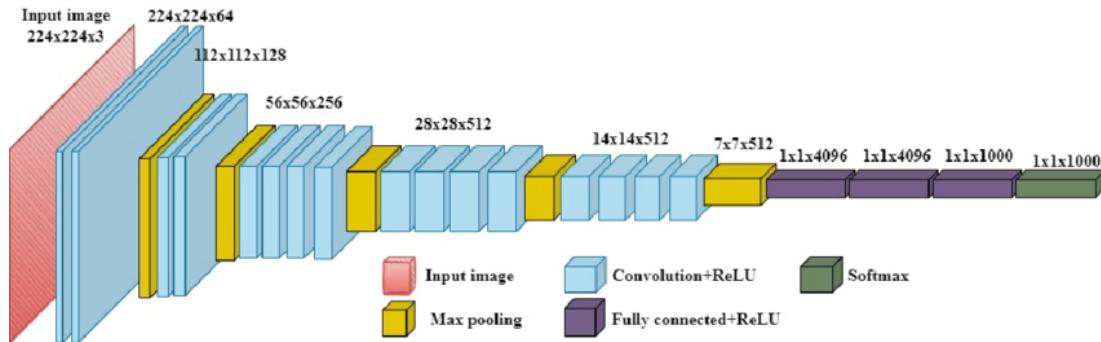


Gambar 2.5. Konsep penggunaan *transfer learning* (Mukhlif et al., 2023)

Model dasar yang digunakan dalam *transfer learning* adalah *pre-trained* model, dimana bobot di seluruh jaringan saraf tiruan sudah disesuaikan untuk data tertentu. Ini memungkinkan model untuk memiliki pemahaman yang lebih mendalam tentang fitur dasar dan fitur tingkat tinggi yang dapat mempercepat proses pelatihan. Secara konsisten, model jaringan saraf yang telah dilatih sebelumnya memberikan hasil prediksi yang lebih tepat dibandingkan dengan jaringan saraf yang dimulai dengan bobot-bobot yang diinisiasi secara acak dalam konteks masalah yang melibatkan data yang sudah memiliki label (Cireşan et al., 2012).

VGG19 merupakan salah satu model *pre-trained* yang digunakan dalam

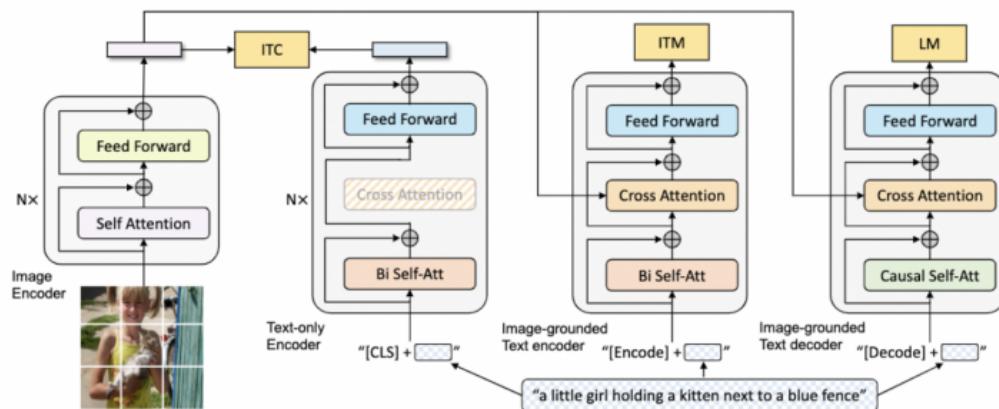
penelitian ini. VGG19 adalah model CNN yang memiliki 19 lapisan. Model ini dilatih pada *dataset* ImageNet yang memiliki 1000 kelas dan 1,2 juta gambar. Model ini memiliki 16 lapisan konvolusi dan 3 lapisan *fully connected*. Model ini memiliki 138 juta parameter dan 20 miliar operasi (Simonyan and Zisserman, 2014). Arsitektur VGG19 dapat dilihat pada Gambar 2.6.



Gambar 2.6. Arsitektur VGG19 (Nguyen et al., 2022)

Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation (BLIP) adalah sebuah kerangka dari *Vision Language Pre-Training* (VLP). BLIP mampu melakukan berbagai tugas *multimodal* seperti *Visual Question Answering* (VQA), *image captioning*, dan lain-lain. Dalam hal ini, BLIP mengusulkan sebuah model *multimodal mixture of encoder-decoder* yang dapat beroperasi secara fleksibel sebagai *encoder* dan *decoder* untuk berbagai tugas penglihatan bahasa. BLIP mampu memajukan *pre-training* terunifikasi untuk transfer pembelajaran lintas tugas *vision-language*. Berikut ini dapat dilihat arsitektur pada BLIP pada Gambar 2.7.

A unified model for vision-language understanding and generation



Gambar 2.7. Arsitektur BLIP (Li et al., 2022)

BLIP memiliki tiga fungsionalitas yang dapat dilihat sebagai berikut:

1. *Unimodal encoder* yang memisahkan proses *encode* gambar dan teks. *Image encoder* didasarkan pada *Vision Transformer* (ViT), sementara *encoder* teks mirip dengan *Bidirectional Encoder Representation from Transformer* (BERT). Disini, terdapat token khusus [CLS] yang ditambahkan di awal *input* teks untuk merangkum kalimat.
2. *Image-grounded text encoder* yang bertujuan memasukkan informasi visual ke dalam *encoder* teks. Pada bagian ini dilakukan dengan menyisipkan lapisan *cross-attention* antara lapisan *self-attention* dan jaringan *feed-forward* untuk setiap blok transformer dari *encoder* teks. Pada bagian ini terdapat token khusus yaitu [Encode] yang ditambahkan ke teks, dan *embedding output* dari [Encode] ini digunakan sebagai representasi multimodal dari pasangan gambar dan teks.
3. *Image grounded text decoder* memodifikasi *encoder* teks dengan menggantikan lapisan *bidirectional self-attention* dengan lapisan *causal self-attention*. Di bagian ini, token khusus [Decode] digunakan untuk menandai awal dari sebuah urutan.

Pada gambar 2.7 dijelaskan tiga tujuan dari arsitektur BLIP. Penjelasan tiga tujuan BLIP dapat dilihat sebagai berikut:

1. *Image-Text Contrastive Loss* (ITC) bertujuan mengaktifkan *unimodal encoder*. ITC mendorong keselarasan dalam ruang fitur transformer visual dan transformer teks dengan memastikan pasangan gambar dan teks yang positif memiliki representasi yang mirip dibandingkan dengan pasangan negatif.
2. *Image-Text Matching Loss* (ITM) bertujuan dalam mengaktifkan *encoder* teks berdasarkan gambar. Bagian ini merupakan tugas klasifikasi biner dimana model memprediksi apakah pasangan gambar dan teks cocok atau tidak cocok berdasarkan fitur *multimodal* mereka.
3. *Language Modelling Loss* (LM) bertujuan mengaktifkan *decoder* teks berdasarkan gambar. Ini berfokus menghasilkan deskripsi teks yang dikondisikan pada gambar yang diberikan.

Intinya, BLIP menggunakan model serbaguna yang dapat menyatukan informasi visual ke dalam pemrosesan teks. Selama tahap pra-pelatihan, BLIP secara simultan mengoptimalkan tujuan keselarasan representasi visual dan teks, pencocokan pasangan gambar dan teks, serta pembangkitan deskripsi dalam bahasa dan gambar (Li et al., 2022).

2.6 MATRIKS EVALUASI

Matriks evaluasi yang digunakan dalam menilai kinerja model *medical VQA* dapat dikategorikan menjadi dua jenis, yaitu *classification-based metrics* dan *language-based metrics*. Pada umumnya metrik yang digunakan pada kasus klasifikasi, seperti akurasi dan *F1-score*. Metrik ini memperlakukan jawaban sebagai hasil dari klasifikasi dan menghitung pencocokan yang tepat untuk akurasi, presisi, *recall* dan lain-lain. Sedangkan metrik yang digunakan pada kasus *language-based* bertujuan untuk mengevaluasi tugas seperti penerjemah, *image captioning*, VQA, dan lain-lain. *Dataset* seperti VQA-Med-2018, VQA-Med-2019, PathVQA, VQA-Med-2020, dan VQA-Med-2021 menggunakan metrik *language-based* seperti *BLEU*, untuk mengevaluasi kinerjanya (Lin et al., 2023).

Confusion matrix adalah suatu alat pengukuran kinerja yang digunakan dalam konteks masalah klasifikasi pada pembelajaran mesin. Matriks ini terdiri dari tabel memberikan gambaran visual dan ringkasan mengenai kinerja algoritma klasifikasi. Tiap baris dalam matriks mencerminkan contoh dalam kelas aktual, sementara setiap kolom mencerminkan contoh dalam kelas yang diprediksi. *Confusion matrix* sangat bermanfaat untuk menilai efektivitas suatu model, terutama ketika terdapat ketidakseimbangan jumlah observasi antar kelas atau ketika menghadapi lebih dari dua kelas dalam suatu *dataset*. *Confusion matrix* merefleksikan nilai *True Positive* (TP) yang menunjukkan klasifikasi yang tepat pada kelas yang relevan, nilai *False Positive* (FP) yang mencerminkan klasifikasi di kelas yang seharusnya tidak relevan, nilai *False Negative* (FN) yang menunjukkan klasifikasi di kelas yang salah ketika seharusnya relevan, dan nilai *True Negative* (TN) yang mencerminkan klasifikasi yang benar di kelas yang tidak relevan. Gambar *confusion matrix* dapat dilihat pada Gambar 2.8 (Kulkarni et al., 2020).

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Gambar 2.8. *Confusion matrix* (Kulkarni et al., 2020)

2.6.1 Akurasi

Pada umumnya akurasi adalah metrik yang mengukur perbandingan antara jumlah prediksi yang benar dengan jumlah total kasus yang dievaluasi (Hossin and Sulaiman, 2015). Pada persamaan 2.7 menunjukkan bahwa rumus dari akurasi adalah jumlah prediksi yang benar dibagi dengan total prediksi yang dilakukan.

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.7)$$

- TP (*True Positive*) adalah jumlah prediksi yang benar positif.
- TN (*True Negative*) adalah jumlah prediksi yang benar negatif.
- FP (*False Positive*) adalah jumlah prediksi yang salah positif.
- FN (*False Negative*) adalah jumlah prediksi yang salah negatif.

2.6.2 Bilingual Evaluation Understudy

Bilingual Evaluation Understudy (BLEU) adalah metrik yang digunakan untuk mengukur kesamaan pada frase (n-grams) antara dua kalimat. BLEU adalah metrik original untuk penerjemah mesin dan juga bisa digunakan untuk tugas seperti pembuatan laporan medis (Li et al., 2021).

BLEU adalah metrik yang digunakan untuk mengevaluasi kualitas sistem terjemahan mesin (Papineni et al., 2002). Ini mengukur kemiripan antara terjemahan yang dihasilkan oleh mesin dan satu atau lebih referensi terjemahan menggunakan metrik presisi yang dimodifikasi dan hukuman singkat yang dimodifikasi dan *brevity penalty* yang dimodifikasi.

$$\text{BLEU} = \text{BP} \times \exp \left(\sum_{n=1}^N w_n \log p_n \right) \quad (2.8)$$

- BP adalah *brevity penalty*.
- N adalah jumlah maksimal n-gram.
- w_n adalah bobot untuk n-gram.
- p_n adalah presisi n-gram.

Berdasarkan persamaan 2.8 *Brevity Penalty* (BP) menghukum jawaban singkat, w_n adalah bobot antara 0 dan 1 untuk $\log p_n$ dan $\sum_{n=1}^N w_n = 1$, p_n adalah nilai rata-rata geometris dari presisi n-gram yang dimodifikasi, dan N adalah panjang maksimum n-gram.

BLEU memiliki beberapa variasi, seperti BLEU-1, BLEU-2, BLEU-3, dan seterusnya. Variasi ini menunjukkan jumlah n-gram yang digunakan dalam perhitungan BLEU. Semakin tinggi nilai BLEU, semakin baik kualitas terjemahan mesin tersebut. Nilai BLEU berkisar dari 0 hingga 100, di mana 100 menunjukkan terjemahan yang sempurna. Berikut adalah penjelasan mengenai berbagai variasi BLEU:

1. BLEU-1 adalah variasi dasar dari metrik BLEU yang hanya mempertimbangkan *unigram* (kata tunggal). Perhitungan BLEU-1 mengukur seberapa baik kata-kata dalam hasil terjemahan sesuai dengan kata-kata dalam kalimat referensi tanpa mempertimbangkan urutan kata.
2. BLEU-2 memperluas perhitungan dengan menggunakan *bigram* (pasangan kata). Dalam BLEU-2, selain mempertimbangkan kesesuaian *unigram*, juga diperiksa kesesuaian pasangan kata dalam hasil terjemahan dan kalimat referensi. Hal ini memberikan evaluasi yang lebih baik terhadap struktur kalimat terjemahan.
3. BLEU-3 melangkah lebih jauh dengan menggunakan *trigram* (tiga kata berturut-turut). Disini, evaluasi BLEU-3 memperhitungkan kesesuaian *trigram* antara hasil terjemahan dan kalimat referensi. Semakin panjang n-gram yang digunakan, semakin ketat evaluasi terhadap kualitas terjemahan, karena memperhatikan konteks yang lebih luas dalam kalimat.

Kesimpulannya, variasi-variasi ini menunjukkan bahwa dengan meningkatnya nilai "n" dalam n-gram, evaluasi terhadap terjemahan mesin menjadi lebih komprehensif dan memperhitungkan konteks yang lebih luas dalam kalimat (Papineni et al., 2002).

Deskripsi BLEU berbeda tergantung pada rentang nilai yang ada. Detail deskripsi untuk masing-masing rentang nilai BLEU dapat ditemukan di Tabel 2.1 (Lavie, 2010).

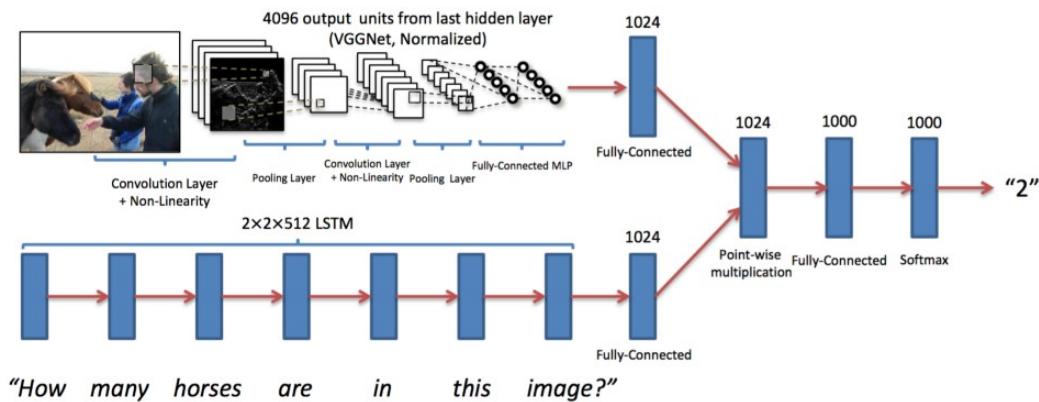
Tabel 2.1. Deskripsi dari masing-masing nilai BLEU

Nilai BLEU	Deskripsi
< 10	Hampir tidak dapat dipahami
10 - 19	Sulit untuk mendapatkan intisari kalimat
20 - 29	Inti kalimat jelas, tetapi memiliki banyak kesalahan pada tata bahasa
30 - 39	Dapat dimengerti sebagai penerjemah kalimat yang baik
40 - 49	Terjemahan kalimat berkualitas tinggi
50 - 59	Terjemahan kalimat sangat berkualitas dan memadai
> 60	Kemampuan menerjemah sering lebih bagus dari manusia

2.7 VISUAL QUESTION ANSWERING

Visual Question Answering (VQA) adalah suatu tugas tentang bagaimana menjawab pertanyaan bebas pada sebuah gambar. Karena membutuhkan pemahaman bahasa yang mendalam tentang pertanyaan dan kemampuan untuk mengaitkannya dengan berbagai objek yang ada dalam gambar. Ini adalah tugas yang ambisius dan memerlukan teknik dari baik *computer vision* dan *natural language processing* (Hildebrandt et al., 2020).

Salah satu contoh percobaan yang dilakukan oleh (Antol et al., 2015), mereka menerapkan *baseline Vanilla VQA* model yang dimana sebagai tolak ukur untuk metode *deep learning*. Model ini menggunakan CNN untuk ekstraksi fitur dan jaringan LSTM atau *recurrent network* untuk pemrosesan bahasa. Fitur-fitur ini digabungkan menggunakan suatu operasi untuk membentuk satu fitur bersama, yang digunakan untuk mengklasifikasikan salah satu jawaban seperti yang ditunjukkan dalam Gambar 2.9.



Gambar 2.9. Vanilla VQA network model (Srivastava et al., 2021)

VQA telah menarik minat besar dan mengalami perkembangan oleh para peneliti dan ilmuwan dari seluruh dunia. Tren terbaru yang diamati adalah dalam pengembangan *dataset* yang terlihat semakin mirip dengan dunia nyata dengan menggabungkan pertanyaan dan jawaban seputar kehidupan nyata. Tren terkini juga terlihat dalam pengembangan model *deep learning* yang lebih canggih dengan lebih baik memanfaatkan petunjuk visual dan petunjuk teks melalui berbagai cara. Kinerja model terbaik saat ini masih tertinggal dan hanya sekitar 60-70% saja. Oleh karena itu, masih merupakan masalah yang terbuka untuk mengembangkan model *deep learning* yang baik serta *dataset* yang lebih menantang untuk *visual question answering* (Srivastava et al., 2021).

Dalam mengembangkan sistem VQA diperlukan dua cabang ilmu seperti yang sudah dijelaskan yaitu *computer vision* dan *natural language processing* dikarenakan melibatkan gambar dan pertanyaan teks yang terkait (Teney et al., 2017).

2.7.1 *Computer Vision*

Computer Vision (CV) bisa diartikan sebagai sekelompok teknik yang digunakan untuk mengumpulkan, memproses, menganalisis, dan memahami data yang kompleks dan berdimensi tinggi dari lingkungan sekitar kita. Tujuannya adalah untuk menjalani eksplorasi ilmiah dan teknis yang melibatkan pemahaman dan interpretasi data visual. Secara sederhana, CV memungkinkan komputer atau sistem komputasi untuk melihat, menginterpretasi, dan merespons informasi visual dari dunia nyata. Teknologi ini memiliki berbagai aplikasi, termasuk pengenalan objek, deteksi pola, dan pengolahan citra, serta mendukung berbagai bidang seperti kecerdasan buatan, penglihatan mesin, dan analisis data visual (Jähne et al., 1999).

2.7.2 *Natural Language Processing*

Natural Language Processing (NLP) adalah cabang ilmu dari pembelajaran mesin yang berfokus pada teks. Metode ini memungkinkan suatu komputer untuk menganalisis, memahami, dan menghasilkan makna dari bahasa manusia dengan cerdas dan bermanfaat. Dengan memanfaatkan NLP, pengembang dapat mengorganisir dan menyusun pengetahuan untuk menjalankan tugas-tugas seperti rangkuman otomatis, terjemahan, pengenalan entitas bernama, ekstraksi hubungan, analisis sentimen, pengenalan ucapan, dan segmentasi topik (Agarwal and Saxena, 2019).

Dalam NLP *word embedding* merupakan representasi dari suatu kata. *Embedding* digunakan dalam analisis teks. Biasanya, representasi tersebut berupa vektor nilai rill yang mengkodekan makna kata sedemikian rupa sehingga kata-kata yang lebih dekat dalam ruang vektor diharapkan memiliki makna yang mirip (Teller, 2000). *Global Vectors for Word Representation* (GloVe) adalah metode untuk membuat representasi kata. GloVe adalah algoritma *unsupervised learning* yang memperoleh representasi vektor untuk kata-kata dengan melatih pada statistik kumulatif keterjadian bersama global kata-kata dari suatu korpus. Representasi yang dihasilkan menunjukkan struktur sublinear menarik dari ruang vektor kata, memungkinkan kata-kata dengan makna serupa berada lebih dekat dalam ruang vektor. Dalam hal ini, GloVe digunakan untuk membangun fitur *embedding* kata semantik dan telah digunakan dalam berbagai tugas dalam bidang NLP (Pennington et al., 2014).

2.8 MEDICAL VISUAL QUESTION ANSWERING DATASET

Medical visual question answering dataset adalah *dataset* yang digunakan untuk melakukan penelitian terkait tugas yang berkaitan dengan *medical visual question answering* (Lin et al., 2023). *Dataset* ini berisi gambar medis dan pertanyaan yang terkait dengan gambar tersebut. Pada penelitian ini akan digunakan *dataset* PathVQA dan VQA-RAD. Berikut adalah penjelasan dari kedua *dataset* tersebut.

