

**IMPLEMENTASI ARSITEKTUR EFFICIENTNETV2-TRANSFORMER  
PADA APLIKASI IMAGE CAPTIONING BAHASA INDONESIA**

**SKRIPSI**

**MUHAMMAD TEGUH SINULINGGA**

**201401057**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**IMPLEMENTASI ARSITEKTUR EFFICIENTNETV2-TRANSFORMER  
PADA APLIKASI IMAGE CAPTIONING BAHASA INDONESIA**

**SKRIPSI**

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah  
Sarjana Ilmu Komputer

**MUHAMMAD TEGUH SINULINGGA  
201401057**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

## PERSETUJUAN

Judul : IMPLEMENTASI ARSITEKTUR EFFICIENTNETV2-  
TRANSFORMER PADA APLIKASI IMAGE  
CAPTIONING BAHASA INDONESIA

Kategori : SKRIPSI

Nama : MUHAMMAD TEGUH SINULINGGA

Nomor Induk Mahasiswa : 201401057

Program Studi : SARJANA (S1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA

Komisi Pembimbing :

Medan, 24 Juni 2024

Pembimbing 2

Pembimbing 1

Ivan Jaya S.Si., M.Kom.

NIP. 198407072015041001

Dr. Amalia ST., M.T.

NIP. 197812212014042001

Diketahui/disetujui oleh

Program Studi S1 Ilmu Komputer



**PERNYATAAN**

IMPLEMENTASI ARSITEKTUR EFFICIENTNETV2-TRANSFORMER PADA  
APLIKASI IMAGE CAPTIONING BAHASA INDONESIA

**SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 13 Mei 2024



Muhammad Teguh Sinulingga

201401057

## PENGHARGAAN

Semua pujiannya milik Allah Subhanahu Wa Ta ‘ala, yang telah memberikan keberkahan, rahmat, taufik, Inayah, serta hidayah-Nya kepada penulis sehingga mampu menyelesaikan salah satu syarat dalam memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara yaitu menyelesaikan skripsi. Shalawat beriringan salam kepada Nabi Muhammad Shallallahu ‘alaihi wasallam, karena dengan perantara beliau kita bisa merasakan nikmat ilmu pengetahuan yang diridhoi oleh Allah Subhanahu Wa Ta ‘ala.

Penulis ingin mengucapkan rasa penghargaan serta rasa terima kasih yang tulus kepada:

1. Orang tua penulis yang selalu memberikan motivasi, dukungan, saran, doa serta kasih sayang penuh kepada penulis selama menyelesaikan pendidikan.
2. Saudara kandung penulis yang selalu mendukung serta mendoakan penulis dalam menjalankan aktivitas kuliah hingga akhirnya menyelesaikan tugas akhir.
3. Ibu Dr. Maya Silvi Lydia B.Sc., M.Sc. sebagai Dekan Fakultas Ilmu Komputer dan Teknologi Informasi di Universitas Sumatera Utara.
4. Ibu Dr. Amalia ST., M.T. yang menjabat sebagai Ketua Program Studi S-1 Ilmu Komputer di Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara dan juga bertindak sebagai dosen pembimbing pertama, telah memberikan bimbingan, kritik, motivasi, dan saran yang berharga kepada penulis dalam penyelesaian skripsi.
5. Bapak Ivan Jaya S.Si., M.Kom. sebagai dosen pembimbing kedua, telah memberikan kontribusi penting dalam penyelesaian skripsi ini dengan menyediakan bimbingan, kritik, motivasi, dan saran kepada penulis.
6. Seluruh Bapak dan Ibu Dosen Program Studi S-1 Ilmu Komputer yang telah memberikan waktu dan tenaga untuk mengajar dan membimbing sehingga penulis dapat sampai kepada tahap penyusunan skripsi ini.
7. Semua civitas akademika di Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara yang telah membantu penulis sepanjang masa perkuliahan dan dalam proses penyusunan skripsi ini.
8. Sahabat perkuliahan penulis, Salsa dan Sukiya yang telah menjadi teman dekat penulis dan bersama-sama menjalani perkuliahan sejak semester 1 hingga sekarang.