2.8.1 PathVQA Dataset

Dataset PathVQA berisi 32.799 pertanyaan dan jawaban dari 1.670 gambar patologi yang dikumpulkan dari dua buku patologi, yaitu '*Textbook of Pathology*' dan '*Basic Pathology*'. Selain itu, terdapat 3.328 gambar lainnya yang dikumpulkan dari perpustakaan digital PEIR. Dengan demikian, total gambar pada *dataset* ini mencapai 4.998. Rata-rata setiap satu gambar memiliki 6,6 pertanyaan. Jumlah pertanyaan maksimal dan minimal adalah 14 dan 1 berturut-turut. Rata-rata jumlah kata per pertanyaan dan perjawaban adalah 9,5 dan 2,5 berturut-turut. Pada *dataset* ini terdapat 7 kategori pertanyaan, yaitu: *what, where, when, whose, how, how much/how many*, dan *yes/no*. Pertanyaan dari enam kategori pertama bersifat *open-ended*, sedangkan pertanyaan dari kategori terakhir bersifat *close-ended* '*yes/no*'. Jumlah jawaban '*yes*' dan '*no*' adalah 8.145 dan 8.189 berturut-turut. Pertanyaan pada *dataset* ini mencakup banyak aspek konten visual, termasuk warna, lokasi, penampilan, bentuk, dan lain-lain. Keragaman klinis ini menimbulkan tantangan yang besar bagi model AI dalam menjawab permasalahan pada gambar patologi (Xuehai et al., 2020).

2.8.2 VQA-RAD Dataset

Dataset VQA-RAD adalah *dataset* yang dibuat secara manual untuk penelitian terkait *Medical Visual Question Answering* (VQA). *Dataset* ini berisi 3.515 pasangan pertanyaan dan jawaban yang dihasilkan oleh para klinisi, serta 315 gambar radiologi yang terbagi merata antara kepala, dada, dan perut. Dengan rata-rata, setiap gambar memiliki 10 pertanyaan terkait. Setiap gambar ini terkait dengan beberapa pertanyaan, yang dikelompokkan ke dalam 11 kategori: kelainan, atribut, modalitas, sistem organ, warna, penghitungan, keberadaan objek/kondisi, ukuran, bidang, penalaran posisional, dan lain-lain. Setengah dari pertanyaan pada *dataset* ini bersifat *close-ended* (*yes/no*), sedangkan setengahnya lagi bersifat *open-ended*. *Dataset* ini merupakan kontribusi berharga dalam pengembangan model VQA di bidang radiologi, dengan pertanyaan-pertanyaan yang disusun oleh para ahli klinis, mencakup berbagai aspek interpretasi gambar medis (Lau et al., 2018).

2.9 PENELITIAN TERKAIT

Penelitian yang dilakukan oleh (Xuehai et al., 2020) berhasil mengatasi tantangan khusus dalam analisis citra patologi dengan menggabungkan metode pemrosesan gambar dan pemrosesan bahasa alami. Mereka melakukan eksperimen dengan tiga metode berbeda. Tiga metode yang digunakan melibatkan integrasi *Gated Recurrent Unit* (GRU) dan Faster R-CNN dengan melibatkan *Bilinear Attention Network*, *Convolutional Neural Network* (CNN) dan *Long Short Term Memory* (LSTM) dengan *compact bilinear pooling*, serta *Stacked Attention Network* (SAN) yang menggabungkan CNN dan LSTM. Pengkodean gambar menggunakan Faster R-CNN dan ResNet-152. Berdasarkan ketiga eksperimen ini menunjukkan bahwa metode pertama mencapai akurasi tertinggi untuk pertanyaan *close-ended* adalah 68,2% dan mendapatkan nilai BLEU-1, BLEU-2, BLEU-3 sebesar 32,4, 22,8, 17,4 berturut-turut untuk pertanyaan *open-ended*. Metode kedua mencapai akurasi sebesar 57,6% untuk pertanyaan *close-ended* dan mendapatkan BLEU-1, BLEU-2, BLEU-3, sebesar 13,3, 9,5, 6,8 berturut-turut untuk pertanyaan *open-ended*. Metode ketiga mendapatkan akurasi sebesar 59,4% untuk pertanyaan *close-ended* dan mendapatkan BLEU-1, BLEU-2, BLEU-3 sebesar 19,2, 17,9, 15,8 berturut-turut untuk pertanyaan *open-ended*. Lalu, hasil yang didapat ketika melakukan pengkodean gambar dengan Faster R-CNN mendapatkan akurasi 62,0% untuk pertanyaan *close-ended* dan mendapatkan BLEU-1, BLEU-2, BLEU-3 sebesar 24,7, 19,1, 16,5 berturut-turut untuk pertanyaan *open-ended*. Sedangkan ketika menggunakan ResNet-152, mereka mendapatkan akurasi 60,1% untuk pertanyaan *close-ended* dan BLEU-1, BLEU-2, BLEU-3 sebesar 19,9, 18,0, 16,0 berturut-turut untuk pertanyaan *open-ended* (Xuehai et al., 2020).

Penelitian lain juga dilakukan untuk meningkatkan diagnosis dengan menjawab pertanyaan klinis yang disajikan dengan gambar medis menggunakan model *Contrastive Language Image Pre Training* (CLIP) yang mengintegrasikan pembelajaran *multimodal embedding* melalui pelatihan bersama *image encoder* dan *text encoder* untuk memaksimalkan kemiripan pasangan gambar-teks yang sesuai. Metodenya melibatkan penggunaan *Vision Transformer* (ViT) untuk *image encoder*, pertanyaan dimasukkan ke dalam transformer *encoder* teks, dan penggabungan representasi visual dan teks dalam *decoder multimodal* untuk menghasilkan jawaban otomatis. Uji coba dilakukan pada dua dataset VQA, PathVQA dan VQA-RAD, dengan konfigurasi menggunakan *multi-head attention* sebesar 512, tingkat *dropout* pada *fully connected layers* 0,1, dan dua blok transformer pada *encoder* dan *decoder*. Fungsi optimasi yang digunakan adalah Adam dengan *learning rate* 0,001, *batch size* 50, dan 50 *epoch* pelatihan, dengan citra diacak dan pengolahan teks menggunakan *bert-base-uncased*. Hasilnya menunjukkan akurasi yang signifikan, dengan akurasi

tertinggi mencapai 84,63% untuk pertanyaan *close-ended* dan akurasi, BLEU-1, BLEU-2, BLEU-3 adalah sebesar 58,29% 61,78, 61,16, 59,28 secara berturut-turut untuk pertanyaan *open-ended* dalam *dataset* PathVQA. Pada *dataset* VQA-RAD akurasi yang didapatkan adalah 82,47% untuk pertanyaan *close-ended* dan akurasi, BLEU-1, BLEU-2, BLEU-3 adalah sebesar 71,49% 71,03, 70,81, 67,01 untuk pertanyaan *open-ended* (Bazi et al., 2023).

BAB III

METODOLOGI PENELITIAN

3.1 WAKTU DAN LOKASI PENELITIAN

Penelitian ini akan bertempat di Ruang Lab Sistem Informasi dan *Database*. Waktu yang dibutuhkan agar penelitian ini dapat diimplementasikan adalah 4 bulan terhitung dari bulan Februari 2024 hingga Mei 2024.

3.2 JADWAL PELAKSANAAN

Untuk lebih memahami alur waktu dari penelitian ini, penulis telah menyusun sebuah jadwal yang rinci yang bisa dilihat pada Tabel 3.1.

Tabel 3.1. Jadwal pelaksanaan penelitian

No	Kegiatan	Bulan Ke -																							
		Januari				Februari				Maret				April				Mei				Juni			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Studi literatur																								
2	Pengumpulan data																								
3	Pemrosesan data																								
4	Membangun model																								
5	Melatih model																								
6	Analisis hasil																								
7	Penyusunan laporan akhir																								

3.3 ALAT DAN BAHAN

Alat dan Bahan yang akan digunakan pada penelitian ini terdiri dari beberapa perangkat keras (*hardware*) dan perangkat lunak. Lalu, data yang digunakan adalah data dari *dataset* PathVQA, dan VQA-RAD.

3.3.1 Perangkat Keras

- Laptop Lenovo Yoga C740 dengan RAM 16GB DDR4, Intel® Core™ i7-10710U. 1.10 - 4.70 GHz, *Solid State Drive* (SSD) 1TB.
- Server spesifikasi processor Intel Xeon Gold 5218, CPU 2.30GHz 64 inti, RAM 128GB, VGA 4 x NVIDIA GeForce RTX 2080 Ti GPU VRAM 12GB, dan memory 8TB.

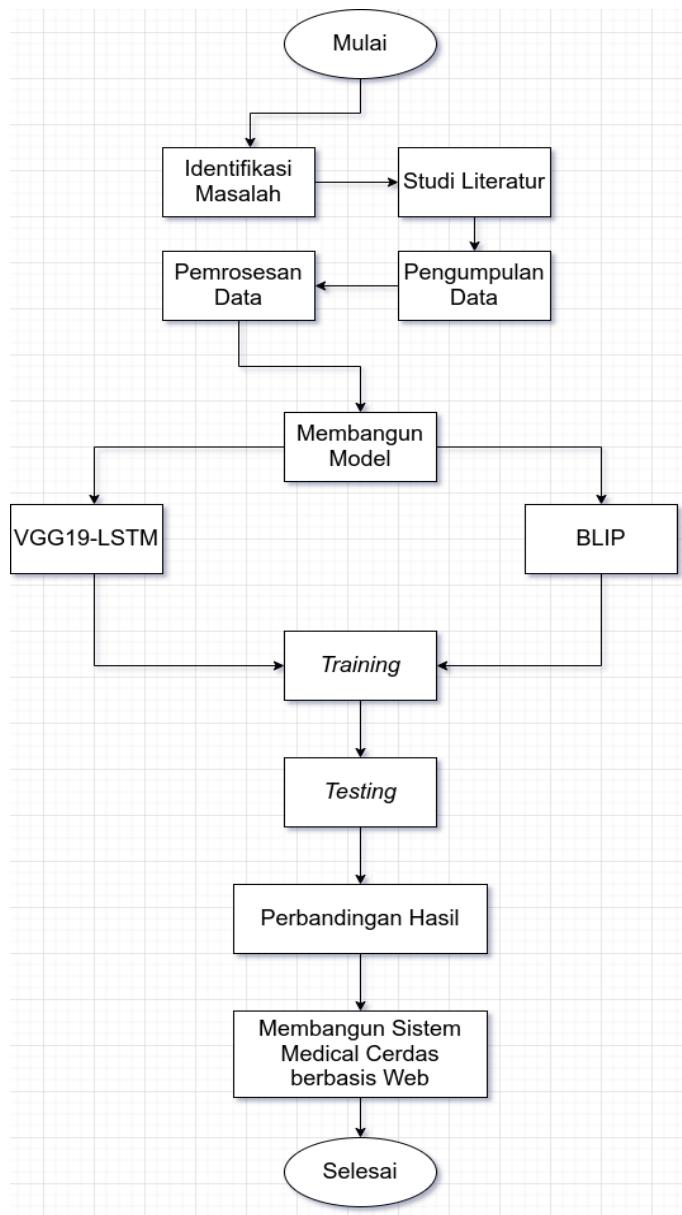
3.3.2 Perangkat Lunak

- Linux Debian Ubuntu versi 22.04 LTS
- Visual Studio Code versi 1.71.0

- c. Python versi 3.8.17
- d. PyTorch versi 2.0.1
- e. Tensorflow versi 2.11.0
- f. Transformers versi 4.31.0
- g. Huggingface Datasets versi 2.18.0
- h. FastAPI versi 0.104.1
- i. nlpaug versi 1.1.11

3.4 METODE PENELITIAN

Penelitian ini akan mengikuti beberapa tahapan yang dirancang secara sistematis untuk mencapai tujuan yang telah ditetapkan. Setiap tahap akan dilaksanakan dengan seksama untuk memastikan hasil yang akurat dan bermanfaat. Skema dari alur tahapan penelitian yang diusulkan dapat dilihat secara rinci pada Gambar 3.1. Pada Gambar 3.1 terdapat beberapa tahapan yang akan dilakukan dalam penelitian ini, yaitu identifikasi masalah, studi literatur, pengumpulan data, pemrosesan data, pembangunan model, pelatihan model, evaluasi model, dan pembangunan sistem medis cerdas berbasis web.



Gambar 3.1. Diagram alir penelitian

3.4.1 Identifikasi Masalah

Identifikasi masalah adalah proses awal dalam mengidentifikasi masalah yang akan diselesaikan dalam penelitian ini. Identifikasi masalah meliputi latar belakang masalah, rumusan masalah, tujuan penelitian serta manfaat yang dapat dihasilkan dari penelitian ini.

3.4.2 Studi Literatur

Studi literatur dilakukan untuk memperoleh informasi mengenai penelitian yang sudah dilakukan sebelumnya. Dalam tahapan ini terdiri dari banyak kegiatan yaitu

membaca referensi yang berkaitan dengan penelitian ini, membaca jurnal atau buku untuk menambah wawasan, sehingga dapat memperoleh informasi yang dibutuhkan untuk penelitian ini.

3.4.3 Pengumpulan Data

Penelitian tugas akhir ini menggunakan data dari *dataset* PathVQA dan VQA-RAD. Pengumpulan *dataset* dilakukan dengan mengunduh data dari platform Hugging Face. Sampel kedua data ini bisa dilihat pada Gambar 3.2 dan Gambar 3.3.

image image · width (px)	question string · lengths	answer string · lengths
285-426 2.7%	58-83 10.3%	23-45 8.3%
	where are liver stem cells (oval cells) located?	in the canals of hering
	what are stained here with an immunohistochemical stain for cytokeratin...	bile duct cells and canals of hering
	what do the areas of white chalky deposits represent?	foci of fat necrosis

Gambar 3.2. Sampel data PathVQA

image image · width (px)	question string · lengths	answer string · lengths
463-670 27.9%	26-37 35.2%	1-13 78.5%
	are regions of the brain infarcted?	yes
	are the lungs normal appearing?	no
	which organ system is abnormal in this image?	cardiovascular

Gambar 3.3. Sampel data VQA-RAD

3.4.4 Pemrosesan Data

Pada bagian ini, dilakukan pemrosesan data dengan tujuan menyelaraskan dan menyusun *dataset* yang diperlukan untuk model *Visual Question Answering* (VQA). Proses ini melibatkan penggabungan antara data visual, seperti gambar, dan data teks, berupa pertanyaan yang berkaitan. Pemrosesan data bertujuan untuk mengkonversi informasi kompleks dari kedua modalitas ini ke dalam format yang dapat dicerna oleh model. Dengan melakukan pemrosesan data ini, kita menciptakan representasi yang optimal untuk menyajikan informasi visual dan teks kepada model VQA. Hasilnya adalah *dataset* yang kohesif dan siap digunakan untuk melatih model VQA agar mampu memberikan jawaban yang tepat terhadap pertanyaan yang diajukan berdasarkan konteks visual yang diberikan. Adapun pemrosesan data yang dilakukan mencakup pemrosesan gambar dan teks.

1. Case Folding

Pada bagian ini proses *case folding* dilakukan untuk mengubah semua karakter dalam teks menjadi huruf kecil. Hal ini dilakukan untuk menghindari duplikasi kata yang sama dengan huruf besar dan kecil. Tabel 3.2 menunjukkan contoh proses *case folding*.

Tabel 3.2. Proses *case folding*

Sebelum	Sesudah
The Quick Brown Fox JUMPS	the quick brown fox jumps

2. Remove Punctuation

Pada bagian ini proses *remove punctuation* dilakukan untuk menghapus tanda baca dalam teks. Hal ini dilakukan untuk mengurangi kompleksitas kata dalam teks. Tabel 3.3 menunjukkan contoh proses *remove punctuation*.

Tabel 3.3. Proses *remove punctuation*

Sebelum	Sesudah
She said, 'Hello!' and waved	She said Hello and waved

3. Stemming

Pada bagian ini proses *stemming* dilakukan untuk mengubah kata-kata dalam teks menjadi kata dasar. Hal ini dilakukan untuk mengurangi kompleksitas kata dalam teks. Tabel 3.4 menunjukkan contoh proses *stemming-text*.

Tabel 3.4. Proses *stemming*

Sebelum	Sesudah
running quickly in the park	run quick in the park

4. Tokenisasi

Pada bagian ini proses tokenisasi dilakukan untuk memecah teks menjadi token-token yang lebih kecil. Hal ini dilakukan untuk mempermudah model VQA memahami dan memproses secara efektif. Tabel 3.5 menunjukkan contoh proses tokenisasi.

Tabel 3.5. Proses tokenisasi

Sebelum	Sesudah
The cat in the hat	[The, cat, in, the, hat]

5. Pemrosesan Gambar

Pada bagian ini proses pemrosesan gambar dilakukan menyamaratakan format gambar. Hal ini dilakukan untuk mempermudah proses pemrosesan data dan mempersiapkan data agar siap digunakan untuk melatih model VQA.

3.4.5 Membangun Model

Proses membangun model VQA ini menggunakan arsitektur VGG19-LSTM dan BLIP. Pada VGG19-LSTM, terdapat dua tahap pemrosesan, yaitu ekstraksi fitur gambar menggunakan VGG19 dan pemrosesan teks menggunakan LSTM. Sedangkan pada BLIP, terdapat dua *encoder*, yaitu *image encoder* dan *text encoder*, yang digunakan untuk mengolah data gambar dan teks. Dalam BLIP, *image encoder* menggunakan *Vision Transformer* (ViT), sedangkan *text encoder* menggunakan *transformer encoder*. Proses ini bertujuan untuk membangun model VQA yang mampu memberikan jawaban yang tepat terhadap pertanyaan yang diberikan berdasarkan konteks visual yang diberikan.

3.4.6 Melatih Model

Tahapan pada pelatihan model ini menggunakan arsitektur VGG19-LSTM dan BLIP. Pada tahap ini akan dilakukan suatu metode yang bertujuan untuk mengubah nilai atau parameter yang ada pada suatu arsitektur. Proses ini disebut sebagai *hyperparameter tuning*. *Hyperparameter* ini mengacu pada parameter yang tidak dapat diubah ketika proses pelatihan model berjalan, contohnya seperti jumlah *hidden layer*,

nilai *epoch*, *batch size*, *learning rate* pada fungsi optimasi, dan lain-lain. *Hyperparameter tuning* ini dilakukan untuk mencari kombinasi nilai yang optimal untuk parameter-parameter tersebut (Yu and Zhu, 2020).

3.4.7 Perbandingan Hasil

Setelah proses pelatihan model selesai, maka dilakukan perbandingan hasil dari kedua model yang telah dibangun dengan menggunakan metrik evaluasi yaitu akurasi dan *Bilingual Evaluation Understudy* (BLEU). Pada metrik BLEU, penulis akan menggunakan BLEU-1, BLEU-2, dan BLEU-3, bilangan pada nama BLEU menunjukkan jumlah n-gram yang digunakan. Kedua metrik ini digunakan untuk mengevaluasi kinerja dari model VQA yang telah dilatih. Perbandingan hasil ini dilakukan dengan cara menguji kedua model menggunakan data uji yang telah disiapkan. Hasil dari perbandingan ini akan digunakan untuk menentukan model mana yang lebih baik dalam memberikan jawaban dari pertanyaan yang diberikan.

3.4.8 Membangun Sistem Medis Cerdas Berbasis Web

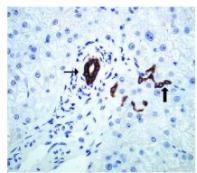
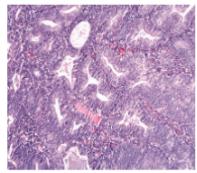
Model yang telah siap dilatih akan di *deploy* ke dalam sistem medis cerdas berbasis web. Dalam proses membangun *website* ini meliputi pengembangan halaman tampilan yang bisa menerima dua *input* yaitu gambar dan pertanyaan dari pengguna. Setelah itu mengembangkan bagian *backend* untuk mengolah kedua *input* tersebut dan memasukkannya ke dalam model yang telah dibangun sebelumnya. Setelah model menghasilkan jawaban, maka jawaban tersebut akan dikirimkan kembali ke halaman tampilan untuk ditampilkan kepada pengguna.

BAB IV

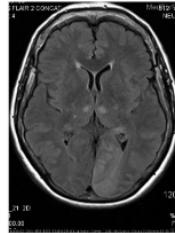
HASIL DAN PEMBAHASAN

4.1 PENGUMPULAN DATA

Pengumpulan *dataset* PathVQA dan VQA-RAD dilakukan melalui pengunduhan dari platform Huggingface menggunakan pustaka Huggingface Datasets versi 2.18.0. Sampel data dari kedua *dataset* ini dapat dilihat pada Gambar 4.1 dan Gambar 4.2.

Image	Question	Answer
	Where are liver stem cells (oval cells) located?	in the canals of hering
	Are bile duct cells and canals of herring stained here with an immunohistochemical stain for cytokeratin 7?	yes
	Is hyperplasia with atypia seen as glandular crowding and cellular atypia?	yes
	What is seen as glandular crowding and cellular atypia?	hyperplasia without atypia

Gambar 4.1. Sampel data dari *dataset* PathVQA

Image	Question	Answer
	What artery is the embolus from?	distal basilar artery
	What pathology does the hyperlucency represent?	embolus
	Is the heart size abnormal?	no
	Is there hilar adenopathy?	yes

Gambar 4.2. Sampel data dari *dataset* VQA-RAD

Kedua *dataset* ini memiliki struktur data bertipe *DatasetDict*, yang merupakan format struktur data khas dari *dataset* Huggingface. Dalam penelitian ini, kedua *dataset* dikonversi menjadi format JSON (*JavaScript Object Notation*) untuk memudahkan penulis dalam mengolah data gambar dan teks. Apabila menggunakan

format *DatasetDict*, penulis perlu mengakses data tersebut dengan menerapkan metode atau teknik tertentu sesuai dengan dokumentasi dari Huggingface. Konversi data ke format JSON memungkinkan penulis untuk mengakses dan melakukan pemrosesan data gambar dan teks sesuai dengan keinginan penulis.

Pada bagian pemrosesan gambar, penulis menemukan ada beberapa gambar di *dataset* PathVQA yang memiliki format gambar selain *Red Green Blue* (RGB), yaitu *Cyan, Magenta, Yellow, Key (Black)* (CMYK). Pada bagian ini, penulis melakukan konversi gambar CMYK menjadi RGB menggunakan pustaka Python Imaging Library (PIL). Konversi gambar ini dilakukan untuk memastikan bahwa semua gambar dalam *dataset* PathVQA memiliki format gambar yang sama, yaitu RGB. Pada gambar di *dataset* VQA-RAD, semua gambar sudah memiliki format RGB, sehingga tidak perlu dilakukan konversi ke dalam bentuk RGB.

Pada bagian pemrosesan teks, kalimat dari pertanyaan telah dilakukan proses *case folding, remove punctuation, stemming*, dan tokenisasi untuk menghasilkan kalimat yang lebih bersih dan lebih mudah diproses oleh model.

Proses *case folding* dilakukan untuk mengubah seluruh huruf dalam kalimat menjadi huruf kecil, sehingga model dapat memahami kata yang sama dengan huruf besar dan huruf kecil sebagai kata yang sama. Pada Tabel 4.1 terdapat contoh hasil dari proses *case folding*.

Tabel 4.1. Proses *case folding* pada sampel *dataset* PathVQA dan VQA-RAD

Sebelum	Sesudah
<i>Where are liver stem cells (oval cells) located?</i>	<i>where are liver stem cells (oval cells) located?</i>
<i>Are bile duct cells and canals of hering stained here with an immunohistochemical stain for cytokeratin 7?</i>	<i>are bile duct cells and canals of hering stained here with an immunohistochemical stain for cytokeratin 7?</i>
<i>Is hyperplasia with atypia seen as glandular crowding and cellular atypia?</i>	<i>is hyperplasia with atypia seen as glandular crowding and cellular atypia?</i>
<i>What is seen as glandular crowding and cellular atypia?</i>	<i>What is seen as glandular crowding and cellular atypia?</i>
<i>What artery is the embolus from?</i>	<i>what artery is the embolus from?</i>
<i>What pathology does the hyperlucency represent?</i>	<i>what pathology does the hyperlucency represent?</i>
<i>Is the heart size abnormal?</i>	<i>is the heart size abnormal?</i>

<i>Is there hilar adenopathy?</i>	<i>is there hilar adenopathy?</i>
-----------------------------------	-----------------------------------

Setelah proses *case folding*, dilakukan proses *remove punctuation* untuk menghapus tanda baca pada kalimat. Sehingga model dapat memahami kata yang sama dengan tanda baca dan tanpa tanda baca sebagai kata yang sama. Pada Tabel 4.2 terdapat contoh hasil dari proses *remove punctuation*.

Tabel 4.2. Proses *remove punctuation* pada sampel dataset PathVQA dan VQA-RAD

Sebelum	Sesudah
<i>where are liver stem cells (oval cells) located?</i>	<i>where are liver stem cells oval cells located</i>
<i>are bile duct cells and canals of hering stained here with an immunohistochemical stain for cytokeratin 7?</i>	<i>are bile duct cells and canals of hering stained here with an immunohistochemical stain for cytokeratin 7</i>
<i>is hyperplasia with atypia seen as glandular crowding and cellular atypia?</i>	<i>is hyperplasia with atypia seen as glandular crowding and cellular atypia</i>
<i>What is seen as glandular crowding and cellular atypia?</i>	<i>what is seen as glandular crowding and cellular atypia</i>
<i>what artery is the embolus from?</i>	<i>what artery is the embolus from</i>
<i>what pathology does the hyperlucency represent?</i>	<i>what pathology does the hyperlucency represent</i>
<i>is the heart size abnormal?</i>	<i>is the heart size abnormal</i>
<i>is there hilar adenopathy?</i>	<i>is there hilar adenopathy</i>

Langkah selanjutnya adalah proses *stemming* yang dilakukan untuk mengubah kata-kata dalam kalimat menjadi kata dasar. Sehingga model dapat memahami kata yang memiliki akar kata yang sama sebagai kata yang sama. Pada Tabel 4.3 terdapat contoh hasil dari proses *stemming*.

Tabel 4.3. Proses *stemming* pada sampel dataset PathVQA dan VQA-RAD

Sebelum	Sesudah
<i>where are liver stem cells oval cells located</i>	<i>where are liver stem cell oval cell locat</i>

<i>are bile duct cells and canals of herring stained here with an immunohistochemical stain for cytokeratin 7</i>	<i>are bile duct cell and canal of here stain here with an immunohistochem stain for cytokeratin 7</i>
<i>is hyperplasia with atypia seen as glandular crowding and cellular atypia</i>	<i>is hyperplasia with atypia seen as glandular crowd and cellular atypia</i>
<i>what is seen as glandular crowding and cellular atypia</i>	<i>what is seen as glandular crowd and cellular atypia</i>
<i>what artery is the embolus from</i>	<i>what arteri is the embolu from</i>
<i>what pathology does the hyperlucency represent</i>	<i>what patholog doe the hyperluc repres</i>
<i>is the heart size abnormal</i>	<i>is the heart size abnorm</i>
<i>is there hilar adenopathy</i>	<i>is there hilar adenopathi</i>

Lalu, setelah proses *stemming*, dilakukan proses tokenisasi untuk mengubah kalimat menjadi token. Tokenisasi dilakukan untuk memecah kalimat menjadi kata-kata yang lebih kecil, sehingga model dapat memahami kata-kata dalam kalimat sebagai token. Pada Tabel 4.4 terdapat contoh hasil dari proses tokenisasi pada sampel dataset PathVQA dan VQA-RAD.

Tabel 4.4. Proses tokenisasi pada sampel dataset PathVQA dan VQA-RAD

Sebelum	Sesudah
<i>where are liver stem cell oval cell locat</i>	<i>[where, are, liver, stem, cell, oval, cell, locat]</i>
<i>are bile duct cell and canal of here stain here with an immunohistochem stain for cytokeratin 7</i>	<i>[are, bile, duct, cell, and, canal, of, here, stain, here, with, an, immunohistochem, stain, for, cytokeratin, 7]</i>
<i>is hyperplasia with atypia seen as glandular crowd and cellular atypia</i>	<i>[is, hyperplasia, with, atypia, seen, as, glandular, crowd, and, cellular, atypia]</i>
<i>what is seen as glandular crowd and cellular atypia</i>	<i>[what, is, seen, as, glandular, crowd, and, cellular, atypia]</i>
<i>what arteri is the embolu from</i>	<i>[what, arteri, is, the, embolu, from]</i>
<i>what patholog doe the hyperluc repres</i>	<i>[what, patholog, doe, the, hyperluc, repres]</i>
<i>is the heart size abnorm</i>	<i>[is, the, heart, size, abnorm]</i>
<i>is there hilar adenopathi</i>	<i>[is, there, hilar, adenopathi]</i>

Setelah semua data gambar dan teks telah dilakukan pemrosesan, data tersebut kemudian dibagi menjadi data latih, data validasi, dan data uji. Data latih digunakan untuk melatih model, data validasi digunakan untuk mengevaluasi model selama proses pelatihan, dan data uji digunakan untuk mengevaluasi model setelah proses pelatihan selesai. Rasio pembagian data latih, data validasi, dan data uji pada *dataset* PathVQA dan VQA-RAD adalah 80%, 10%, dan 10%.

4.2 PENGEMBANGAN MODEL

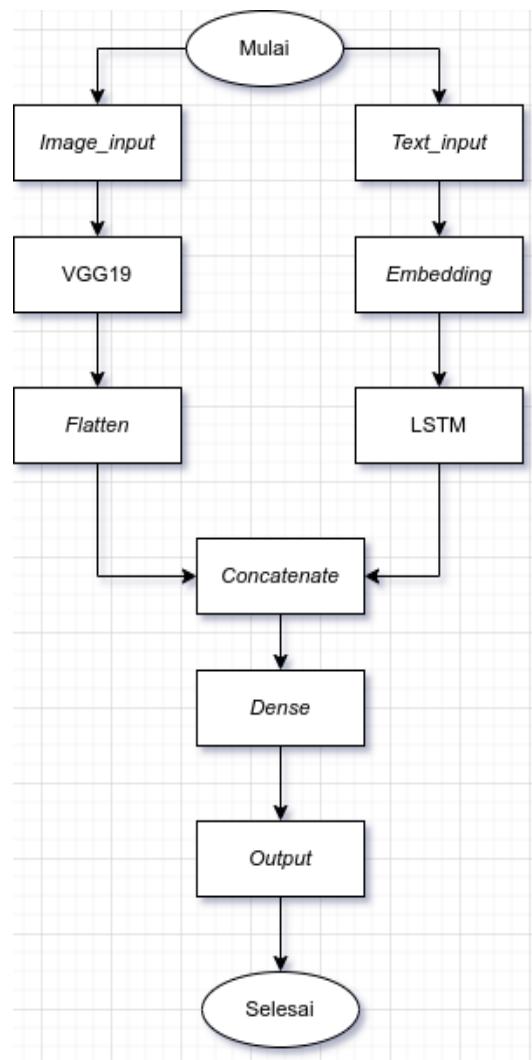
Adapun langkah-langkah yang dilakukan dalam pembangunan model VQA dengan arsitektur VGG19-LSTM dan BLIP adalah sebagai berikut:

4.2.1 VGG19-LSTM

Pengembangan model dengan VGG19-LSTM dilakukan menggunakan pustaka *TensorFlow* versi 2.11.0. Model VGG19 yang telah dilatih pada *dataset* ImageNet digunakan sebagai *pretrained model* untuk mengekstraksi fitur dari gambar yang telah di *resize* menjadi 224×224 piksel. Fitur ini kemudian digabungkan dengan fitur teks yang diolah menggunakan GloVe *embedding layer* berdimensi 100. Fitur teks ini mengubah teks menjadi vektor numerik yang dapat diproses oleh model. Fitur gabungan ini kemudian dijadikan sebagai *input* untuk LSTM yang memiliki 128 unit, yang bertugas memprediksi jawaban dari pertanyaan yang diberikan berdasarkan fitur gabungan tersebut.

Dalam pemrosesan teks, teks yang telah dilakukan pemrosesan sebelumnya diubah menjadi vektor dengan menggunakan GloVe *embedding*. Setiap kata dalam teks pertanyaan yang terdapat dalam GloVe diwakili sebagai vektor 100 dimensi. Proses ini memungkinkan model untuk menginterpretasikan pertanyaan dalam bentuk numerik yang lebih mudah diproses oleh jaringan saraf. Dalam menangani data jawaban dilakukan suatu teknik bernama *one-hot encoding*. Teknik *one-hot encoding* digunakan untuk merepresentasikan jawaban yang mungkin. Pada hal ini, representasi *one-hot encoding*, setiap jawaban diubah menjadi sebuah vektor di mana hanya satu elemen yang bernilai satu dan sisanya bernilai nol. Panjang vektor ini sesuai dengan jumlah total jawaban yang mungkin. Proses ini mengubah kategori jawaban menjadi format yang dapat langsung diproses oleh lapisan *softmax* dari model, yang memudahkan model dalam melakukan klasifikasi dan menghitung *loss* dengan efektif menggunakan fungsi *loss categorical crossentropy*. Dengan ini, setiap probabilitas *output* dari model langsung mencerminkan kepercayaan model terhadap masing-masing kategori jawaban yang mungkin, memfasilitasi interpretasi yang lebih mudah dan evaluasi performa model yang lebih akurat.

Output dari lapisan VGG19 dan LSTM kemudian digabungkan melalui operasi *concatenate*. Gabungan fitur ini diolah lebih lanjut menggunakan lapisan *fully connected* dengan 256 unit dan fungsi aktivasi *Rectified Linear Unit* ReLU, dilanjutkan dengan lapisan *output* yang menggunakan fungsi aktivasi *softmax* untuk mengklasifikasikan jawaban dari beberapa pilihan jawaban yang mungkin. Struktur dari model VGG19-LSTM yang digunakan dalam penelitian pada saat melatih *dataset* PathVQA dan VQA-RAD dapat dilihat pada Gambar 4.3.



Gambar 4.3. Struktur arsitektur VGG19-LSTM

Berdasarkan Gambar 4.3, model VGG19-LSTM terdiri dari beberapa lapisan, penjelasan dari setiap lapisan tersebut adalah sebagai berikut:

1. Lapisan "image_input" adalah lapisan *input* untuk gambar yang menerima gambar dengan ukuran 224×224 piksel dan 3 saluran warna (RGB). Lapisan ini tidak memiliki parameter yang bisa dilatih karena hanya berfungsi sebagai *input*.

2. Lapisan "*text_input*" adalah lapisan *input* untuk teks yang menerima teks dengan panjang maksimal 24 kata (atau token). Seperti halnya "*image_input*", lapisan ini juga tidak memiliki parameter yang bisa dilatih karena hanya berfungsi sebagai *input*.
3. Lapisan "vgg19" adalah model VGG19 yang sudah dilatih sebelumnya, digunakan untuk ekstraksi fitur gambar. *Output* dari lapisan ini adalah tensor dengan dimensi yang belum ditentukan (akan ditentukan saat runtime) dan memiliki 512 fitur. Lapisan ini terhubung ke lapisan "*image_input*" karena menerima *output* dari lapisan tersebut.
4. Lapisan "*embedding*" adalah lapisan yang mengubah token teks menjadi vektor *embedding*. *Output* dari lapisan ini adalah tensor dengan panjang 24 dan vektor *embedding* dengan 100 dimensi. Lapisan ini terhubung ke lapisan "*text_input*" karena menerima *output* dari lapisan tersebut.
5. Lapisan "*flatten*" adalah lapisan yang mengubah tensor multi-dimensi menjadi vektor 1 dimensi. *Output* dari lapisan ini adalah vektor dengan panjang 25.088. Lapisan ini terhubung ke *output* dari lapisan "vgg19" karena menerima *output* dari lapisan tersebut.
6. Lapisan "lstm" adalah lapisan LSTM yang mengolah urutan *embedding* teks. *Output* dari lapisan ini adalah tensor dengan 128 unit. Lapisan ini terhubung ke *output* dari lapisan "*embedding*" karena menerima *output* dari lapisan tersebut.
7. Lapisan "*concatenate*" adalah lapisan yang menggabungkan fitur gambar dan teks. *Output* dari lapisan ini adalah tensor gabungan dengan panjang 25.216. Lapisan ini terhubung ke *output* dari lapisan "flatten" dan "lstm" karena menerima *output* dari kedua lapisan tersebut.
8. Lapisan "*dense*" setelah lapisan "*concatenate*" adalah lapisan "*dense*" dengan 256 unit. *Output* dari lapisan ini adalah tensor dengan 256 unit. Lapisan ini terhubung ke *output* dari lapisan "*concatenate*" karena menerima *output* dari lapisan tersebut.
9. Lapisan "*output*" adalah lapisan yang digunakan untuk prediksi akhir dengan 4879 kelas pada *dataset* PathVQA dan 517 kelas pada *dataset* VQA-RAD. *Output* dari lapisan ini adalah tensor dengan 4879 unit (kelas) pada *dataset* PathVQA dan 517 unit (kelas) pada *dataset* VQA-RAD.

Struktur dari model VGG19-LSTM yang dilatih dengan *dataset* PathVQA dan VQA-RAD hanya berbeda pada lapisan terakhir, yaitu lapisan "*dense*" yang memiliki

jumlah kelas yang berbeda. Pada *dataset* PathVQA, jumlah kelas yang digunakan adalah 4879 kelas, sedangkan pada *dataset* VQA-RAD, jumlah kelas yang digunakan adalah 517 kelas. Hal ini disebabkan oleh perbedaan jumlah kelas jawaban pada kedua *dataset*. Sehingga, model VGG19-LSTM yang dilatih dengan *dataset* PathVQA memiliki parameter yang lebih banyak dibandingkan dengan model VGG19-LSTM yang dilatih dengan *dataset* VQA-RAD. Jumlah parameter pada model VGG19-LSTM yang dilatih dengan *dataset* PathVQA adalah 27.851.187, sedangkan jumlah parameter pada model VGG19-LSTM yang dilatih dengan *dataset* VQA-RAD adalah 26.730.153.

Model ini dikompilasi dengan menggunakan fungsi optimasi Adam dan menggunakan fungsi *loss categorical crossentropy* dikarenakan masalah yang dihadapi adalah masalah klasifikasi multikelas. Dengan adanya fungsi optimasi disini navigasi ruang parameter dapat terjadi yaitu dengan adanya fungsi optimasi yang bekerja dengan menghitung gradien dari fungsi *loss* terhadap masing-masing parameter model. Gradien disini dapat memberitahu fungsi optimasi arah untuk mengubah parameter guna mengurangi *loss*. Secara intuitif, gradien akan menunjukkan arah tercepat untuk turun dari bukit dalam ruang parameter menuju ke titik terendah. Sehingga, model dapat belajar dari data latih dan meningkatkan performa model secara iteratif.

Secara keseluruhan, model VGG19-LSTM yang dilatih dengan *dataset* PathVQA dan VQA-RAD memiliki arsitektur yang sama, namun berbeda pada jumlah kelas yang digunakan. Model ini memiliki tiga lapisan utama, yaitu lapisan VGG19, lapisan LSTM, dan lapisan *output* yang digunakan untuk memprediksi jawaban. Model ini memiliki kemampuan untuk memproses gambar dan teks secara bersamaan, dan menghasilkan prediksi jawaban berdasarkan gambar dan pertanyaan yang diberikan.

4.2.2 BLIP

Pengembangan model dengan BLIP dilakukan menggunakan pustaka PyTorch versi 2.0.1 dan Transformers versi 4.31.0. Di sini, PyTorch digunakan sebagai *deep learning framework* untuk melatih model, sedangkan Transformers digunakan untuk mengakses *pre-trained* model BLIP. Sebelum melatih model dengan BLIP, penulis melakukan pemrosesan khusus pada data gambar dan data teks menggunakan pustaka Transformers dari Huggingface untuk mengkonversi data tersebut menjadi format yang dapat diproses oleh BLIP. Pustaka Transformers yang digunakan adalah *BlipImageProcessor* dan *BlipProcessor*. Kedua pustaka ini mengkonversi data gambar dan teks menjadi format yang sesuai untuk pemrosesan oleh BLIP.

Pada model BLIP, data gambar diolah menggunakan *BlipImageProcessor* dari pustaka Transformers. Fungsi *BlipImageProcessor* mengubah ukuran gambar menjadi 128×128 piksel, dan mengubah gambar yang telah diolah menjadi format tensor yang dapat diproses oleh model BLIP. Sedangkan, untuk data teks diolah menggunakan

BlipProcessor. Dalam hal ini, *BlipProcessor* melakukan tugas-tugas khusus seperti *padding*, *truncation*, dan *encoding*. *Padding* bertugas menambahkan token khusus pada data teks untuk memastikan bahwa semua data teks memiliki panjang yang sama. *Truncation* bertugas memotong data teks yang melebihi panjang maksimum yang telah ditentukan, yaitu 32 token. *Encoding* mengkonversi data teks menjadi format numerik yang dapat diproses oleh model BLIP.

Setelah data gambar dan teks diolah, penulis kemudian mengunduh model yang telah dilatih sebelumnya dari BLIP, yang digunakan untuk tugas *Visual Question Answering* (VQA) menggunakan pustaka Transformers. Model ini dilatih pada *dataset* Visual Genome dan VQA v2.0, dan kemudian dilakukan *fine-tuning* menggunakan *dataset* PathVQA dan VQA-RAD. Dalam proses *fine-tuning* ini, model BLIP dilatih menggunakan fungsi optimasi Adam dan fungsi *loss categorical crossentropy*, karena masalah yang dihadapi adalah klasifikasi multikelas. Fungsi optimasi Adam di sini memainkan peran penting dalam meminimalkan nilai *loss* yang dihasilkan oleh fungsi *loss categorical crossentropy*. Fungsi *loss* mengukur kesalahan prediksi model terhadap kelas yang sebenarnya, dan nilai *loss* yang lebih rendah menunjukkan kesesuaian yang lebih baik antara prediksi model dan kenyataan. Fungsi optimasi, dalam hal ini Adam, secara efektif mengatur proses pembaharuan bobot model berdasarkan gradien dari fungsi *loss*. Proses ini penting untuk mengarahkan model agar mencapai efisiensi maksimal dalam memprediksi kelas yang benar, secara bertahap menurunkan kesalahan prediksi sepanjang iterasi pelatihan. Oleh karena itu, pemilihan fungsi optimasi yang tepat sangat krusial untuk memastikan keberhasilan *fine-tuning* model dalam mencapai hasil yang akurat dan relevan.

Struktur model BLIP pada tugas *Visual Question Answering* (VQA) terdiri dari tiga lapisan utama, yaitu *vision_model*, *text_encoder*, dan *text_decoder*. Ketiga lapisan ini berfungsi untuk memproses *input* gambar dan teks, serta menghasilkan prediksi jawaban akhir berdasarkan gambar dan pertanyaan yang diberikan. Struktur dari ketiga lapisan ini dapat dilihat pada Lampiran 1, Lampiran 2, dan Lampiran 3.