9. Tim SSA (Sekali Sekali Ambis), Wana dan Wilbert, yang merupakan sahabat seperjuangan penulis dalam belajar pemrograman dan kompetisi serta terus memberikan dukungan moral dan bantuan selama penyelesaian skripsi ini.
10. Anggota grup Dominantes, yaitu Chalil, Iqbal, Andrew, Yunisa, Rio Fransiskus dan teman-teman lainnya yang telah sering membantu dan bekerja sama ketika menjalani perkuliahan.
11. Seluruh teman-teman Stambuk 2020, terkhusus penghuni KOM B yang telah mendukung penyusunan skripsi ini dan belajar bersama di perkuliahan.
12. Pengurus Departemen Wawasan Kontemporer IMILKOM Tahun 2022/2023 yang telah banyak membantu selama kegiatan penulis di lingkungan organisasi IMILKOM.
13. Selain itu, penulis juga ingin mengucapkan terima kasih kepada semua pihak yang telah memberikan bantuan dan dukungan, yang tidak mungkin disebutkan satu per satu..

Semoga rahmat Allah Subhanahu Wa Ta ‘ala senantiasa menyertai setiap langkah penulis serta diberi keberkahan kepada semua orang yang telah memberikan bimbingan, motivasi, dan dukungan dalam penyelesaian skripsi ini. Harapan juga agar skripsi ini membawa manfaat bagi pribadi penulis, keluarga, masyarakat, organisasi, dan negara.

Medan, 13 Mei 2024



Muhammad Teguh Sinulingga  
201401057

## **Implementasi Arsitektur EfficientNetV2-Transformer pada Aplikasi Image Captioning Bahasa Indonesia**

### **ABSTRAK**

*Image captioning* adalah bidang yang menggabungkan *computer vision*, *natural language processing* (NLP), dan *machine learning*. Dalam tugas ini, model tidak hanya perlu mengenali objek atau pemandangan dalam gambar, tetapi juga harus mampu mendeskripsikan hubungan antar objek atau pemandangan tersebut. *Image captioning* memiliki berbagai manfaat, seperti menambahkan judul pada gambar berita, membuat deskripsi untuk gambar medis, mendukung pencarian gambar berbasis teks, memberikan informasi gambar bagi pengguna tunanetra, dan memfasilitasi interaksi antara manusia dan robot. Saat ini, penelitian tentang *image captioning* dalam Bahasa Indonesia yang menggunakan kombinasi arsitektur CNN-Transformer masih terbatas. Penelitian terbaru menunjukkan bahwa salah satu keluarga CNN, EfficientNetV2, sebagai pengembangan dari EfficientNet, memiliki kinerja yang baik dalam ekstraksi fitur gambar. Selain itu, arsitektur Transformer telah banyak digunakan dalam tugas-tugas berbasis NLP seperti penerjemahan mesin. Namun, hingga kini belum ada studi yang mengembangkan sistem *image captioning* dalam Bahasa Indonesia dengan menggunakan kombinasi dua arsitektur tersebut. Penelitian ini bertujuan mengembangkan sistem *image captioning* yang dapat menghasilkan deskripsi gambar dalam Bahasa Indonesia. Hasil pengujian menunjukkan bahwa model yang dikembangkan mampu mencapai skor metrik BLEU-1, BLEU-2, BLEU-3, dan BLEU-4 terbaik masing-masing sebesar {0.6028, 0.3547, 0.2247, 0.1572}. Penelitian ini juga menemukan bahwa penggunaan EfficientNetV2 skala kecil dan skala medium menghasilkan deskripsi gambar yang berbeda dan nilai evaluasi yang bervariasi.

**Kata Kunci:** *Image Captioning, Natural Language Processing, Computer vision, EfficientNetV2, Transformer, Bahasa Indonesia*

## **Implementation of EfficientNetV2-Transformer Architecture for Indonesian Image Captioning Application**

### **ABSTRACT**

Image captioning is a task that combines computer vision, natural language processing (NLP), and machine learning. In this task, the model not only needs to recognize objects or scenes in the image, but also needs to be able to describe the relationships between them. Image captioning has various use case, such as adding titles to news images, creating descriptions for medical images, supporting text-based image search, providing image information for visually impaired users, and facilitating interaction between humans and robots. Currently, research on image captioning in Bahasa Indonesia using a combination of CNN-Transformer architectures is still limited. Recent research shows that one of the CNN families, EfficientNetV2, as a development from EfficientNet, has good performance in image feature extraction. In addition, the Transformer architecture has been widely used in NLP-based tasks such as machine translation. However, until now there has been no study that develops an image captioning system in Bahasa Indonesia using a combination of these two architectures. This research aims to develop an image captioning system that can generate image descriptions in Bahasa Indonesia. The test results show that the developed model is able to achieve the best BLEU-1, BLEU-2, BLEU-3, and BLEU-4 metric scores of {0.6028, 0.3547, 0.2247, 0.1572} respectively. This study also found that the use of EfficientNetV2 at small scale and medium scale resulted in different image descriptions and varied evaluation scores.

**Keywords:** Image Captioning, Computer vision, Natural Language Processing, EfficientNetV2, Transformer, Bahasa Indonesia

## DAFTAR ISI

PERSETUJUAN .....	ii
PERNYATAAN .....	iii
PENGHARGAAN .....	iv
ABSTRAK.....	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	xi
DAFTAR GAMBAR .....	xii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	3
1.3 Batasan Masalah .....	4
1.4 Tujuan Penelitian .....	4
1.5 Manfaat Penelitian .....	4
1.6 Metodologi Penelitian .....	5
1.7 Sistematika Penulisan .....	6
BAB II LANDASAN TEORI .....	7
2.1 <i>Image Captioning</i> .....	7
2.2 <i>Convolutional Neural Network</i> .....	8
2.2.1 Lapisan <i>Convolutional</i> .....	8
2.2.2 Lapisan <i>Pooling</i> .....	8
2.2.3 Lapisan <i>Fully-Connected</i> .....	9
2.3 EfficientNetV2 .....	9
2.4 Transformer.....	11
2.5 Tumpukan <i>Encoder-Decoder</i> .....	12
2.5.1 <i>Scaled Dot-Product Attention</i> .....	13
2.5.2 <i>Multi-Head Attention</i> .....	14
2.5.3 <i>Position-wise Feed-Forward Networks</i> .....	15
2.5.4 <i>Positional Encoding</i> .....	16
2.6 Metrik Evaluasi .....	16

2.6.1 BLEU .....	16
<b>BAB III ANALISIS DAN PERANCANGAN SISTEM .....</b>	<b>17</b>
3.1 Analisis Sistem.....	17
3.1.1 Analisis Masalah.....	17
3.1.2 Analisis Kebutuhan.....	17
3.2 Diagram Umum Sistem.....	18
3.3 Arsitektur Penelitian .....	19
3.4 Pengumpulan dan Pemrosesan Data .....	20
3.4.1 Pengumpulan Data .....	20
3.4.2 Penyederhanaan Data.....	21
3.4.3 Prapemrosesan Gambar.....	21
3.4.4 Prapemrosesan Kalimat.....	22
3.4.5 Pembagian Data .....	23
3.5 Pembangunan Model.....	23
3.5.1 Inisialisasi Model .....	24
3.5.2 Pelatihan dan Evaluasi Model.....	28
3.5.3 Pengujian Model .....	28
3.5.4 Ekspor Model.....	29
3.6 Implementasi Model .....	29
3.6.1 Pembuatan API dan Integrasi dengan Model.....	29
3.6.2 Perancangan Antarmuka <i>Website</i> .....	30
<b>BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM .....</b>	<b>33</b>
4.1 Lingkungan Implementasi.....	33
4.1.1 Perangkat Keras .....	33
4.1.2 Perangkat Lunak .....	33
4.2 Implementasi Pengumpulan dan Pemrosesan Data.....	34
4.2.1 Pengumpulan Data .....	34
4.2.2 Penyederhanaan Data.....	36
4.2.3 Prapemrosesan Gambar.....	36
4.2.4 Prapemrosesan Kalimat.....	37
4.3 Implementasi Pembangunan Model.....	39
4.3.1 Inisialisasi Model .....	39
4.3.2 Implementasi Pelatihan Model.....	40
4.3.3 Implementasi Evaluasi Model.....	42
4.3.4 Implementasi Pengujian Model.....	44
4.4 Penerapan Model .....	48
4.4.1 Implementasi Pembuatan API.....	48