Lapisan *vision_model* bertugas untuk memproses *input* gambar dan terdiri dari dua bagian utama: *embeddings* dan *encoder*. Di bagian *embeddings* menggunakan *backbone Vision Transformer* (ViT) untuk mengubah gambar menjadi *patch embedding*. Bagian *encoder* terdiri dari beberapa lapisan *encoder* yang menangkap hubungan antar *patch* gambar menggunakan mekanisme *self-attention*. Setiap *layer encoder* dilengkapi dengan normalisasi dan *Multi-Layer Perceptron* (MLP) dengan aktivasi *Gaussian Error Linear Unit* (GELU) untuk transformasi lebih lanjut. Berikut adalah penjelasan dari setiap lapisan dalam *vision_model*:

1. *Embeddings*: Ini merupakan komponen dari *BlipVisionModel* yang menggunakan

BlipVisionEmbeddings untuk menerapkan konvolusi 2D (Conv2d) dengan menggunakan *backbone* ViT untuk mengubah gambar menjadi representasi vektor dari potongan-potongan gambar (*patches*).

2. *BlipVisionEmbeddings*: Ini merupakan komponen yang mencakup *patch embeddings* menggunakan Conv2d. Dimensi *input* adalah 3 saluran warna RGB, dimensi *output* adalah 768, ukuran kernel 16×16 , dan stride 16×16 .
3. *Encoder*: Bagian yang mencakup *BlipEncoder* yang terdiri dari beberapa lapisan *encoder* yang masing-masing menggunakan mekanisme *self-attention* untuk menangkap hubungan antar *patch* gambar.
4. *BlipEncoder*: Ini merupakan komponen utama *encoder* yang terdiri dari beberapa *BlipEncoderLayer*, dengan jumlah total 12 lapisan.
5. *BlipEncoderLayer*: Bagian dari *BlipEncoder* yang dimana setiap lapisan yang terdiri dari bagian terdapat beberapa subkomponen yaitu *self-attention*, MLP, dua *LayerNorm*. Di bagian ini, *input* dari gambar diolah menggunakan mekanisme *self-attention* dan MLP untuk transformasi lebih lanjut.
6. *BlipAttention*: Bagian ini menangkap hubungan antar *patch* gambar menggunakan mekanisme *self-attention*. Komponen ini menggunakan *query*, *key*, dan *value* dengan dimensi 768×2304 dan proyeksi dengan dimensi 768×768 . Nilai *dropout* adalah 0,0.
7. *BlipMLP*: Menggunakan lapisan linear untuk transformasi lebih lanjut dari hasil *attention* dan fungsi aktivasi GELU untuk menambah non-linearitas. Lapisan pertama (fc1) memiliki dimensi 768×3072 dan lapisan kedua (fc2) memiliki dimensi 3072×768 .
8. *LayerNorm*: Digunakan untuk normalisasi pada beberapa bagian model, termasuk *layer_norm1*, *layer_norm2*, dan *post_layernorm*. Parameter *LayerNorm* memiliki nilai epsilon 1×10^{-5} .
9. *PostLayerNorm*: Merupakan lapisan normalisasi akhir setelah seluruh lapisan *encoder* selesai memproses *input*. Tujuannya adalah untuk menstabilkan dan menormalisasi keluaran akhir dari *encoder* sebelum digunakan oleh bagian lain dari model.

Lapisan *text_encoder* bertugas untuk memproses *input* teks dan terdiri dari dua bagian utama: *embeddings* dan *encoder*. Bagian *embeddings* menyediakan representasi kata dan posisi dalam teks, sementara bagian *encoder* terdiri dari beberapa lapisan

yang menangkap hubungan antar kata menggunakan mekanisme *attention*. Selain *self-attention*, terdapat *cross-attention* yang menangkap hubungan antara teks dan fitur gambar jika ada. Setiap *layer encoder* juga dilengkapi dengan normalisasi dan MLP untuk transformasi lebih lanjut. Berikut adalah penjelasan dari setiap lapisan dalam *text_encoder*:

1. *Embeddings*: Pada bagian ini terdiri dari *word embeddings* yang merupakan representasi vektor dari kata-kata dalam teks dan *position embeddings* yang memberikan informasi posisi dari setiap kata dalam teks. Jumlah fitur yang digunakan adalah 768 dengan *padding* indeks 0.
2. *BlipTextEmbeddings*: Ini merupakan komponen yang mencakup *word embeddings* dan *position embeddings*, serta menggunakan *LayerNorm* untuk normalisasi dan *dropout* untuk mengurangi *overfitting*. Dimensi *word embeddings* adalah 30524×768 dimensi, sedangkan dimensi *position embeddings* adalah 512×768 dimensi. Parameter *LayerNorm* memiliki nilai epsilon 1×10^{-12} , dengan nilai *dropout* 0,0.
3. *Encoder*: Pada bagian ini terdiri dari beberapa lapisan yang masing-masing menggunakan mekanisme *attention* untuk menangkap hubungan antar kata. Hal ini bertujuan untuk memahami dan mengintegrasikan informasi pada pertanyaan yang diajukan.
4. *BlipTextEncoder*: Ini merupakan komponen *encoder* yang terdiri dari beberapa lapisan *BlipTextLayer*.
5. *BlipTextLayer*: Setiap lapisan terdiri dari beberapa subkomponen yaitu *attention*, *crossattention*, *intermediate*, dan *output*, dengan jumlah total 12 lapisan *BlipTextLayer*.
6. *BlipTextAttention*: Hal ini berguna untuk menangkap hubungan antar kata dalam teks menggunakan *self-attention* dan hubungan antara teks dan fitur gambar menggunakan *cross-attention*. Dimensi *query*, *key*, dan *value* masing-masing adalah 768×768 dengan nilai *dropout* 0,0.
7. *BlipTextSelfAttention*: Bagian dari *BlipTextAttention* yang menggunakan *query*, *key*, dan *value* untuk menghitung perhatian antar kata dalam teks, dengan dimensi *query*, *key*, dan *value* masing-masing 768×768 .
8. *BlipTextSelfOutput*: Bagian ini menghasilkan *output* dari mekanisme *attention* dan menggunakan *LayerNorm* serta *dropout* untuk normalisasi dan mengurangi

overfitting. Dimensi dense layer adalah 768×768 dengan parameter *LayerNorm* epsilon 1×10^{-12} dan nilai *dropout* 0,0.

9. *BlipTextIntermediate*: Bagian ini menggunakan lapisan linear untuk transformasi lebih lanjut dari hasil *attention* dan fungsi aktivasi GELU untuk menambah non-linearitas. Dimensi dense layer adalah 768×3072 dengan fungsi aktivasi GELU.
10. *BlipTextOutput*: Di sini dihasilkan *output* akhir dari lapisan *encoder* dengan lapisan linear dan *LayerNorm* untuk normalisasi. Dimensi dense layer adalah 3072×768 dengan parameter *LayerNorm* epsilon 1×10^{-12} serta nilai *dropout* 0,0.

Lapisan *text_decoder* bertugas untuk mendekode dan menghasilkan teks dari fitur yang telah diproses, menggunakan struktur yang mirip dengan model *Bidirectional Encoder Representation From Transformers* (BERT). Bagian *decoder* terdiri dari *embeddings* untuk kata dan posisi, serta beberapa lapisan *encoder* yang menggunakan mekanisme *self-attention* dan *cross-attention*. Setiap lapisan dilengkapi dengan MLP dan normalisasi untuk transformasi lebih lanjut. Bagian *cls* dari *decoder* bertugas untuk menghasilkan prediksi jawaban akhir berdasarkan fitur yang telah diproses. Berikut adalah penjelasan dari setiap lapisan dalam *text_decoder*:

1. *Embeddings*: Menggunakan *word embeddings* dan *position embeddings* untuk menyediakan representasi vektor dari kata dan posisi dalam teks. Pada bagian ini, dimensi pada *word embeddings* adalah 30524×768 , sedangkan dimensi *position embeddings* adalah 512×768 . Lapisan ini juga menggunakan *LayerNorm* dengan nilai epsilon 1×10^{-12} dan *dropout* dengan nilai 0,0.
2. *BlipTextEmbeddings*: Bagian ini, merupakan komponen utama yang mencakup *word embeddings* dan *position embeddings*, serta menggunakan *LayerNorm* dan *dropout* untuk normalisasi dan mengurangi *overfitting*. Pada bagian ini, dimensi *word embeddings* adalah 30524×768 , dimensi *position embeddings* adalah 512×768 , nilai epsilon 1×10^{-12} , dan nilai *dropout* 0,0.
3. *Encoder*: Terdiri dari beberapa lapisan *encoder* yang menggunakan mekanisme *self-attention* dan *cross-attention* untuk menangkap hubungan antar kata dan antara kata dengan fitur lain. Terdapat total 12 lapisan *BlipTextLayer*.
4. *BlipTextEncoder*: Ini merupakan komponen utama *encoder* yang terdiri dari beberapa *BlipTextLayer*, dengan jumlah total 12 lapisan.

5. *BlipTextLayer*: Setiap lapisan terdiri dari beberapa subkomponen yaitu *attention*, *crossattention*, *intermediate*, dan *output*. Lapisan *attention* dan *crossattention* masing-masing menggunakan *self-attention* dengan *query*, *key*, dan *value*, serta proyeksi dengan dimensi 768×768 dan nilai *dropout* 0,0. Lapisan *intermediate* menggunakan lapisan linear dengan dimensi 768×3072 dan fungsi aktivasi GELU, sementara lapisan *output* menggunakan lapisan linear dengan dimensi 3072×768 dan *LayerNorm* dengan nilai epsilon 1×10^{-12} .
6. *BlipTextAttention*: Hal ini berguna untuk menangkap hubungan antar kata dalam teks menggunakan mekanisme *self-attention*. Komponen ini menggunakan *query*, *key*, dan *value* dengan dimensi 768×768 dan proyeksi dengan dimensi 768×768 . Nilai *dropout* adalah 0,0.
7. *BlipTextSelfAttention*: Bagian dari *BlipTextAttention* yang menggunakan *query*, *key*, dan *value* untuk menghitung perhatian antar kata dalam teks, dengan dimensi *query*, *key*, dan *value* masing-masing 768×768 .
8. *BlipTextSelfOutput*: Bagian ini, menghasilkan *output* dari mekanisme *attention* dan menggunakan *LayerNorm* serta *dropout* untuk normalisasi dan mengurangi *overfitting*. Dimensi dense layer adalah 768×768 dengan parameter *LayerNorm* epsilon 1×10^{-12} serta nilai *dropout* 0,0.
9. *BlipTextIntermediate*: Menggunakan lapisan linear untuk transformasi lebih lanjut dari hasil *attention* dan fungsi aktivasi GELU untuk menambah non-linearitas. Dimensi dense layer adalah 768×3072 dengan fungsi aktivasi GELU.
10. *BlipTextOutput*: Menghasilkan *output* akhir dari lapisan *encoder* dengan lapisan linear dan *LayerNorm* untuk normalisasi. Dimensi dense layer adalah 3072×768 dengan parameter *LayerNorm* epsilon 1×10^{-12} serta nilai *dropout* 0,0.
11. *BlipTextOnlyMLMHead*: Bertugas untuk menghasilkan prediksi jawaban akhir berdasarkan fitur yang telah diproses. Bagian dari komponen ini mencakup *BlipTextLMPredictionHead* yang terdiri dari transformasi dan *decoder*. Transform menggunakan lapisan linear dengan dimensi 768×768 , fungsi aktivasi GELU, dan *LayerNorm* dengan nilai epsilon 1×10^{-12} . Bagian *decoder* menggunakan lapisan linear dengan dimensi 768×30524 .
12. *BlipTextLMPredictionHead*: Ini merupakan komponen dari *cls* yang mencakup transformasi dan *decoder*. Disini, dilakukan transformasi dengan menggunakan lapisan linear dengan dimensi 768×768 , fungsi aktivasi GELU, dan *LayerNorm*

dengan nilai epsilon 1×10^{-12} . Disini, *decoder* menggunakan lapisan linear dengan dimensi 768×30524 .

13. *BlipTextPredictionHeadTransform*: Bagian dari *BlipTextLMPredictionHead* yang menggunakan lapisan linear untuk transformasi fitur dengan dimensi 768×768 , fungsi aktivasi GELU, dan *LayerNorm* dengan nilai epsilon 1×10^{-12} .

Secara keseluruhan, model BLIP pada tugas VQA yang terdiri dari tiga lapisan utama, yaitu *vision_model*, *text_encoder*, dan *text_decoder*, berguna untuk memproses kedua *input* yaitu gambar dan pertanyaan, serta menghasilkan prediksi jawaban akhir berdasarkan kedua *input* tersebut. Dengan adanya tiga lapisan ini, model BLIP dapat memahami dan merespon pertanyaan visual dengan lebih efektif dan efisien, sehingga meningkatkan akurasi dan relevansi jawaban yang dihasilkan, sehingga dapat membantu dalam mendukung pertanyaan dengan konteks visual yang ada.

4.3 AUGMENTASI DATA

Pada penelitian ini, dilakukan augmentasi data untuk kalimat pertanyaan pada *dataset* PathVQA dan VQA-RAD. Augmentasi ini menggunakan teknik augmentasi teks dari pustaka nlpaug versi 1.1.11. Teknik ini memanfaatkan fungsi dari nlpaug untuk menghasilkan variasi parafrase dari setiap pertanyaan dalam *dataset*. Pada penelitian ini, setiap pertanyaan diubah menjadi tujuh variasi yang berbeda. Setiap versi parafrase dari pertanyaan tersebut kemudian dibuat sebagai entri baru dalam *dataset*, di mana setiap entri baru masih mengacu pada gambar dan jawaban yang sama dengan pertanyaan asli. Metode augmentasi ini meningkatkan variasi linguistik dalam *dataset*, yang sangat bermanfaat untuk melatih model NLP agar dapat memahami dan merespon berbagai formulasi pertanyaan secara lebih efektif. Augmentasi dilakukan dengan menggunakan sinonim dari leksikal WordNet untuk menghasilkan sinonim untuk kata-kata dalam kalimat yang ada, sehingga menghasilkan variasi parafrase dari kalimat asli. Pada saat proses augmentasi ada bermunculan data yang duplikat, oleh karena itu penulis melakukan penghapusan data duplikat tersebut. Dengan menerapkan teknik augmentasi ini, awalnya *dataset* PathVQA memiliki 32.632 data, setelah dilakukan augmentasi data menjadi 179.167 data. Sedangkan *dataset* VQA-RAD yang awalnya memiliki 2.248 data, setelah dilakukan augmentasi data menjadi 14.347 data. Pada hal ini tujuan dilakukan augmentasi data karena pertanyaan terkait masalah medis sangatlah beragam dan kompleks, sehingga diperlukan variasi pertanyaan yang lebih banyak dan mengatasi kurangnya variasi pertanyaan pada *dataset* yang ada. Pada Tabel 4.5 menunjukkan contoh pertanyaan yang telah diaugmentasi dengan memanfaatkan teknik augmentasi teks dari pustaka nlpaug dengan menggunakan sinonim dari leksikal WordNet.

Tabel 4.5. Contoh pertanyaan yang telah diaugmentasi

Pertanyaan original	Pertanyaan augmentasi
	<i>What is the elemental use of magnetic resonance imaging in aesculapian diagnosis?</i>
<i>What is the primary use of MRI in medical diagnosis?</i>	<i>What be the primary usance of magnetic resonance imaging in medical diagnosis?</i>
	<i>What embody the main use of magnetic resonance imaging in aesculapian diagnosis?</i>
	<i>What is the primary habit of magnetic resonance imaging in medical diagnosis?</i>
	<i>What be the primary use of MRI in medical diagnosis?</i>
	<i>What is the primary usance of magnetic resonance imaging in aesculapian diagnosis?</i>
	<i>What be the primary use of magnetic resonance imaging in medical diagnosis?</i>

4.4 ANALISIS HASIL

Dalam rangka melakukan eksperimen, penulis telah melakukan konfigurasi dari berbagai *hyperparameter* untuk melatih model VGG19-LSTM dan BLIP, dengan tujuan meningkatkan efektivitas dan efisiensi pelatihan. Dalam hal ini, *hyperparameter* yang diatur mencakup jumlah *epochs*, *learning rate*, dan *batch size*. Peran ketiga *hyperparameter* ini memiliki peran penting dalam menentukan keberhasilan pelatihan model, karena mempengaruhi proses optimisasi model untuk mencapai hasil yang optimal. *Hyperparameter* ini memungkinkan penyesuaian yang lebih spesifik terhadap karakteristik data dan kebutuhan eksperimental, sehingga memungkinkan model untuk belajar dengan lebih baik dan menghasilkan respons yang lebih akurat terhadap pertanyaan visual yang diajukan. Tabel 4.6 menunjukkan konfigurasi *hyperparameter* yang digunakan dalam eksperimen ini untuk melatih model VGG19-LSTM dan BLIP dengan *dataset* PathVQA dan VQA-RAD, sebelum dilakukan augmentasi data. Hal ini dilakukan untuk melihat pengaruh konfigurasi *hyperparameter* terhadap performa model pada data asli sebelum dilakukan augmentasi.

Tabel 4.6. Konfigurasi *hyperparameter* pada awal eksperimen

No	Konfigurasi <i>hyperparameter</i>		
	Epochs	Learning rate	Batch size
1	15	1×10^{-5}	8
			16
			32
	30	5×10^{-5}	8
			16
			32
2	45	1×10^{-5}	8
			16
			32
	30	5×10^{-5}	8
			16
			32
3	15	1×10^{-5}	8
			16
			32
	45	5×10^{-5}	8
			16
			32

Konfigurasi *hyperparameter* yang digunakan dalam eksperimen ini merupakan pilihan berdasarkan preferensi peneliti untuk mendapatkan kombinasi awal yang rasional. Eksperimen dengan jumlah *epochs* 15, 30, dan 45, *batch size* 8, 16, dan 32, serta *learning rate* sebesar 1×10^{-5} , dan 5×10^{-5} . Dengan menggunakan kombinasi *hyperparameter* ini, penulis akan mencari kombinasi yang paling optimal dengan menggunakan metode *grid search* untuk mencari kombinasi *hyperparameter* yang paling optimal.

Dari hasil eksperimen yang dilakukan dengan VGG19-LSTM dengan *dataset* PathVQA dan VQA-RAD didapatlah hasil untuk pengujian pada pertanyaan yang bersifat *close-ended* dapat dilihat pada Lampiran 4, sedangkan untuk pengujian pada pertanyaan yang bersifat *open-ended* dapat dilihat pada Lampiran 5, dan untuk pengujian pada keseluruhan pertanyaan dapat dilihat pada Lampiran 6. Dari hasil eksperimen tersebut, dapat dilihat bahwa beberapa hasil akurasi dari model VGG19-LSTM dengan *dataset* PathVQA dan VQA-RAD dapat dilihat pada Tabel 4.7. Pada Tabel 4.8 disajikan informasi mengenai hasil BLEU pada pertanyaan *open-ended* dengan VGG19-LSTM.

Tabel 4.8 menunjukkan beberapa hasil terbaik dari model VGG19-LSTM dengan *dataset* PathVQA dan VQA-RAD berdasarkan nilai BLEU-1, BLEU-2, dan BLEU-3 untuk melihat kualitas jawaban yang bersifat *open-ended* yang dihasilkan oleh model.

Tabel 4.7. Hasil akurasi pelatihan dengan VGG19-LSTM

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	Akurasi <i>close ended</i>	Akurasi <i>open ended</i>
PathVQA	15	8	1×10^{-5}	26,96%	43,02%	10,16%
	15	8	5×10^{-5}	37,07%	53,44%	19,90%
	45	16	5×10^{-5}	35,39%	51,23%	19,77%
	45	8	5×10^{-5}	35,20%	48,66%	22,15%
	15	16	5×10^{-5}	34,68%	51,44%	18,53%
VQA-RAD	30	8	5×10^{-5}	46,67%	69,05%	18,18%
	45	16	5×10^{-5}	45,78%	66,93%	18,37%
	45	8	5×10^{-5}	41,78%	71,05%	11,71%
	45	32	5×10^{-5}	41,33%	68,75%	14,16%
	45	8	1×10^{-5}	39,11%	62,81%	11,54%

Baseline model yang digunakan dalam eksperimen dengan menggunakan VGG19-LSTM adalah model yang dilatih pada *dataset* PathVQA dan VQA-RAD dengan *hyperparameter* *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} . Model ini digunakan sebagai pembanding untuk melihat apakah model yang dilatih dengan konfigurasi *hyperparameter* yang berbeda dapat menghasilkan performa yang lebih baik. Dari hasil eksperimen yang dilakukan, dapat dilihat bahwa model VGG19-LSTM yang memiliki nilai akurasi terbaik pada *dataset* PathVQA adalah ketika *epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} dengan akurasi 37,07%, akurasi *close-ended* 53,44%, dan akurasi *open-ended* 19,90%. Pada *dataset* VQA-RAD adalah ketika *epochs* 30, *batch size* 8, dan *learning rate* 5×10^{-5} dengan akurasi 46,67%, akurasi *close-ended* 69,05%, dan akurasi *open-ended* 18,18%.