4.4.2 Hasil Pembuatan Antarmuka <i>Website</i> .....	49
4.4.3 Pengujian Aplikasi <i>Website</i> .....	51
BAB V PENUTUP .....	55
5.1 KESIMPULAN.....	55
5.2 SARAN .....	56
BAB VI DAFTAR PUSTAKA.....	57

**DAFTAR TABEL**

<b>Tabel 2.1</b> Arsitektur EfficientNetV2-S .....	10
<b>Tabel 3.1</b> Proporsi Pembagian Data .....	23
<b>Tabel 3.2</b> Spesifikasi EfficientNetV2-S dan EfficientNetV2-M.....	25
<b>Tabel 4.1</b> Contoh Data Gambar dan Kalimat.....	34
<b>Tabel 4.2</b> Rincian Model yang Dibangun .....	40
<b>Tabel 4.3</b> Konfigurasi <i>Hyperparameter</i> .....	41
<b>Tabel 4.4</b> Perbandingan <i>Train Loss</i> , <i>Validation Loss</i> , dan Durasi Pelatihan .....	43
<b>Tabel 4.5</b> Hasil Prediksi EffNetV2S-Trans dan EffNetV2M-Trans .....	44
<b>Tabel 4.6</b> Perbandingan Kalimat Prediksi dan Kalimat Referensi .....	45
<b>Tabel 4.7</b> Hasil Evaluasi Skor BLEU Kedua Model.....	47
<b>Tabel 4.8</b> Perbandingan Evaluasi Skor BLEU dengan Penelitian Terdahulu .....	48
<b>Tabel 4.9</b> Pengujian pada <i>Website</i> untuk EffNetV2S-Trans .....	52
<b>Tabel 4.10</b> Pengujian pada <i>Website</i> untuk EffNetV2M-Trans.....	53

## DAFTAR GAMBAR

<b>Gambar 2.1</b> Contoh Penggunaan Arsitektur <i>Convolutional Neural Networks</i> .....	8
<b>Gambar 2.2</b> Struktur MBConv dan Fused-MBConv .....	10
<b>Gambar 2.3</b> Perbandingan Model CNN .....	11
<b>Gambar 2.4</b> Arsitektur Model Transformer .....	12
<b>Gambar 2.5</b> Scaled Dot-Product Attention pada Arsitektur Transformer.....	14
<b>Gambar 2.6</b> Multi-Head Attention pada Arsitektur Transformer.....	14
<b>Gambar 3.1</b> Diagram Umum Sistem .....	18
<b>Gambar 3.2</b> Arsitektur Penelitian .....	19
<b>Gambar 3.3</b> Ilustrasi Arsitektur <i>encoder</i> dan <i>decoder</i> .....	24
<b>Gambar 3.4</b> Ilustrasi Arsitektur EfficientNetV2-S dan EfficientNetV2-M .....	25
<b>Gambar 3.5</b> Ilustrasi Arsitektur <i>Decoder</i> dari Transformer .....	27
<b>Gambar 3.6</b> <i>Sequence Diagram</i> .....	29
<b>Gambar 3.7</b> Rancangan Antarmuka Halaman Beranda.....	30
<b>Gambar 3.8</b> Rancangan Antarmuka <i>Image Caption Generator</i> .....	31
<b>Gambar 3.9</b> Rancangan Antarmuka Image Caption Generator (Lanjutan) .....	31
<b>Gambar 3.10</b> Rancangan Antarmuka Image Caption Generator (Lanjutan) .....	32
<b>Gambar 3.11</b> Rancangan Antarmuka Halaman <i>About</i> .....	32
<b>Gambar 4.1</b> Gambaran Konten pada Data Teks .....	35
<b>Gambar 4.2</b> Ilustrasi Perbandingan Data Setelah Penyederhanaan Data .....	36
<b>Gambar 4.3</b> Ilustrasi Hasil Prapemrosesan Terhadap Data Gambar .....	37
<b>Gambar 4.4</b> Ilustrasi Hasil Prapemrosesan Kalimat.....	37
<b>Gambar 4.5</b> Kosakata dengan Frekuensi Kemunculan Paling Tinggi.....	38
<b>Gambar 4.6</b> Perbandingan <i>Training</i> dan <i>Validation Loss</i> .....	42
<b>Gambar 4.7</b> Implementasi Halaman Beranda.....	49
<b>Gambar 4.8</b> Implementasi Halaman Prediksi.....	50
<b>Gambar 4.9</b> Implementasi Halaman Prediksi Setelah Gambar Dipilih.....	50
<b>Gambar 4.10</b> Halaman Prediksi Setelah Melakukan Pembuatan <i>Caption</i> .....	51
<b>Gambar 4.11</b> Implementasi Halaman <i>About</i> .....	51
<b>Gambar 4.12</b> Kumpulan Gambar sebagai Objek Pengujian pada <i>Website</i> .....	52