Tabel 4.8. Hasil BLEU pada pertanyaan open-ended dengan VGG19-LSTM

Dataset	Epochs	Batch size	Learning rate	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	10,89	3,86	2,57
	15	8	5×10^{-5}	21,41	8,01	5,38
	45	16	5×10^{-5}	22,32	9,63	6,72
	45	8	5×10^{-5}	24,28	10,02	6,81
	15	16	5×10^{-5}	19,95	7,96	5,23
VQA-RAD	30	8	5×10^{-5}	20,17	14,34	11,32
	45	16	5×10^{-5}	20,69	16,69	11,29
	45	8	5×10^{-5}	13,08	7,45	4,74
	45	32	5×10^{-5}	15,16	11,64	7,85
	45	8	1×10^{-5}	12,26	7,23	5,51

Pada Tabel 4.8 dapat dilihat bahwa model VGG19-LSTM yang memiliki nilai BLEU terbaik pada *dataset* PathVQA adalah ketika *epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} dengan BLEU-1 24,28, BLEU-2 10,02, BLEU-3 6,81. Dari hasil ini, dapat diinterpretasikan bahwa model VGG19-LSTM ketika mengevaluasi jawaban untuk kata tunggal, model mampu menunjukkan bahwa intisari terjemahan cukup jelas meskipun terdapat kesalahan tata bahasa yang signifikan, dikarenakan nilai BLEU-1 yang didapat adalah 24,28. Sementara itu, nilai BLEU-2 yang tercatat sebesar 10,02 menandakan bahwa model sulit untuk memahami pasangan dua kata yang berurutan. Lalu, nilai BLEU-3 yang tercatat sebesar 6,81 menandakan bahwa model hampir tidak mampu memahami frasa tiga kata yang berurutan.

Berdasarkan Tabel 4.8 dapat dilihat bahwa model VGG19-LSTM yang memiliki nilai BLEU terbaik pada *dataset* VQA-RAD adalah ketika *epochs* ketika *epochs* 45, *batch size* 16, dan *learning rate* 5×10^{-5} dengan BLEU-1 20,69, BLEU-2 16,69, dan BLEU-3 11,29. Dari hasil ini, dapat diinterpretasikan bahwa model VGG19-LSTM ketika mengevaluasi jawaban untuk kata tunggal, model mampu menunjukkan bahwa intisari terjemahan cukup jelas meskipun terdapat kesalahan tata bahasa yang signifikan, dikarenakan nilai BLEU-1 yang didapat adalah 20,69. Sementara itu, nilai BLEU-2, dan BLEU-3 yang tercatat sebesar 16,69 dan 11,29 menandakan bahwa model sulit untuk memahami pasangan dua kata yang berurutan dan frasa tiga kata yang berurutan.

Dari hasil eksperimen yang dilakukan, dapat dilihat bahwa setelah dilakukan *hyperparameter tuning*, model VGG19-LSTM yang dilatih dengan *dataset* PathVQA dan VQA-RAD menghasilkan performa yang lebih baik dibandingkan dengan *baseline* model. Hal ini menunjukkan bahwa konfigurasi *hyperparameter* yang tepat dapat

meningkatkan performa model dalam memprediksi jawaban dari pertanyaan yang diberikan.

Eksperimen yang dilakukan dengan BLIP dengan *dataset* PathVQA dan VQA-RAD didapatkan hasil untuk pengujian pada pertanyaan yang bersifat *close-ended* dapat dilihat pada Lampiran 7, sedangkan untuk pengujian pada pertanyaan yang bersifat *open-ended* dapat dilihat pada Lampiran 8, dan untuk pengujian pada keseluruhan pertanyaan dapat dilihat pada Lampiran 9. Dari hasil eksperimen tersebut, dapat dilihat bahwa beberapa hasil dari model BLIP dengan *dataset* PathVQA dan VQA-RAD dapat dilihat pada Tabel 4.9. Tabel 4.10 disajikan informasi mengenai hasil BLEU pada pertanyaan *open-ended* dengan BLIP. Tabel 4.10 menunjukkan beberapa hasil terbaik dari model BLIP dengan *dataset* PathVQA dan VQA-RAD berdasarkan nilai BLEU-1, BLEU-2, dan BLEU-3 untuk melihat kualitas jawaban yang bersifat *open-ended* yang dihasilkan oleh model.

Tabel 4.9. Hasil akurasi pelatihan dengan BLIP

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	Akurasi <i>close ended</i>	Akurasi <i>open ended</i>
PathVQA	15	8	1×10^{-5}	52,94%	83,44%	22,74%
	15	16	5×10^{-5}	51,84%	82,85%	19,42%
	30	32	5×10^{-5}	48,50%	86,00%	11,13%
	45	32	5×10^{-5}	48,44%	81,70%	14,89%
	15	32	5×10^{-5}	47,03%	79,25%	15,12%
VQA-RAD	15	8	1×10^{-5}	39,73%	61,54%	39,64%
	45	8	5×10^{-5}	37,33%	56,67%	15,24%
	15	16	5×10^{-5}	32,00%	51,24%	9,62%
	30	8	5×10^{-5}	32,00%	52,85%	6,86%
	15	8	5×10^{-5}	30,67%	56,90%	2,75%

Baseline model yang diterapkan untuk melatih model dengan *dataset* PathVQA dan VQA-RAD dengan BLIP adalah model yang dilatih dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} . Tabel 4.9 menunjukkan bahwa model BLIP yang memiliki nilai akurasi terbaik pada *dataset* PathVQA adalah ketika *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} dengan akurasi 52.94%, akurasi *close-ended* 83.44%, dan akurasi *open-ended* 22.74%. Pada *dataset* VQA-RAD adalah ketika *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} dengan akurasi 39.73%, akurasi *close-ended* 61.54%, dan akurasi *open-ended* 39.64%.

Tabel 4.10. Hasil BLEU pada pertanyaan open-ended dengan BLIP

Dataset	Epochs	Batch size	Learning rate	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	26,11	11,69	8
	15	16	5×10^{-5}	21,78	11,37	7,67
	30	32	5×10^{-5}	14,19	7,8	5,8
	45	32	5×10^{-5}	17,68	9,6	6,68
	15	32	5×10^{-5}	18,05	9,48	6,69
VQA-RAD	15	8	1×10^{-5}	43,13	30,54	22,58
	45	8	5×10^{-5}	18,11	9,48	5,88
	15	16	5×10^{-5}	14,28	7,92	5,6
	30	8	5×10^{-5}	9,36	3,6	2,3
	15	8	5×10^{-5}	5,86	2,33	1,49

Pada Tabel 4.10 dapat dilihat bahwa model BLIP yang memiliki nilai BLEU terbaik pada *dataset* PathVQA adalah ketika *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} dengan BLEU-1 26,11, BLEU-2 11,69, BLEU-3 8,00. Dari hasil ini, dapat diinterpretasikan bahwa model BLIP ketika mengevaluasi jawaban untuk kata tunggal, model mampu menunjukkan bahwa intisari terjemahan cukup jelas meskipun terdapat kesalahan tata bahasa yang signifikan, dikarenakan nilai BLEU-1 yang didapat adalah 26,11. Sementara itu, nilai BLEU-2 yang tercatat sebesar 11,69 menandakan bahwa model sulit untuk memahami pasangan dua kata yang berurutan. Lalu, nilai BLEU-3 yang tercatat sebesar 8,00 menandakan bahwa model hampir tidak mampu memahami frasa tiga kata yang berurutan.

Berdasarkan Tabel 4.10 model BLIP ketika mengevaluasi *dataset* VQA-RAD yang memiliki nilai BLEU terbaik adalah ketika *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} dengan BLEU-1 43,13, BLEU-2 30,54, BLEU-3 22,58. Dari hasil ini, dapat diinterpretasikan bahwa model BLIP ketika mengevaluasi jawaban untuk kata tunggal mampu menunjukkan bahwa hasil yang diberikan berkualitas tinggi, dikarenakan nilai BLEU-1 yang didapat adalah 43,13. Sementara itu, model BLIP ini juga mampu dianggap untuk memahami pasangan dua kata yang berurutan, hal ini dikarenakan nilai BLEU-2 yang tercatat sebesar 30,54. Lalu, nilai BLEU-3 yang tercatat sebesar 22,58 menandakan bahwa model mampu menunjukkan bahwa intisari terjemahan cukup jelas meskipun terdapat kesalahan tata bahasa yang signifikan pada frasa tiga kata yang berurutan.

Setelah dilakukan *hyperparameter tuning*, model yang dikembangkan tidak menunjukkan peningkatan kinerja yang signifikan dibandingkan dengan *baseline*. Hasil

ini mengindikasikan bahwa peningkatan nilai *hyperparameter* seperti *learning rate*, *batch size*, dan jumlah *epochs* tidak selalu berbanding lurus dengan peningkatan performa. Dalam beberapa kasus, nilai *hyperparameter* yang lebih tinggi dapat justru menghambat proses pembelajaran atau menyebabkan model terlalu cepat konvergen pada solusi yang suboptimal, yang tidak efektif untuk generalisasi pada data baru. Hal ini menegaskan pentingnya pemilihan *hyperparameter* yang tepat, yang memerlukan eksperimen yang cermat dan evaluasi berkelanjutan terhadap pengaruhnya terhadap kinerja model.

Pada hasil eksperimen yang dilakukan dengan *dataset* PathVQA dan VQA-RAD yang telah dilakukan augmentasi pertanyaan dengan menciptakan tujuh pertanyaan yang diparafrase dari setiap pertanyaan asli dapat dilihat pada Tabel 4.11 dan Tabel 4.12.

Tabel 4.11. Hasil akurasi setelah dilakukan augmentasi data pertanyaan

Model	Dataset	Epochs	Batch size	Learning rate	Akurasi	Akurasi close ended	Akurasi open ended
VGG19-LSTM	PathVQA	15	8	1×10^{-5}	25,87%	41,73%	0,10%
BLIP		15	8	1×10^{-5}	83,91%	97,43%	66,15%
VGG19-LSTM	VQA-RAD	15	8	1×10^{-5}	41,64%	62,40%	16,54%
BLIP		15	8	5×10^{-5}	66,83%	79,41%	51,62%
BLIP	VQA-RAD	15	8	1×10^{-5}	64,73%	85,55%	39,57%
BLIP		30	8	1×10^{-5}	77,33%	78,39%	76,04%
BLIP	VQA-RAD	30	8	5×10^{-5}	70,89%	84,02%	55,02%
BLIP		45	8	1×10^{-5}	59,90%	64,19%	54,71%
BLIP	VQA-RAD	45	8	5×10^{-5}	82,86%	87,85%	76,82%

Tabel 4.11 menunjukkan hasil akurasi setelah dilakukan augmentasi data pertanyaan dengan menciptakan tujuh pertanyaan yang diparafrase dari setiap pertanyaan asli. Dari hasil eksperimen yang dilakukan, dapat dilihat bahwa model BLIP yang dilatih dengan *dataset* PathVQA dan VQA-RAD menghasilkan performa yang lebih baik dibandingkan dengan model VGG19-LSTM. Model BLIP yang dilatih dengan *dataset* PathVQA dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan akurasi 83,91%, akurasi *close-ended* 97,43%, dan akurasi *open-ended* 66,15%. Model BLIP yang dilatih dengan *dataset* VQA-RAD dengan *hyperparameter epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} menghasilkan akurasi 82,86%, akurasi *close-ended* 87,85%, dan akurasi *open-ended* 76,82%.

Tabel 4.12. Hasil nilai BLEU setelah dilakukan augmentasi data pertanyaan

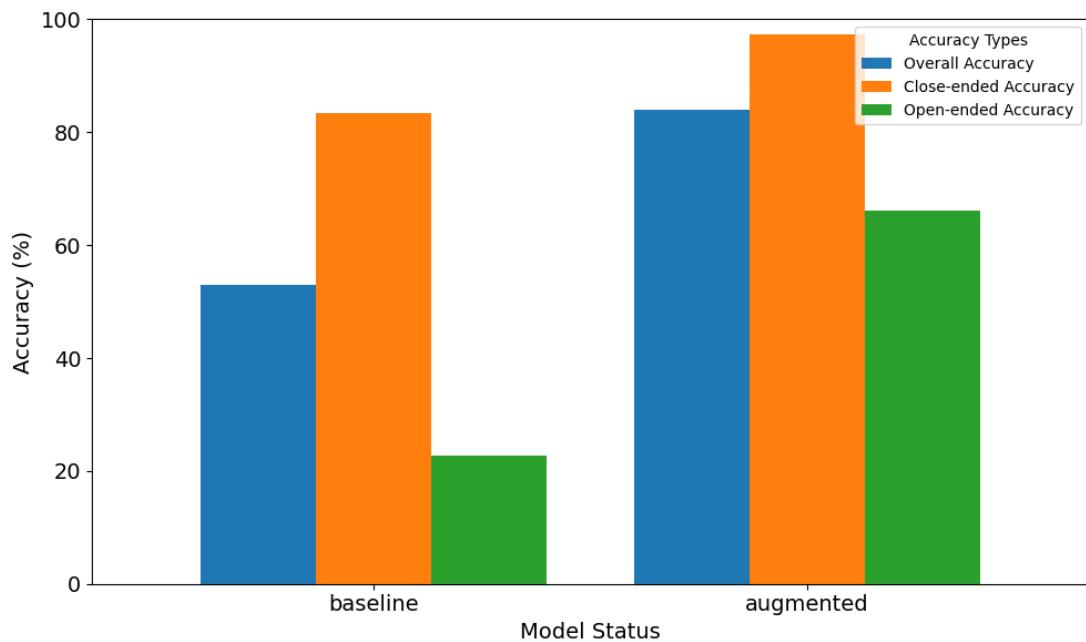
Model	Dataset	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	BLEU-1	BLEU-2	BLEU-3
VGG19-LSTM	PathVQA	15	8	1×10^{-5}	0,19	0,06	0,05
BLIP		15	8	1×10^{-5}	69,6	49,8	39,87
VGG19-LSTM	VQA-RAD	15	8	1×10^{-5}	17,17	10,5	7,39
BLIP		15	8	5×10^{-5}	54,36	35,09	25,55
BLIP		15	8	1×10^{-5}	43,33	29,21	20,33
BLIP		30	8	1×10^{-5}	78,12	53,63	40,35
BLIP		30	8	5×10^{-5}	58,72	36,26	26,4
BLIP		45	8	1×10^{-5}	57,25	36,89	26,29
BLIP		45	8	5×10^{-5}	80,19	53,36	40,47

Selanjutnya pada Tabel 4.12 menunjukkan hasil nilai BLEU untuk pertanyaan *open-ended* setelah dilakukan augmentasi data pertanyaan dengan menciptakan tujuh pertanyaan yang diparafrase dari setiap pertanyaan asli. Dari hasil eksperimen yang dilakukan, dapat dilihat bahwa model BLIP yang dilatih dengan *dataset* PathVQA dan VQA-RAD menghasilkan performa yang lebih baik dibandingkan dengan model VGG19-LSTM. Model BLIP yang dilatih dengan *dataset* PathVQA dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan BLEU-1 69,6, BLEU-2 49,8, dan BLEU-3 39,87. Hal ini menunjukkan bahwa kemampuan model BLIP untuk mengevaluasi jawaban untuk kata tunggal lebih bagus dari manusia, dikarenakan nilai BLEU-1 yang didapat adalah 69,6. Sementara itu, nilai BLEU-2 yang tercatat sebesar 49,8 menandakan bahwa model memiliki kemampuan berkualitas tinggi untuk memahami pasangan dua kata yang berurutan. Lalu, nilai BLEU-3 yang tercatat sebesar 39,87 menandakan bahwa model mampu memahami frasa tiga kata yang berurutan dengan baik.

Berdasarkan Tabel 4.12 model BLIP yang dilatih dengan *dataset* VQA-RAD dengan *hyperparameter epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} menghasilkan BLEU-1 80,19, BLEU-2 53,36, dan BLEU-3 40,47. Hal ini menunjukkan bahwa kemampuan model BLIP untuk mengevaluasi jawaban untuk kata tunggal lebih bagus dari manusia, dikarenakan nilai BLEU-1 yang didapat adalah 80,19. Sementara itu, nilai BLEU-2 yang tercatat sebesar 53,36 menandakan bahwa model memiliki kemampuan berkualitas tinggi dan memadai untuk memahami pasangan dua kata yang berurutan. Lalu, nilai BLEU-3 yang tercatat sebesar 40,47 menandakan bahwa model memiliki kemampuan berkualitas tinggi dalam memahami frasa tiga kata yang berurutan.

Dari hasil eksperimen yang dilakukan, dapat dilihat bahwa model BLIP yang

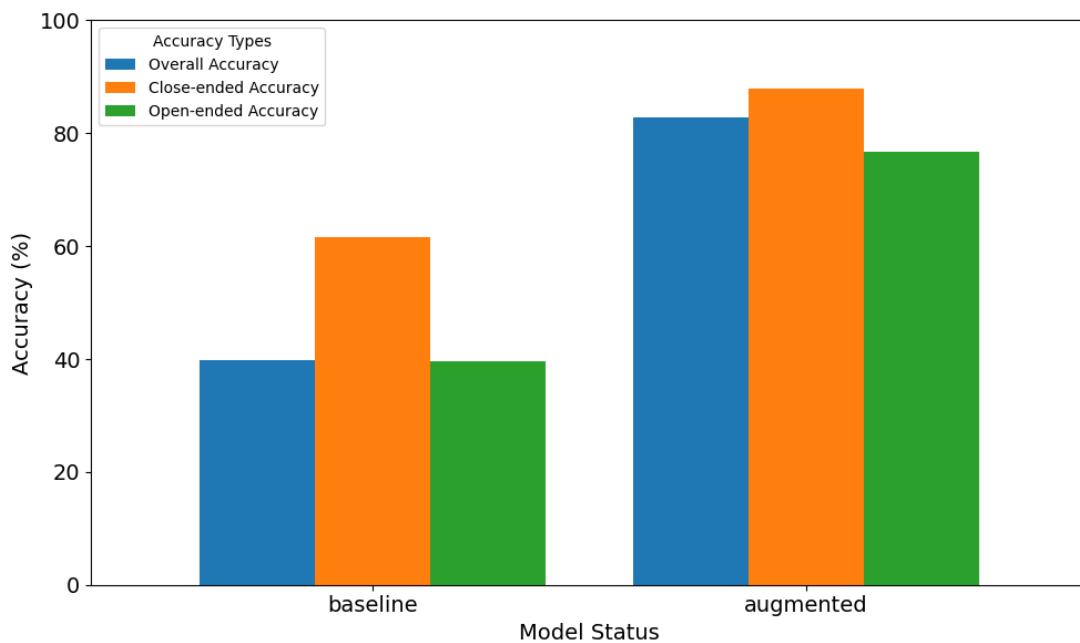
dilatih dengan *dataset* PathVQA yang sudah dilakukan augmentasi data pertanyaan menghasilkan performa yang baik. Pada hal ini dikatakan lebih baik setelah dilakukan augmentasi karena meningkatnya nilai akurasi, akurasi *close-ended*, dan akurasi *open-ended* dari *baseline* model yang sebelumnya dilatih tanpa augmentasi data pertanyaan dengan konfigurasi *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan akurasi 52,94%, akurasi *close-ended* 83,44%, dan akurasi *open-ended* 22,74%. Pada *dataset* PathVQA yang telah dilakukan augmentasi data pertanyaan, model BLIP menghasilkan akurasi 83,91%, akurasi *close-ended* 97,43%, dan akurasi *open-ended* 66,15% dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} . Hal ini menunjukkan bahwa nilai akurasi meningkat sebesar 31,97%, dikarenakan akurasi *close-ended* meningkat sebesar 13,99% dan akurasi *open-ended* yang juga meningkat sebesar 43,41%. Gambar 4.4 menunjukkan perbandingan *baseline* model dengan model BLIP setelah dilakukan augmentasi data pertanyaan.