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Di era digital saat ini, konten visual seperti gambar dan video menjadi bagian dari cara berkomunikasi dan berbagi informasi. Namun, tidak semua orang dapat mengakses konten visual ini dengan cara yang sama, seperti individu dengan keterbatasan penglihatan (Al-Faruq & Fudholi, 2021). Teks *caption* kemudian dikembangkan untuk lebih memberikan pemahaman suatu informasi yang ingin disampaikan pada citra yang ditunjukkan. Proses penerjemahan gambar menjadi teks *caption* secara otomatis untuk menggambarkan objek yang ada di dalamnya disebut dengan *image captioning* (Nursikuwagus et al., 2020).

*Image captioning* merupakan masalah bidang komputer yang melibatkan *computer vision*, *natural language processing* (NLP), dan pembelajaran mesin (Nursikuwagus et al., 2020). Dalam *image captioning*, model pembangkit deskripsi tidak hanya harus bisa menangkap objek atau pemandangan yang ada dalam sebuah gambar, tetapi juga harus mampu menggambarkan bagaimana objek atau pemandangan tersebut berhubungan satu sama lain (Fudholi et al., 2022). Manfaat *image captioning* di antaranya termasuk penambahan judul pada gambar-gambar berita, penulisan deskripsi untuk gambar-gambar medis, pencarian gambar yang didasarkan pada teks, menyediakan informasi gambar bagi pengguna tunanetra, serta memfasilitasi interaksi antara manusia dan robot (Liu et al., 2018).

Pada pengembangan sistem *image captioning* yang berbasis neural, sebagian besar dikembangkan dengan *framework encoder-decoder*, di mana sebuah gambar masukan dikodekan menjadi representasi dari informasi yang terkandung dalam gambar tersebut, kemudian didekodekan menjadi urutan teks deskriptif. Cara ini awalnya digunakan untuk pemodelan bahasa yang pertama kali diterapkan oleh Kiros dan timnya (Kiros et al., 2014). Selanjutnya metode ini dikembangkan peneliti lain menggunakan kombinasi dari *Convolutional Neural Network* (CNN) untuk pengolahan gambar dan *Long-Term Short Memory* (LSTM) untuk

pengolahan teks. Banyak sistem pembuatan *caption* modern menggunakan metode ini karena berhasil memberikan hasil yang baik (Donahue et al., 2017).

Dengan kemajuan dalam *machine translation*, muncul arsitektur inovatif bernama Transformer. Arsitektur ini memanfaatkan mekanisme *self-attention* dan kini merupakan teknologi terdepan dalam bidang NLP. Transformer mempercepat proses *training* dengan menggunakan *self-attention* untuk mengidentifikasi dependensi global antar *input* dan *output*. Dalam konteks *machine translation*, Transformer mampu menjalankan *training* dengan lebih efisien dan cepat dibandingkan dengan arsitektur yang mengandalkan lapisan *recurrent* atau *convolutional* (Vaswani et al., 2017).

Terdapat penelitian terdahulu terkait *image captioning* Bahasa Indonesia seperti penelitian (Nugraha et al., 2019) mengeksplor penggunaan arsitektur InceptionV3 dan Gated Recurrent Unit (GRU) dengan hasil evaluasi metrik BLEU-1, BLEU-2, BLEU-3, dan BLEU-4 didapatkan nilai 0.363, 0.170, 0.060, dan 0.014 secara berurut. Terdapat juga penelitian dengan basis arsitektur CNN-LSTM (Mulyawan, E. et al., 2019) yang mendapatkan skor BLEU-1 hingga BLEU-4 sebesar 0.500, 0.314, 0.239, dan 0.131 masing-masingnya. Penelitian berikutnya oleh Mulyawan et al. (2023) telah berhasil meningkatkan akurasi dengan menggunakan arsitektur CNN-Transformer, khususnya model CNN EfficientNet V1. Hasilnya, skor BLEU-1 hingga BLEU-4 yang dicapai adalah 0.577, 0.42, 0.302, dan 0.212 berturut-turut.