Gambar 4.4. Perbandingan akurasi model BLIP dengan *dataset* PathVQA sebelum dan sesudah dilakukan augmentasi data pertanyaan

Lalu, untuk model BLIP yang dilatih dengan *dataset* VQA-RAD juga menghasilkan performa yang lebih baik setelah dilakukan augmentasi data pertanyaan. Hal ini dapat dilihat dari peningkatan nilai akurasi, akurasi *close-ended*, dan akurasi *open-ended* dari *baseline* model yang sebelumnya dilatih tanpa augmentasi data pertanyaan dengan konfigurasi *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan akurasi 39,73%, akurasi *close-ended* 61,54%, dan akurasi *open-ended* 39,64%. Pada *dataset* VQA-RAD yang telah dilakukan augmentasi data

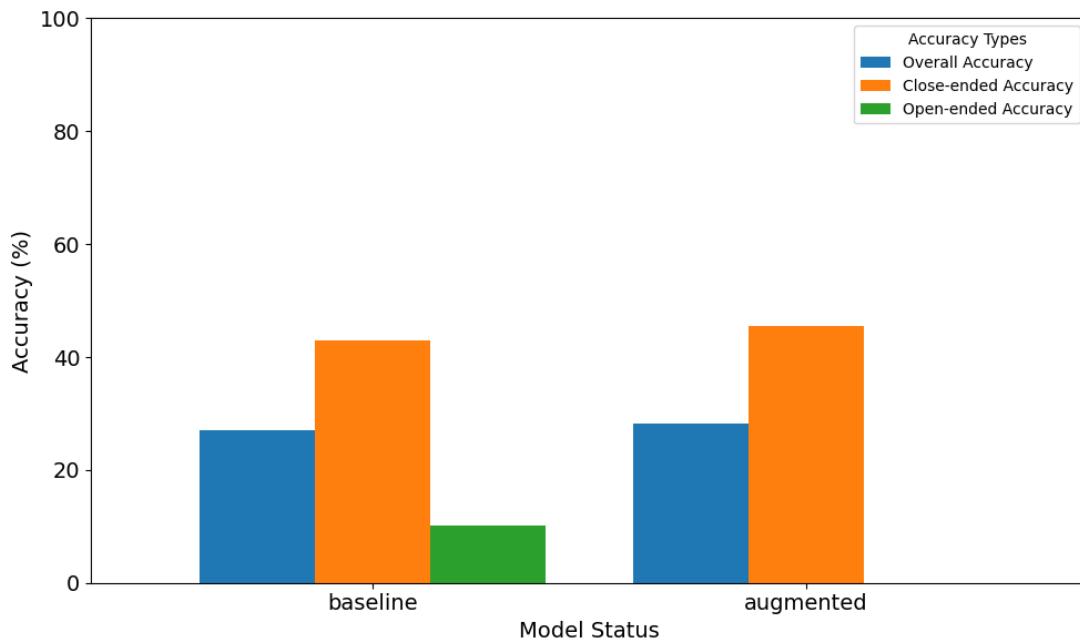
pertanyaan, model BLIP menghasilkan akurasi 82,86%, akurasi *close-ended* 87,85%, dan akurasi *open-ended* 76,82% dengan *hyperparameter epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} . Hal ini menunjukkan bahwa nilai akurasi meningkat sebesar 43,13%, dikarenakan akurasi *close-ended* meningkat sebesar 26,31% dan akurasi *open-ended* yang juga meningkat sebesar 37,18%. Pada Gambar 4.5 menunjukkan perbandingan *baseline* model dengan model BLIP setelah dilakukan augmentasi data pertanyaan.



Gambar 4.5. Perbandingan akurasi model BLIP dengan *dataset* VQA-RAD sebelum dan sesudah dilakukan augmentasi data pertanyaan

Sedangkan untuk model VGG19-LSTM yang dilatih dengan *dataset* PathVQA tidak menghasilkan performa yang lebih baik setelah dilakukan augmentasi data pertanyaan. Hal ini karena hasil yang didapatkan tidak menunjukkan peningkatan nilai akurasi, akurasi *close-ended*, dan akurasi *open-ended* dari *baseline* model yang sebelumnya dilatih tanpa augmentasi data pertanyaan dengan konfigurasi *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan akurasi 26,96%, akurasi *close-ended* 43,73%, dan akurasi *open-ended* 10,16%. Pada *dataset* PathVQA yang telah dilakukan augmentasi data pertanyaan, model VGG19-LSTM menghasilkan akurasi 25,87%, akurasi *close-ended* 41,73%, dan akurasi *open-ended* 0,10% dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} . Hal ini menunjukkan bahwa adanya penurunan nilai akurasi sebesar 1,09%, dikarenakan akurasi *close-ended* yang mengalami penurunan sebesar 2,00% dan akurasi *open-ended* yang mengalami penurunan sebesar 10,06%. Lalu, pada

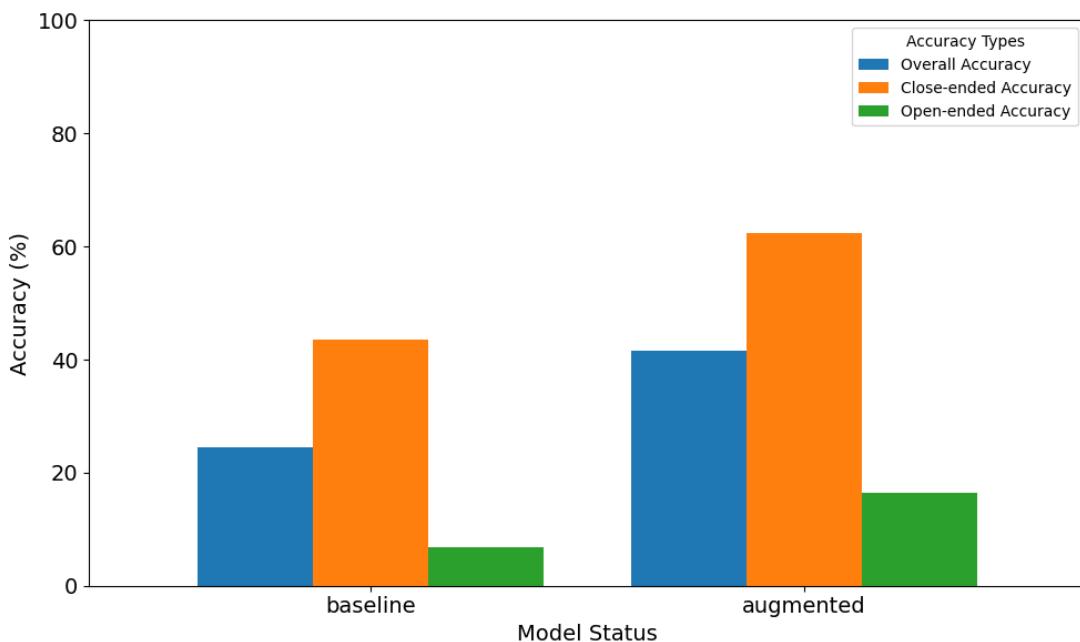
saat mengevaluasi dengan BLEU pada pertanyaan *open-ended* menunjukkan bahwa kemampuan model VGG19-LSTM yang dilatih dengan *dataset* PathVQA hampir tidak dapat dipahami dalam mengevaluasi jawaban untuk kata tunggal, pasangan dua kata yang berurutan, dan frasa tiga kata yang berurutan, dikarenakan nilai BLEU-1, BLEU-2, dan BLEU-3 yang didapat adalah 0,19, 0,06, dan 0,05. Pada Gambar 4.6 menunjukkan perbandingan *baseline* model dengan model VGG19-LSTM setelah dilakukan augmentasi data pertanyaan.



Gambar 4.6. Perbandingan akurasi model VGG19-LSTM dengan *dataset* PathVQA sebelum dan sesudah dilakukan augmentasi data pertanyaan

Lalu, untuk model VGG19-LSTM yang dilatih dengan *dataset* VQA-RAD menghasilkan performa yang baik setelah dilakukan augmentasi data pertanyaan. Walaupun hasil yang didapat tidak sebaik model BLIP, namun hasil yang didapat menunjukkan peningkatan performa setelah dilakukan augmentasi data pertanyaan. Hal ini dapat dilihat dari peningkatan nilai akurasi, akurasi *close-ended*, dan akurasi *open-ended* dari *baseline* model yang sebelumnya dilatih tanpa augmentasi data pertanyaan dengan konfigurasi *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan akurasi 24,44%, akurasi *close-ended* 43,52%, dan akurasi *open-ended* 6,84%. Pada *dataset* VQA-RAD yang telah dilakukan augmentasi data pertanyaan, model VGG19-LSTM menghasilkan akurasi 41,64%, akurasi *close-ended* 62,40%, dan akurasi *open-ended* 16,54% dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} . Hal ini menunjukkan bahwa nilai akurasi meningkat sebesar 17,20%, dikarenakan akurasi *close-ended* meningkat sebesar 18,88% dan

akurasi *open-ended* yang juga meningkat sebesar 9,70%. Ketika mengevaluasi dengan BLEU pada pertanyaan *open-ended* menunjukkan bahwa kemampuan model VGG19-LSTM yang dilatih dengan *dataset* VQA-RAD sulit untuk memahami kata tunggal dan dua kata yang berurutan, dikarenakan nilai BLEU-1 dan BLEU-2 yang didapat adalah 17,17 dan 10,5. Sementara itu, model VGG19-LSTM ini juga hampir tidak mampu memahami frasa tiga kata yang berurutan, hal ini dikarenakan nilai BLEU-3 yang tercatat sebesar 7,39. Pada Gambar 4.7 menunjukkan perbandingan *baseline* model dengan model VGG19-LSTM setelah dilakukan augmentasi data pertanyaan.

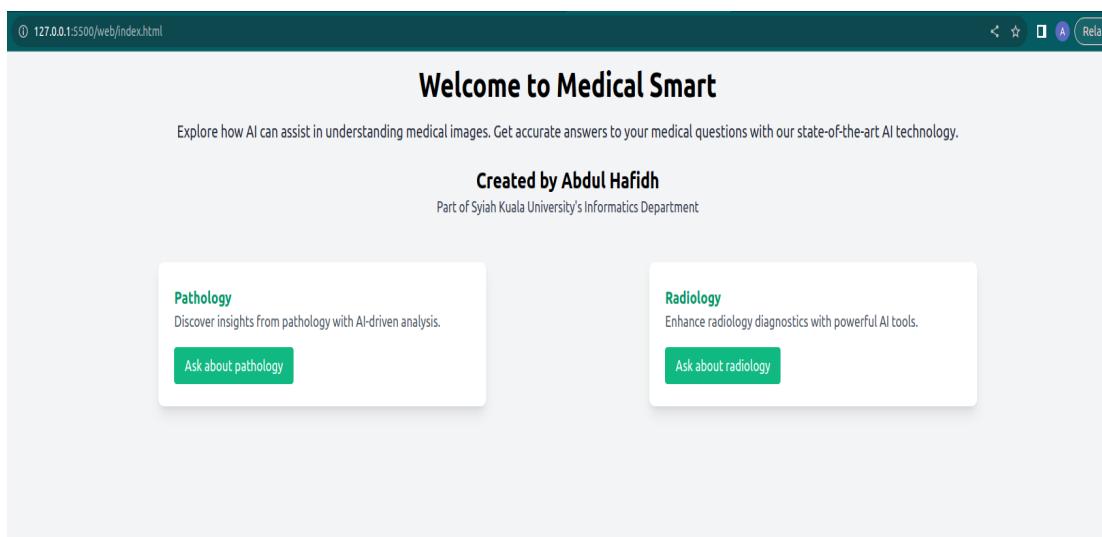


Gambar 4.7. Perbandingan akurasi model VGG19-LSTM dengan *dataset* VQA-RAD sebelum dan sesudah dilakukan augmentasi data pertanyaan

Dari semua eksperimen yang sudah dilakukan, dapat dilihat bahwa nilai akurasi untuk pertanyaan *close-ended* lebih tinggi dibandingkan dengan pertanyaan *open-ended*. Hal ini bisa saja disebabkan dikarenakan pertanyaan *close-ended* memiliki distribusi yang lebih banyak dibandingkan dengan pertanyaan *open-ended*. Lalu, dapat disimpulkan juga bahwa model BLIP yang dilatih dengan *dataset* PathVQA dan VQA-RAD menghasilkan performa yang lebih baik dibandingkan dengan model VGG19-LSTM. Hal ini menunjukkan bahwa model BLIP yang dilatih dengan *dataset* PathVQA dan VQA-RAD dapat digunakan untuk menjawab pertanyaan berdasarkan citra patologi dan radiologi dengan performa yang lebih baik dibandingkan dengan model VGG19-LSTM.

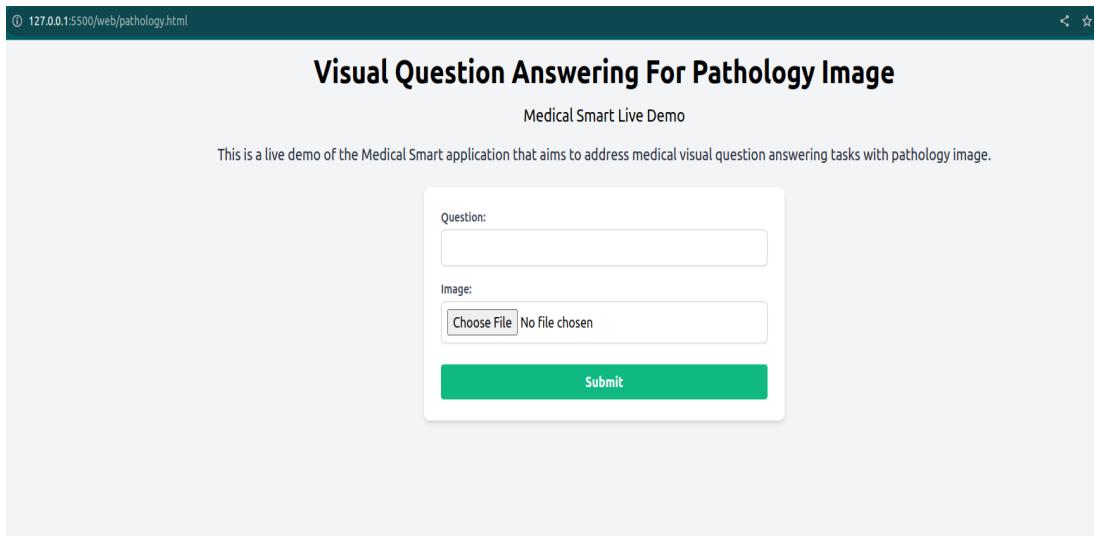
4.5 PENGEMBANGAN SISTEM MEDIS CERDAS BERBASIS WEB

Pada pengembangan sistem medis cerdas berbasis web ini meliputi tiga halaman, yaitu halaman utama, halaman untuk menjawab pertanyaan berdasarkan citra patologi, dan halaman untuk menjawab pertanyaan berdasarkan citra radiologi. Pada halaman utama, pengguna dapat memilih jenis citra yang diinginkan, yaitu citra patologi atau citra radiologi. Halaman utama ini dapat dilihat pada Gambar 4.8. Halaman utama ini akan mengarahkan pengguna ke halaman untuk menjawab pertanyaan berdasarkan citra patologi atau radiologi sesuai dengan pilihan pengguna.



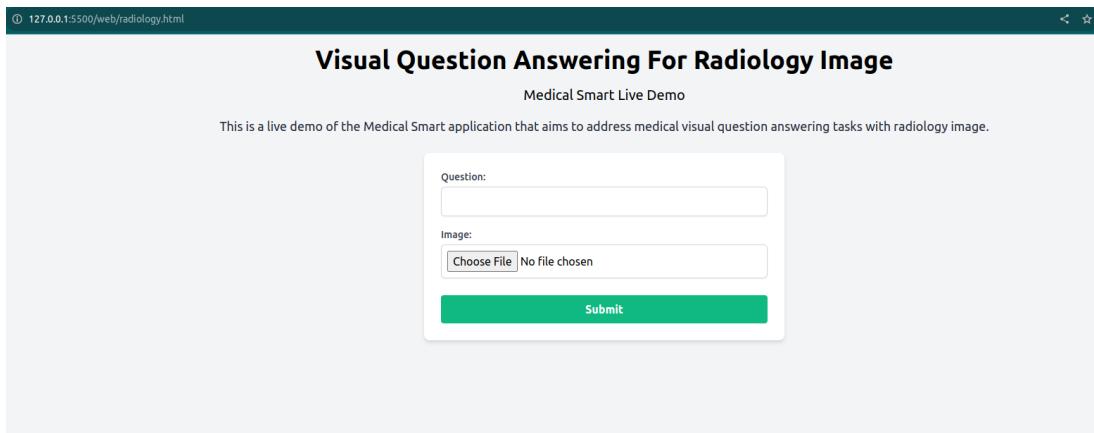
Gambar 4.8. Halaman utama sistem medis cerdas berbasis web

Pada halaman untuk menjawab pertanyaan berdasarkan citra patologi, pengguna dapat mengunggah citra patologi dan menuliskan pertanyaan yang ingin diajukan. Halaman ini akan menampilkan jawaban dari pertanyaan yang diajukan berdasarkan citra patologi yang diunggah. Halaman ini dapat dilihat pada Gambar 4.9.



Gambar 4.9. Halaman untuk menjawab pertanyaan berdasarkan citra patologi

Pada halaman untuk menjawab pertanyaan berdasarkan citra radiologi, pengguna dapat mengunggah citra radiologi dan menuliskan pertanyaan yang ingin diajukan. Halaman ini akan menampilkan jawaban dari pertanyaan yang diajukan berdasarkan citra radiologi yang diunggah. Halaman ini dapat dilihat pada Gambar 4.10.



Gambar 4.10. Halaman untuk menjawab pertanyaan berdasarkan citra radiologi

Model yang digunakan dalam sistem ini adalah model BLIP, dikarenakan model BLIP yang dilatih dengan *dataset* PathVQA dan VQA-RAD yang telah dilakukan augmentasi data pertanyaan menghasilkan performa yang lebih baik dibandingkan dengan model VGG19-LSTM. Model BLIP yang dilatih dengan *dataset* PathVQA yang telah dilakukan augmentasi data pertanyaan dengan *hyperparameter epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} akan digunakan untuk menjawab pertanyaan berdasarkan citra patologi. Model BLIP yang dilatih dengan *dataset* VQA-RAD yang telah dilakukan augmentasi data pertanyaan dengan *hyperparameter epochs* 45, *batch*

size 8, dan *learning rate* 5×10^{-5} akan digunakan untuk menjawab pertanyaan berdasarkan citra radiologi.

Setelah memilih model yang akan digunakan, disini akan dilakukan tahap kuantisasi model yang bertujuan dalam memperkecil ukuran model tanpa mengorbankan performa dan mempercepat proses inferensi. Pada tahap kuantisasi model ini akan dilakukan dengan menggunakan metode *dynamic quantization* yang disediakan oleh *framework* PyTorch untuk tahapan kuantisasi ini. Pada Tabel 4.13 menunjukkan spesifikasi model BLIP sebelum dan sesudah dikuantisasi pada *dataset* PathVQA dan VQA-RAD untuk membandingkan ukuran model dan rata-rata waktu inferensi.

Tabel 4.13. Spesifikasi model BLIP sebelum dan sesudah dikuantisasi

<i>Dataset</i>	Status	Ukuran model	Rata-rata waktu inferensi
PathVQA	Tidak dikuantisasi	1467,73 MB	0,24 detik
	Sudah dikuantisasi	508,21 MB	0,20 detik
VQA-RAD	Tidak dikuantisasi	1467,73 MB	0,21 detik
	Sudah dikuantisasi	508,20 MB	0,17 detik

Pada proses pengembangan *backend* pada sistem ini menggunakan *framework* FastAPI untuk dilakukan *deploy* model yang telah dikuantisasi menjadi sebuah API. Disini dipilih model yang telah dikuantisasi karena ukuran model yang lebih kecil dan rata-rata waktu inferensi yang lebih cepat dibandingkan dengan model yang tidak dikuantisasi sesuai dengan Tabel 4.13. Tabel 4.14 menunjukkan daftar API *endpoint* pada sistem medis cerdas berbasis web yang digunakan untuk menjawab pertanyaan berdasarkan citra patologi dan radiologi.

Tabel 4.14. Daftar API *endpoint* pada sistem medis cerdas berbasis web

No	Method	Endpoint	Deskripsi
1	POST	/predict/pathology	Menjawab pertanyaan berdasarkan citra patologi
2		/predict/radiology	Menjawab pertanyaan berdasarkan citra radiologi

Tabel 4.14 memaparkan daftar API *endpoint* pada sistem medis cerdas berbasis web, yang dirancang untuk menjawab pertanyaan yang berkaitan dengan citra patologi dan radiologi. Kedua *endpoint* ini diakses melalui *method* "POST" dan menerima masukan berupa gambar serta pertanyaan yang diinginkan. Hasil yang diberikan adalah jawaban terhadap pertanyaan tersebut dalam format JSON.

Dalam konteks sistem ini, ketika pengguna menekan tombol "Submit" pada halaman yang ditujukan untuk pertanyaan berbasis citra patologi atau radiologi, akan terjadi *request* ke API *endpoint* yang relevan. Dalam hal ini, *request* ini mengirimkan gambar bersama dengan pertanyaan yang akan dijawab. Setelah *request* diterima oleh *endpoint*, sistem akan melakukan inferensi menggunakan model yang telah dikuantisasi untuk memproses data tersebut. Hasil inferensi berupa lima jawaban yang mungkin benar akan disimpan dalam format JSON, kemudian hasil dari JSON akan dikirim ke halaman pengguna untuk ditampilkan. Pada saat yang sama, hasil jawaban akan diberi *confidence score* yang menunjukkan tingkat keyakinan model terhadap jawaban yang diberikan.

Setelah pengembangan *backend* selesai, maka proses percobaan untuk menjawab pertanyaan berdasarkan citra patologi dan radiologi dapat dilakukan. Pada uji coba dalam menjawab pertanyaan berdasarkan citra patologi dapat dilihat pada Lampiran 10, sedangkan pada uji coba dalam menjawab pertanyaan berdasarkan citra radiologi dapat dilihat pada Lampiran 11.

BAB V

KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Berdasarkan hasil eksperimen dan perbandingan terhadap arsitektur VGG19-LSTM dan BLIP pada *Visual Question Answering* (VQA) dengan *dataset* PathVQA dan VQA-RAD, diperoleh kesimpulan sebagai berikut:

1. Penelitian ini berhasil menerapkan teknik *transfer learning* untuk membangun model VQA dengan menggunakan arsitektur VGG19-LSTM dan BLIP. Arsitektur VGG19-LSTM menggunakan model VGG19 yang telah dilatih pada *dataset* ImageNet untuk ekstraksi fitur citra medis dan model LSTM untuk memproses pertanyaan dari *dataset* PathVQA dan VQA-RAD. Sedangkan BLIP memanfaatkan *pre-trained* model pada tugas VQA yang dilatih dengan *dataset* Visual Genome dan VQA v2.0 untuk memproses citra medis dan pertanyaan dari *dataset* PathVQA dan VQA-RAD.
2. Model BLIP dengan konfigurasi *hyperparameter*: *epochs* 15, *batch size* 8, dan *learning rate* 1×10^{-5} menghasilkan performa terbaik pada *dataset* PathVQA yang telah dilakukan augmentasi pertanyaan dengan menciptakan tujuh variasi pertanyaan yang baru dengan akurasi 83,91%, akurasi *close-ended* 97,43%, dan akurasi *open-ended* 66,15%. Sedangkan untuk nilai BLEU-1, BLEU-2, dan BLEU-3 berturut-turut sebesar 69,6, 49,8, 39,87 untuk pertanyaan *open-ended*.
3. Model BLIP dengan konfigurasi *hyperparameter*: *epochs* 45, *batch size* 8, dan *learning rate* 5×10^{-5} menghasilkan performa terbaik pada *dataset* VQA-RAD yang telah dilakukan augmentasi pertanyaan dengan menciptakan tujuh variasi pertanyaan yang baru dengan akurasi 82,86%, akurasi *close-ended* 87,85%, dan akurasi *open-ended* 76,82%. Sedangkan untuk nilai BLEU-1, BLEU-2, dan BLEU-3 berturut-turut sebesar 80,19, 53,36, 40,47 untuk pertanyaan *open-ended*.
4. Model VGG19-LSTM menunjukkan keterbatasan dalam menyelesaikan tugas VQA, hal ini terlihat dari hasil eksperimen yang dilakukan pada kedua *dataset* PathVQA dan VQA-RAD. Walaupun telah dilakukan augmentasi data pada kedua *dataset* dan *hyperparameter tuning* pada model VGG19-LSTM, performa model tersebut masih jauh dari nilai yang optimal dibandingkan dengan model BLIP. Oleh karena itu, model BLIP akan diimplementasikan ke dalam sistem medis cerdas berbasis web untuk menjawab pertanyaan berdasarkan citra medis.

5. Model BLIP yang telah dikuantisasi menunjukkan performa yang lebih baik dibandingkan tanpa kuantisasi. Untuk *dataset* PathVQA, ukuran model pra-kuantisasi adalah 1467,73 MB dengan waktu inferensi 0,24 detik, sedangkan pasca-kuantisasi adalah 508,21 MB dengan waktu inferensi 0,20 detik. Untuk *dataset* VQA-RAD, ukuran model pra-kuantisasi adalah 1467,73 MB dengan waktu inferensi 0,21 detik, dan pasca-kuantisasi adalah 508,20 MB dengan waktu inferensi 0,17 detik. Oleh karena itu, model BLIP yang telah dikuantisasi akan diimplementasikan ke dalam sistem medis cerdas berbasis web agar mempercepat waktu inferensi pada saat proses inferensi berlangsung.

5.2 SARAN

Berdasarkan kesimpulan yang telah dijelaskan sebelumnya, maka penulis memberikan saran untuk penelitian selanjutnya sebagai berikut:

1. Mengimplementasikan metode *embedding* yang lainnya seperti *word2vec* atau *fastText* pada model VGG19-LSTM untuk melihat apakah performa model tersebut dapat ditingkatkan.
2. Melakukan eksperimen dengan menggunakan *pre-trained* model lainnya dengan basis *transformer* seperti *Vision Language Transformers* (ViLT), *Vision-and-Language BERT* (ViLBERT), *Learning Cross-Modality Encoder Representations from Transformers* (LXMERT), dan lain-lain pada tugas VQA.
3. Menerapkan teknik augmentasi data VQA yang lebih kompleks dengan mengimplementasikan model generatif untuk menciptakan gambar dan pertanyaan baru, Sehingga berkemungkinan dapat meningkatkan performa model VQA.

DAFTAR PUSTAKA

- Agarwal, M. & Saxena, A. (2019). An overview of natural language processing. *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, 7(5):2811–2813.
- Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidiqe, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *electronics*, 8(3):292.
- Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., & Farhan, L. (2021). Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8:1–74.
- Andono, P. N., Sutojo, T., et al. (2018). *Pengolahan citra digital*. Penerbit Andi.
- Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., & Parikh, D. (2015). Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.
- Ardiansyah, R. F. (2013). Pengenalan pola tanda tangan dengan menggunakan metode principal component analysis (pca). *Fakultas Ilmu Komputer Universitas Dian Nuswantoro*, 2:14.
- Basu, K., Sinha, R., Ong, A., & Basu, T. (2020). Artificial intelligence: How is it changing medical sciences and its future? *Indian journal of dermatology*, 65(5):365.
- Bazi, Y., Rahhal, M. M. A., Bashmal, L., & Zuair, M. (2023). Vision–language model for visual question answering in medical imagery. *Bioengineering*, 10(3):380.
- Bergstra, J. & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).
- Brownlee, J. (2017). A gentle introduction to long short-term memory networks for the experts. *Machine Learning Mastery*.
- Cireşan, D. C., Meier, U., & Schmidhuber, J. (2012). Transfer learning for latin and chinese characters with deep neural networks. In *The 2012 international joint conference on neural networks (IJCNN)*, pages 1–6. IEEE.
- Fernando, D. & Harsiti, H. (2019). Studi literatur: Robotic process automation. *JSiI (Jurnal Sistem Informasi)*, 6(1):6–11.
- Gurney, K. (1997). *An Introduction to Neural Networks*. Taylor & Francis, Inc., USA.
- Hendrycks, D. & Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

- Henningsen-Schomers, M. R. & Pulvermüller, F. (2022). Modelling concrete and abstract concepts using brain-constrained deep neural networks. *Psychological research*, 86(8):2533–2559.
- Hildebrandt, M., Li, H., Koner, R., Tresp, V., & Günemann, S. (2020). Scene graph reasoning for visual question answering. *arXiv preprint arXiv:2007.01072*.
- Ho, Y. & Wookey, S. (2019). The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. *IEEE access*, 8:4806–4813.
- Hossin, M. & Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1.
- Huang, D., Fu, Y., Qin, N., & Gao, S. (2021). Fault diagnosis of high-speed train bogie based on lstm neural network. *Sci. Chin. Inf. Sci*, 64:1–3.
- Ilahiyah, S. & Nilogiri, A. (2018). Implementasi deep learning pada identifikasi jenis tumbuhan berdasarkan citra daun menggunakan convolutional neural network. *JUSTINDO (Jurnal Sistem Dan Teknologi Informasi Indonesia)*, 3(2):49–56.
- Jähne, B., Haussecker, H., & Geissler, P. (1999). *Handbook of computer vision and applications*, volume 2. Citeseer.
- Kelleher, J. D. (2019). *Deep learning*. MIT press.
- Khan, A., Sohail, A., Zahoor, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, 53:5455–5516.
- Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kristiyanti, D. A. & Saputra, I. (2023). Machine learning untuk pemula.
- Kulkarni, A., Chong, D., & Batarseh, F. A. (2020). 5 - foundations of data imbalance and solutions for a data democracy. In Batarseh, F. A. & Yang, R., editors, *Data Democracy*, pages 83–106. Academic Press.
- Kusuma, A. W. & Ellyana, R. L. (2018). Penerapan citra terkompresi pada segmentasi citra menggunakan algoritma k-means. *Jurnal Terapan Teknologi Informasi*, 2(1):65–74.
- Kusuma, I. W. A. W. & Kusumadewi, A. (2020). Penerapan metode contrast stretching, histogram equalization dan adaptive histogram equalization untuk meningkatkan kualitas citra medis mri. *Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, 11(1):1–10.
- Lau, J. J., Gayen, S., Ben Abacha, A., & Demner-Fushman, D. (2018). A dataset of clinically generated visual questions and answers about radiology images. *Scientific data*, 5(1):1–10.

- Lavie, A. (2010). Evaluating the output of machine translation systems. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Tutorials*.
- Li, J., Li, D., Xiong, C., & Hoi, S. (2022). Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR.
- Li, K. & Malik, J. (2017). Learning to optimize neural nets. *arXiv preprint arXiv:1703.00441*.
- Li, M., Cai, W., Liu, R., Weng, Y., Zhao, X., Wang, C., Chen, X., Liu, Z., Pan, C., Li, M., et al. (2021). Ffa-ir: Towards an explainable and reliable medical report generation benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- Lin, Z., Zhang, D., Tao, Q., Shi, D., Haffari, G., Wu, Q., He, M., & Ge, Z. (2023). Medical visual question answering: A survey. *Artificial Intelligence in Medicine*, page 102611.
- Luo, N., Zhong, X., Su, L., Cheng, Z., Ma, W., & Hao, P. (2023). Artificial intelligence-assisted dermatology diagnosis: From unimodal to multimodal. *Computers in Biology and Medicine*, 165:107413.
- Mauli, D. (2018). Tanggung jawab hukum dokter terhadap kesalahan diagnosis penyakit kepada pasien. *Cepalo*, 2(1):33–42.
- Monedero, I., Barbancho, J., Márquez, R., & Beltrán, J. F. (2021). Cyber-physical system for environmental monitoring based on deep learning. *Sensors*, 21(11):3655.
- Mukhlif, A. A., Al-Khateeb, B., & Mohammed, M. A. (2023). Incorporating a novel dual transfer learning approach for medical images. *Sensors*, 23(2):570.
- Nguyen, B. D., Do, T.-T., Nguyen, B. X., Do, T., Tjiputra, E., & Tran, Q. D. (2019). Overcoming data limitations in medical visual question answering. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*.
- Nguyen, T.-H., Nguyen, T.-N., & Ngo, B.-V. (2022). A vgg-19 model with transfer learning and image segmentation for classification of tomato leaf disease. *AgriEngineering*, 4(4):871–887.
- Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*.
- Pan, S. J. & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

- Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.
- Parai, A., Deshpande, S., Iyer, A., Kumbhare, A., & Bendale, S. (2022). Predictive cancer detection and treatment using machine learning and artificial intelligence.
- Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Puttagunta, M. & Ravi, S. (2021). Medical image analysis based on deep learning approach. *Multimedia tools and applications*, 80:24365–24398.
- Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Sorace, J., Aberle, D. R., Elimam, D., Lawvere, S., Tawfik, O., & Wallace, W. D. (2012). Integrating pathology and radiology disciplines: an emerging opportunity? *BMC medicine*, 10(1):1–6.
- Srivastava, Y., Murali, V., Dubey, S. R., & Mukherjee, S. (2021). Visual question answering using deep learning: A survey and performance analysis. In *Computer Vision and Image Processing: 5th International Conference, CVIP 2020, Prayagraj, India, December 4-6, 2020, Revised Selected Papers, Part II 5*, pages 75–86. Springer.
- Teller, V. (2000). Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition.
- Teney, D., Wu, Q., & van den Hengel, A. (2017). Visual question answering: A tutorial. *IEEE Signal Processing Magazine*, 34(6):63–75.
- Tschandl, P., Rinner, C., Apalla, Z., Argenziano, G., Codella, N., Halpern, A., Janda, M., Lallas, A., Longo, C., Malvehy, J., et al. (2020). Human–computer collaboration for skin cancer recognition. *Nature Medicine*, 26(8):1229–1234.
- Vardhan, J. & Swetha, K. S. (2023). Detection of healthy and diseased crops in drone captured images using deep learning. *arXiv preprint arXiv:2305.13490*.
- Wardani, K. R. & Leonardi, L. (2023). Klasifikasi penyakit pada daun anggur menggunakan metode convolutional neural network. *Jurnal Tekno Insentif*, 17(2):112–126.
- Xuehai, H., Yichen, Z., Luntian, M., Eric, X., & Pengtao, X. (2020). Pathvqa: 30000+ questions for medical visual question answering. *arXiv preprint arXiv:2003.10286*.
- Yu, T. & Zhu, H. (2020). Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*.
- Zhang, Z. (2016). Neural networks: further insights into error function, generalized weights and others. *Annals of translational medicine*, 4(16).

Lampiran 2. Layer text encoder pada model BLIP

```
● ● ●  
Layer: text_encoder  
BlipTextModel(  
    (embeddings): BlipTextEmbeddings(  
        (word_embeddings): Embedding(30524, 768, padding_idx=0)  
        (position_embeddings): Embedding(512, 768)  
        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)  
        (dropout): Dropout(p=0.0, inplace=False)  
    )  
    (encoder): BlipTextEncoder(  
        (layer): ModuleList(  
            (0-11): 12 x BlipTextLayer(  
                (attention): BlipTextAttention(  
                    (self): BlipTextSelfAttention(  
                        (query): Linear(in_features=768, out_features=768, bias=True)  
                        (key): Linear(in_features=768, out_features=768, bias=True)  
                        (value): Linear(in_features=768, out_features=768, bias=True)  
                        (dropout): Dropout(p=0.0, inplace=False)  
                    )  
                    (output): BlipTextSelfOutput(  
                        (dense): Linear(in_features=768, out_features=768, bias=True)  
                        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)  
                        (dropout): Dropout(p=0.0, inplace=False)  
                    )  
                )  
                (crossattention): BlipTextAttention(  
                    (self): BlipTextSelfAttention(  
                        (query): Linear(in_features=768, out_features=768, bias=True)  
                        (key): Linear(in_features=768, out_features=768, bias=True)  
                        (value): Linear(in_features=768, out_features=768, bias=True)  
                        (dropout): Dropout(p=0.0, inplace=False)  
                    )  
                    (output): BlipTextSelfOutput(  
                        (dense): Linear(in_features=768, out_features=768, bias=True)  
                        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)  
                        (dropout): Dropout(p=0.0, inplace=False)  
                    )  
                )  
                (intermediate): BlipTextIntermediate(  
                    (dense): Linear(in_features=768, out_features=3072, bias=True)  
                    (intermediate_act_fn): GELUActivation()  
                )  
                (output): BlipTextOutput(  
                    (dense): Linear(in_features=3072, out_features=768, bias=True)  
                    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)  
                    (dropout): Dropout(p=0.0, inplace=False)  
                )  
            )  
        )  
    )  
)
```

Lampiran 3. Layer text decoder pada model BLIP

```
● ● ●

Layer: text_decoder
BlipTextLMHeadModel(
    (bert): BlipTextModel(
        (embeddings): BlipTextEmbeddings(
            (word_embeddings): Embedding(30524, 768, padding_idx=0)
            (position_embeddings): Embedding(512, 768)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.0, inplace=False)
        )
        (encoder): BlipTextEncoder(
            (layer): ModuleList(
                (0-11): 12 x BlipTextLayer(
                    (attention): BlipTextAttention(
                        (self): BlipTextSelfAttention(
                            (query): Linear(in_features=768, out_features=768, bias=True)
                            (key): Linear(in_features=768, out_features=768, bias=True)
                            (value): Linear(in_features=768, out_features=768, bias=True)
                            (dropout): Dropout(p=0.0, inplace=False)
                        )
                        (output): BlipTextSelfOutput(
                            (dense): Linear(in_features=768, out_features=768, bias=True)
                            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
                            (dropout): Dropout(p=0.0, inplace=False)
                        )
                    )
                    (crossattention): BlipTextAttention(
                        (self): BlipTextSelfAttention(
                            (query): Linear(in_features=768, out_features=768, bias=True)
                            (key): Linear(in_features=768, out_features=768, bias=True)
                            (value): Linear(in_features=768, out_features=768, bias=True)
                            (dropout): Dropout(p=0.0, inplace=False)
                        )
                        (output): BlipTextSelfOutput(
                            (dense): Linear(in_features=768, out_features=768, bias=True)
                            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
                            (dropout): Dropout(p=0.0, inplace=False)
                        )
                    )
                )
                (intermediate): BlipTextIntermediate(
                    (dense): Linear(in_features=768, out_features=3072, bias=True)
                    (intermediate_act_fn): GELUActivation()
                )
                (output): BlipTextOutput(
                    (dense): Linear(in_features=3072, out_features=768, bias=True)
                    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
                    (dropout): Dropout(p=0.0, inplace=False)
                )
            )
        )
    )
    (cls): BlipTextOnlyMLMHead(
        (predictions): BlipTextLMPredictionHead(
            (transform): BlipTextPredictionHeadTransform(
                (dense): Linear(in_features=768, out_features=768, bias=True)
                (transform_act_fn): GELUActivation()
                (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            )
            (decoder): Linear(in_features=768, out_features=30524, bias=True)
        )
    )
)
```

Lampiran 4. Hasil pelatihan dengan VGG19-LSTM untuk pertanyaan *close ended*

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	43,02%	43,02	13,60	9,41
PathVQA	15	8	5×10^{-5}	53,44%	53,45	16,90	11,69
PathVQA	15	16	1×10^{-5}	44,18%	44,18	13,97	9,67
PathVQA	15	16	5×10^{-5}	51,44%	51,44	16,27	11,25
PathVQA	15	32	1×10^{-5}	47,99%	47,99	15,18	10,50
PathVQA	15	32	5×10^{-5}	45,40%	45,40	14,36	9,93
PathVQA	30	8	1×10^{-5}	39,45%	39,45	12,48	8,63
PathVQA	30	8	5×10^{-5}	48,85%	48,85	15,45	10,69
PathVQA	30	16	1×10^{-5}	52,25%	52,25	16,52	11,43
PathVQA	30	16	5×10^{-5}	49,42%	49,42	15,63	10,81
PathVQA	30	32	1×10^{-5}	38,98%	38,98	12,33	8,53
PathVQA	30	32	5×10^{-5}	49,60%	49,60	15,69	10,85
PathVQA	45	8	1×10^{-5}	48,25%	48,25	15,26	10,56
PathVQA	45	8	5×10^{-5}	48,66%	48,66	15,39	10,65
PathVQA	45	16	1×10^{-5}	38,31%	38,32	12,12	8,39
PathVQA	45	16	5×10^{-5}	51,23%	51,24	16,20	11,21
PathVQA	45	32	1×10^{-5}	49,63%	49,65	15,70	10,86
PathVQA	45	32	5×10^{-5}	14,80%	14,80	4,68	3,24
VQA-RAD	15	8	1×10^{-5}	43,52%	43,52	13,76	9,52
VQA-RAD	15	8	5×10^{-5}	52,85%	52,85	16,71	11,56
VQA-RAD	15	16	1×10^{-5}	47,46%	47,46	15,01	10,38
VQA-RAD	15	16	5×10^{-5}	60,71%	60,71	19,20	13,28
VQA-RAD	15	32	1×10^{-5}	57,27%	57,27	18,11	12,53
VQA-RAD	15	32	5×10^{-5}	62,79%	62,79	19,86	13,74
VQA-RAD	30	8	1×10^{-5}	58,68%	58,68	18,56	12,84
VQA-RAD	30	8	5×10^{-5}	69,05%	69,05	21,83	15,11
VQA-RAD	30	16	1×10^{-5}	60,83%	60,83	19,24	13,31
VQA-RAD	30	16	5×10^{-5}	66,67%	66,67	21,08	14,59
VQA-RAD	30	32	1×10^{-5}	61,02%	61,02	19,30	13,35
VQA-RAD	30	32	5×10^{-5}	57,14%	57,14	18,07	12,50
VQA-RAD	45	8	1×10^{-5}	62,81%	62,81	19,86	13,74
VQA-RAD	45	8	5×10^{-5}	71,05%	71,05	22,47	15,54

VQA-RAD	45	16	1×10^{-5}	57,50%	57,50	18,18	12,58
VQA-RAD	45	16	5×10^{-5}	66,93%	66,93	21,16	14,64
VQA-RAD	45	32	1×10^{-5}	57,98%	57,98	18,34	12,69
VQA-RAD	45	32	5×10^{-5}	68,75%	68,75	21,74	15,04

Lampiran 5. Hasil pelatihan dengan VGG19-LSTM untuk pertanyaan *open ended*

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	10,16%	10,89	3,86	2,57
PathVQA	15	8	5×10^{-5}	19,90%	21,41	8,01	5,38
PathVQA	15	16	1×10^{-5}	1,29%	1,37	0,48	0,32
PathVQA	15	16	5×10^{-5}	18,53%	19,95	7,96	5,23
PathVQA	15	32	1×10^{-5}	0,43%	0,48	0,15	0,10
PathVQA	15	32	5×10^{-5}	15,63%	16,67	6,25	4,01
PathVQA	30	8	1×10^{-5}	16,17%	17,94	6,67	4,49
PathVQA	30	8	5×10^{-5}	20,43%	22,46	9,02	5,87
PathVQA	30	16	1×10^{-5}	14,59%	16,22	6,49	4,29
PathVQA	30	16	5×10^{-5}	14,48%	17,32	7,30	5,08
PathVQA	30	32	1×10^{-5}	7,36%	8,01	2,83	1,88
PathVQA	30	32	5×10^{-5}	16,49%	18,69	7,56	5,15
PathVQA	45	8	1×10^{-5}	15,46%	18,06	7,43	5,09
PathVQA	45	8	5×10^{-5}	22,15%	24,28	10,02	6,81
PathVQA	45	16	1×10^{-5}	17,61%	19,65	7,86	5,14
PathVQA	45	16	5×10^{-5}	19,77%	22,32	9,63	6,72
PathVQA	45	32	1×10^{-5}	13,72%	15,74	6,51	4,31
PathVQA	45	32	5×10^{-5}	1,95%	2,65	1,25	0,88
VQA-RAD	15	8	1×10^{-5}	6,84%	7,87	5,53	4,27
VQA-RAD	15	8	5×10^{-5}	15,69%	17,60	10,74	7,37
VQA-RAD	15	16	1×10^{-5}	5,61%	6,30	4,10	2,75
VQA-RAD	15	16	5×10^{-5}	8,85%	10,27	7,01	4,19
VQA-RAD	15	32	1×10^{-5}	1,74%	1,90	1,90	1,44
VQA-RAD	15	32	5×10^{-5}	5,21%	5,79	3,26	2,34
VQA-RAD	30	8	1×10^{-5}	10,58%	12,55	8,99	7,25
VQA-RAD	30	8	5×10^{-5}	18,18%	20,17	14,34	11,32
VQA-RAD	30	16	1×10^{-5}	3,81%	5,11	3,18	2,92

VQA-RAD	30	16	5×10^{-5}	6,31%	8,17	5,68	3,91
VQA-RAD	30	32	1×10^{-5}	5,61%	6,17	6,17	4,88
VQA-RAD	30	32	5×10^{-5}	7,96%	9,37	6,01	4,65
VQA-RAD	45	8	1×10^{-5}	11,54%	12,26	7,23	5,51
VQA-RAD	45	8	5×10^{-5}	11,71%	13,08	7,45	4,74
VQA-RAD	45	16	1×10^{-5}	11,43%	13,19	10,84	6,99
VQA-RAD	45	16	5×10^{-5}	18,37%	20,69	16,69	11,29
VQA-RAD	45	32	1×10^{-5}	3,77%	4,97	3,80	2,97
VQA-RAD	45	32	5×10^{-5}	14,16%	15,16	11,64	7,85

Lampiran 6. Hasil pelatihan dengan VGG19-LSTM untuk semua pertanyaan

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	26,96%	27,32	8,84	6,07
PathVQA	15	8	5×10^{-5}	37,07%	37,81	12,56	8,61
PathVQA	15	16	1×10^{-5}	22,73%	22,77	7,23	4,99
PathVQA	15	16	5×10^{-5}	34,68%	35,41	12,04	8,19
PathVQA	15	32	1×10^{-5}	24,39%	24,41	7,72	5,34
PathVQA	15	32	5×10^{-5}	30,70%	31,21	10,35	7,01
PathVQA	30	8	1×10^{-5}	27,76%	28,65	9,56	6,55
PathVQA	30	8	5×10^{-5}	34,41%	35,44	12,18	8,24
PathVQA	30	16	1×10^{-5}	33,03%	33,86	11,40	7,79
PathVQA	30	16	5×10^{-5}	31,89%	33,32	11,45	7,94
PathVQA	30	32	1×10^{-5}	23,19%	23,51	7,59	5,21
PathVQA	30	32	5×10^{-5}	33,18%	34,27	11,65	8,02
PathVQA	45	8	1×10^{-5}	31,80%	33,11	11,33	7,82
PathVQA	45	8	5×10^{-5}	35,20%	36,28	12,67	8,70
PathVQA	45	16	1×10^{-5}	27,97%	28,99	9,99	6,77
PathVQA	45	16	5×10^{-5}	35,39%	36,67	12,89	8,95
PathVQA	45	32	1×10^{-5}	31,43%	32,46	11,04	7,54
PathVQA	45	32	5×10^{-5}	8,12%	8,48	2,90	2,01
VQA-RAD	15	8	1×10^{-5}	24,44%	24,98	9,48	6,79
VQA-RAD	15	8	5×10^{-5}	36,00%	36,87	14,00	9,66
VQA-RAD	15	16	1×10^{-5}	27,56%	27,89	9,82	6,75
VQA-RAD	15	16	5×10^{-5}	34,67%	35,38	13,08	8,71

VQA-RAD	15	32	1×10^{-5}	28,89%	28,97	9,83	6,86
VQA-RAD	15	32	5×10^{-5}	38,22%	38,47	12,77	8,87
VQA-RAD	30	8	1×10^{-5}	36,44%	37,36	14,13	10,26
VQA-RAD	30	8	5×10^{-5}	46,67%	47,54	18,54	13,44
VQA-RAD	30	16	1×10^{-5}	34,22%	34,83	11,74	8,46
VQA-RAD	30	16	5×10^{-5}	36,89%	37,81	13,48	9,32
VQA-RAD	30	32	1×10^{-5}	34,67%	34,94	13,06	9,32
VQA-RAD	30	32	5×10^{-5}	32,44%	33,15	12,01	8,56
VQA-RAD	45	8	1×10^{-5}	39,11%	39,45	14,03	9,94
VQA-RAD	45	8	5×10^{-5}	41,78%	42,45	15,06	10,21
VQA-RAD	45	16	1×10^{-5}	36,00%	36,82	14,76	9,97
VQA-RAD	45	16	5×10^{-5}	45,78%	46,79	19,22	13,18
VQA-RAD	45	32	1×10^{-5}	32,44%	33,01	11,49	8,11
VQA-RAD	45	32	5×10^{-5}	41,33%	41,83	16,67	11,43

Lampiran 7. Hasil pelatihan dengan BLIP untuk pertanyaan *close ended*

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	83,44%	83,44	26,38	18,25
PathVQA	15	8	5×10^{-5}	49,27%	49,27	15,58	10,78
PathVQA	15	16	1×10^{-5}	72,38%	72,38	22,89	15,83
PathVQA	15	16	5×10^{-5}	82,85%	82,85	26,20	18,13
PathVQA	15	32	1×10^{-5}	64,24%	64,24	20,32	14,05
PathVQA	15	32	5×10^{-5}	79,25%	79,25	25,06	17,34
PathVQA	30	8	1×10^{-5}	77,15%	77,15	24,40	16,88
PathVQA	30	8	5×10^{-5}	75,46%	75,46	23,86	16,51
PathVQA	30	16	1×10^{-5}	74,43%	74,43	23,54	16,28
PathVQA	30	16	5×10^{-5}	71,00%	71,00	22,45	15,53
PathVQA	30	32	1×10^{-5}	76,11%	76,11	24,07	16,65
PathVQA	30	32	5×10^{-5}	86,00%	86,00	27,20	18,82
PathVQA	45	8	1×10^{-5}	76,02%	76,02	24,04	16,63
PathVQA	45	8	5×10^{-5}	44,42%	44,42	14,05	9,72
PathVQA	45	16	1×10^{-5}	77,33%	77,33	24,45	16,92
PathVQA	45	16	5×10^{-5}	81,77%	81,77	25,86	17,89
PathVQA	45	32	1×10^{-5}	76,56%	76,56	24,21	16,75

PathVQA	45	32	5×10^{-5}	81,70%	81,70	25,83	17,87
VQA-RAD	15	8	1×10^{-5}	61,54%	61,54	19,46	13,46
VQA-RAD	15	8	5×10^{-5}	56,90%	56,90	17,99	12,45
VQA-RAD	15	16	1×10^{-5}	0,84%	0,84	0,27	0,18
VQA-RAD	15	16	5×10^{-5}	51,24%	51,24	16,20	11,21
VQA-RAD	15	32	1×10^{-5}	0,90%	0,90	0,28	0,20
VQA-RAD	15	32	5×10^{-5}	46,49%	46,49	14,70	10,17
VQA-RAD	30	8	1×10^{-5}	22,94%	22,94	7,25	5,02
VQA-RAD	30	8	5×10^{-5}	52,85%	52,85	16,71	11,56
VQA-RAD	30	16	1×10^{-5}	9,35%	9,35	2,96	2,04
VQA-RAD	30	16	5×10^{-5}	45,00%	45,00	14,23	9,84
VQA-RAD	30	32	1×10^{-5}	1,67%	1,67	0,53	0,36
VQA-RAD	30	32	5×10^{-5}	10,00%	10,00	3,16	2,19
VQA-RAD	45	8	1×10^{-5}	34,17%	34,17	10,80	7,47
VQA-RAD	45	8	5×10^{-5}	56,67%	56,67	17,92	12,40
VQA-RAD	45	16	1×10^{-5}	12,50%	12,50	3,95	2,73
VQA-RAD	45	16	5×10^{-5}	47,50%	47,50	15,02	10,39
VQA-RAD	45	32	1×10^{-5}	3,33%	3,33	1,05	0,73
VQA-RAD	45	32	5×10^{-5}	51,67%	51,67	16,34	11,30

Lampiran 8. Hasil pelatihan dengan BLIP untuk pertanyaan *open ended*

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	22,74%	26,11	11,69	8,00
PathVQA	15	8	5×10^{-5}	0,94%	0,96	0,34	0,26
PathVQA	15	16	1×10^{-5}	7,63%	9,98	4,17	2,88
PathVQA	15	16	5×10^{-5}	19,42%	21,78	11,37	7,67
PathVQA	15	32	1×10^{-5}	7,50%	9,67	3,84	2,57
PathVQA	15	32	5×10^{-5}	15,12%	18,05	9,48	6,69
PathVQA	30	8	1×10^{-5}	8,21%	10,59	5,19	3,80
PathVQA	30	8	5×10^{-5}	3,57%	5,93	3,35	2,51
PathVQA	30	16	1×10^{-5}	13,04%	15,53	6,15	4,36
PathVQA	30	16	5×10^{-5}	5,00%	7,67	5,82	3,79
PathVQA	30	32	1×10^{-5}	6,04%	7,39	2,85	1,99
PathVQA	30	32	5×10^{-5}	11,13%	14,19	7,80	5,80

PathVQA	45	8	1×10^{-5}	9,00%	13,39	7,15	5,20
PathVQA	45	8	5×10^{-5}	2,35%	2,51	1,08	0,69
PathVQA	45	16	1×10^{-5}	5,68%	8,46	4,04	2,94
PathVQA	45	16	5×10^{-5}	12,29%	14,93	8,13	5,82
PathVQA	45	32	1×10^{-5}	10,10%	11,60	4,82	3,49
PathVQA	45	32	5×10^{-5}	14,89%	17,68	9,60	6,68
VQA-RAD	15	8	1×10^{-5}	39,64%	43,13	30,54	22,58
VQA-RAD	15	8	5×10^{-5}	2,75%	5,86	2,33	1,49
VQA-RAD	15	16	1×10^{-5}	1,89%	5,75	2,13	1,52
VQA-RAD	15	16	5×10^{-5}	9,62%	14,28	7,92	5,60
VQA-RAD	15	32	1×10^{-5}	2,63%	3,53	1,75	1,05
VQA-RAD	15	32	5×10^{-5}	5,41%	6,37	2,92	2,38
VQA-RAD	30	8	1×10^{-5}	0,86%	5,09	2,88	1,91
VQA-RAD	30	8	5×10^{-5}	6,86%	9,36	3,60	2,30
VQA-RAD	30	16	1×10^{-5}	0,85%	3,48	1,93	1,71
VQA-RAD	30	16	5×10^{-5}	3,81%	5,65	2,90	1,76
VQA-RAD	30	32	1×10^{-5}	2,86%	3,70	1,28	0,92
VQA-RAD	30	32	5×10^{-5}	1,90%	4,76	2,40	1,60
VQA-RAD	45	8	1×10^{-5}	3,81%	7,46	3,94	2,93
VQA-RAD	45	8	5×10^{-5}	15,24%	18,11	9,48	5,88
VQA-RAD	45	16	1×10^{-5}	5,71%	8,04	3,95	2,95
VQA-RAD	45	16	5×10^{-5}	4,76%	7,51	4,69	2,95
VQA-RAD	45	32	1×10^{-5}	3,81%	5,21	3,01	2,23
VQA-RAD	45	32	5×10^{-5}	5,71%	8,49	3,93	2,54

Lampiran 9. Hasil pelatihan dengan BLIP untuk semua pertanyaan

<i>Dataset</i>	<i>Epochs</i>	<i>Batch size</i>	<i>Learning rate</i>	Akurasi	BLEU-1	BLEU-2	BLEU-3
PathVQA	15	8	1×10^{-5}	52,94%	54,63	19,00	13,10
PathVQA	15	8	5×10^{-5}	24,14%	24,15	7,66	5,31
PathVQA	15	16	1×10^{-5}	39,37%	40,57	13,34	9,23
PathVQA	15	16	5×10^{-5}	51,84%	52,99	18,95	13,01
PathVQA	15	32	1×10^{-5}	36,43%	37,49	12,24	8,43
PathVQA	15	32	5×10^{-5}	47,03%	48,50	17,23	11,99
PathVQA	30	8	1×10^{-5}	42,40%	43,60	14,71	10,29

PathVQA	30	8	5×10^{-5}	39,64%	40,82	13,64	9,54
PathVQA	30	16	1×10^{-5}	43,41%	44,67	14,75	10,26
PathVQA	30	16	5×10^{-5}	37,01%	38,38	13,89	9,48
PathVQA	30	32	1×10^{-5}	41,97%	42,63	13,73	9,51
PathVQA	30	32	5×10^{-5}	48,50%	50,03	17,48	12,29
PathVQA	45	8	1×10^{-5}	43,17%	45,32	15,76	11,03
PathVQA	45	8	5×10^{-5}	23,04%	23,12	7,46	5,13
PathVQA	45	16	1×10^{-5}	42,19%	43,55	14,44	10,06
PathVQA	45	16	5×10^{-5}	46,26%	47,61	16,80	11,72
PathVQA	45	32	1×10^{-5}	43,29%	44,04	14,51	10,11
PathVQA	45	32	5×10^{-5}	48,44%	49,82	17,75	12,30
VQA-RAD	15	8	1×10^{-5}	39,73%	43,21	30,50	22,54
VQA-RAD	15	8	5×10^{-5}	30,67%	32,17	10,41	7,14
VQA-RAD	15	16	1×10^{-5}	1,33%	3,15	1,14	0,81
VQA-RAD	15	16	5×10^{-5}	32,00%	34,16	12,37	8,62
VQA-RAD	15	32	1×10^{-5}	1,78%	2,23	1,03	0,63
VQA-RAD	15	32	5×10^{-5}	26,22%	26,70	8,89	6,33
VQA-RAD	30	8	1×10^{-5}	11,56%	13,74	5,00	3,42
VQA-RAD	30	8	5×10^{-5}	32,00%	33,13	10,77	7,36
VQA-RAD	30	16	1×10^{-5}	4,89%	6,27	2,42	1,87
VQA-RAD	30	16	5×10^{-5}	25,78%	26,64	8,94	6,07
VQA-RAD	30	32	1×10^{-5}	2,22%	2,62	0,88	0,62
VQA-RAD	30	32	5×10^{-5}	6,22%	7,56	2,81	1,91
VQA-RAD	45	8	1×10^{-5}	20,00%	21,70	7,60	5,35
VQA-RAD	45	8	5×10^{-5}	37,33%	38,67	13,98	9,36
VQA-RAD	45	16	1×10^{-5}	9,33%	10,42	3,95	2,84
VQA-RAD	45	16	5×10^{-5}	27,56%	28,84	10,20	6,92
VQA-RAD	45	32	1×10^{-5}	3,56%	4,21	1,97	1,43
VQA-RAD	45	32	5×10^{-5}	30,22%	31,52	10,55	7,21

Lampiran 10. Proses demo pada citra patologi di halaman web

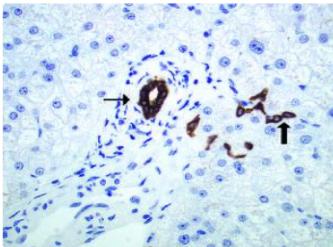
Visual Question Answering For Pathology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with pathology image.

Question:
what are strained here with an immunohistochemical stain?

Image:
 img_0.jpg



Submit

Answer: bile duct cells and canals of hering (Score: 37.51%)

Answer: bile duct cells and canals of herage (Score: 19.34%)

Answer: bile duct cells and canals of heris (Score: 17.86%)

Answer: bile duct cells and blood vessels (Score: 12.66%)

Answer: bile duct cells and canals of herin (Score: 12.62%)

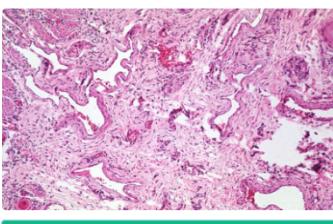
Visual Question Answering For Pathology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with pathology image.

Question:
what are strained here with an immunohistochemical stain?

Image:
 image_5.jpg



Submit

Answer: esophagus (Score: 26.41%)

Answer: varices (Score: 21.92%)

Answer: gastric ulcers (Score: 19.40%)

Answer: nodular hyperplasia (Score: 16.20%)

Answer: hodgkins disease (Score: 16.07%)

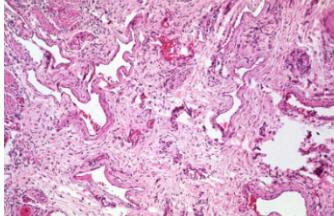
Visual Question Answering For Pathology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with pathology image.

Question:
is normal palmer creases present?

Image:
 image_5.jpg



Submit

Answer: no (Score: 94.47%)

Answer: yes (Score: 2.62%)

Answer: no no (Score: 2.15%)

Answer: not (Score: 0.40%)

Answer: brain (Score: 0.36%)

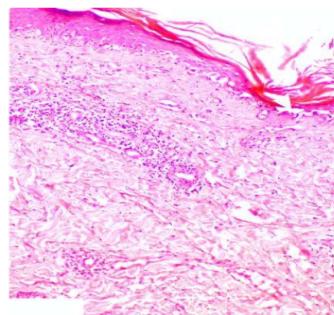
Visual Question Answering For Pathology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with pathology image.

Question:
what is there?

Image:
 image_32622.jpg



Submit

Answer: hyperkeratosis (Score: 38.05%)

Answer: hyperplasia (Score: 19.64%)

Answer: hyper caseous necrosis (Score: 16.22%)

Answer: hyper caseation necrosis (Score: 15.57%)

Answer: hyper caseation (Score: 10.52%)

Lampiran 11. Proses demo pada citra radiologi di halaman web

Visual Question Answering For Radiology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with radiology image.

question:
which organ system is abnormal in this image?

image:
 image_2.jpg



Submit

Answer: cardiovascular (Score: 74.06%)

Answer: respiratory system (Score: 11.83%)

Answer: brain (Score: 6.31%)

Answer: no (Score: 4.98%)

Answer: cardiovascular lobes (Score: 2.82%)

Visual Question Answering For Radiology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with radiology image.

question:
is there an evidence of an aortic aneurysm?

image:
 image_2.jpg



Submit

Answer: yes (Score: 78.23%)

Answer: no (Score: 11.90%)

Answer: on the left (Score: 6.20%)

Answer: yes lobes (Score: 2.88%)

Answer: mid (Score: 0.79%)

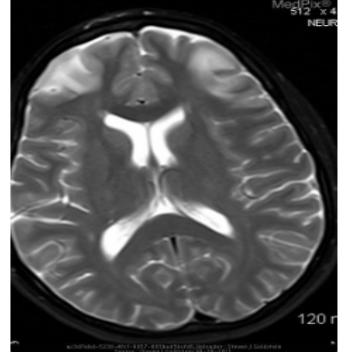
Visual Question Answering For Radiology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with radiology image.

Question:
which plane is in this image?

Image:
 image_12.jpg



MedPix®
512 × 4
NELL
120 r
120 r
Submit

Answer: axial (Score: 70.95%)

Answer: pa (Score: 18.35%)

Answer: yes (Score: 5.02%)

Answer: axial lobes (Score: 2.88%)

Answer: above (Score: 2.80%)

Answer: below (Score: 0.00%)

Visual Question Answering For Radiology Image

Medical Smart Live Demo

This is a live demo of the Medical Smart application that aims to address medical visual question answering tasks with radiology image.

Question:
is there air-fluid levels

Image:
 image_305.jpg



MedPix®
120 r
120 r
Submit

Answer: yes (Score: 84.60%)

Answer: no (Score: 8.40%)

Answer: yes lobes (Score: 3.18%)

Answer: yes ct (Score: 2.27%)

Answer: maybe (Score: 1.55%)

Answer: unknown (Score: 0.00%)

Lampiran 12. Biodata Penulis

BIODATA

- | | | |
|--------------------------|---|--------------------------------------|
| 1. Nama | : | Abdul Hafidh |
| 2. Tempat, tanggal lahir | : | Banda Aceh, 29 Maret 2002 |
| 3. Alamat | : | Jalan Pari No 49 Lampriet Banda Aceh |
| 4. Nama Ayah | : | Fadhli |
| 5. Pekerjaan Ayah | : | Swasta |
| 6. Nama Ibu | : | Dewi Ratna Sari |
| 7. Pekerjaan Ibu | : | Pegawai Negeri Sipil |
| 8. Alamat Orang Tua | : | Jalan Pari No 49 Lampriet Banda Aceh |
| 9. Riwayat Pendidikan | : | |

Jenjang	Nama Sekolah	Bidang Studi	Tempat	Tahun Ijazah
SD	SD Kartika XIV-1 Banda Aceh	-	Banda Aceh	2014
SMP	SMP Fatih Bilingual School Banda Aceh	-	Banda Aceh	2017
SMA	SMA Fatih Bilingual School Banda Aceh	IPA	Banda Aceh	2020

Banda Aceh, 8 Juli 2024

Abdul Hafidh
NPM. 2008107010056