Mengikuti kemajuan di bidang ekstraksi fitur gambar, Tan & Le (2021) memperkenalkan model EfficientNetV2 sebagai terobosan baru dalam arsitektur *Convolutional Neural Network* yang memiliki kecepatan *training* lebih cepat dan lebih baik dalam efisiensi parameter dibandingkan model-model sebelumnya. Model ini dibangun dengan metode *progressive learning* yang mampu memangkas durasi training dan pada saat yang sama meningkatkan akurasi untuk berbagai jaringan neural. Model ini dilatih dengan dataset ImageNet, dan kemampuannya diuji dengan *transfer learning* pada dataset CIFAR, *Cars*, dan *Flowers* untuk mengevaluasi ke domain yang berbeda. Dataset ImageNet dan CIFAR mencakup gambar objek sehari-hari dalam berbagai kategori. Sementara itu, dataset *Cars*

berisi berbagai jenis mobil, dan dataset *Flowers* terdiri dari gambar-gambar bunga yang dikategorikan ke dalam ratusan jenis. Hasil evaluasi menunjukkan EfficientNetV2 mampu melampaui model sebelumnya, EfficientNets, dengan perbedaan yang signifikan: EfficientNetV2-M berhasil menurunkan jumlah parameter hingga 17% dan mengurangi FLOPs sampai 37%, serta meningkatkan kecepatan pelatihan hingga 4.1 kali lebih cepat dan kecepatan inferensi 3.1 kali lebih cepat daripada EfficientNet-B7.

Penelitian terbaru di bidang *image captioning* Bahasa Indonesia telah mengeksplorasi penggunaan arsitektur Transformer dan diintegrasikan dengan model ekstraksi fitur gambar seperti EfficientNet V1. Namun, belum ada penelitian pengembangan *image captioning* Bahasa Indonesia dengan metode CNN-Transformer yang memanfaatkan EfficientNetV2 di mana arsitektur ini membuka perspektif baru dengan *progressive learning* yang dapat meningkatkan akurasi sambil mempertahankan efisiensi parameter dan kecepatan *training* dibandingkan versi pendahulunya (Tan & Le, 2021). Oleh karena itu, pada penelitian ini, penulis menggunakan model EfficientNetV2 sebagai *encoder*, dan Transformer pada sisi *decoder*. Penelitian ini memanfaatkan dataset Flickr8k, yang terdiri dari 8.000 gambar. Setiap gambar disertai dengan lima *caption* berbeda yang menunjukkan bahwa terdapat lima variasi berbeda yang dapat diterima untuk mendeskripsikan sebuah gambar.

## 1.2 Rumusan Masalah

Di era digital saat ini, konten visual seperti gambar dan video menjadi bagian penting dalam komunikasi dan berbagi informasi. Namun, tidak semua orang dapat mengakses konten visual ini dengan cara yang sama, seperti individu dengan keterbatasan penglihatan. *Image captioning* dapat meningkatkan aksesibilitas konten visual, tidak terkecuali bagi orang-orang dengan keterbatasan penglihatan. Namun, terdapat beberapa tantangan dalam membangun *image captioning* dengan Bahasa Indonesia. Di antaranya keterbatasan dataset yang memadai, kompleksitas bahasa, dan kesulitan dalam mencapai tingkat akurasi yang tinggi. Untuk mengatasi tantangan tersebut, aplikasi *image captioning* telah banyak dikembangkan dengan arsitektur CNN-Transformer. Namun, belum terdapat penelitian yang memanfaatkan kombinasi arsitektur CNN EfficientNetV2 dengan Transformer

untuk aplikasi *image captioning* dalam Bahasa Indonesia. Oleh karena itu, pada penelitian ini, penulis akan mengeksplorasi penggunaan arsitektur CNN, khususnya EfficientNetV2, bersama Transformer dalam *image captioning* Bahasa Indonesia untuk mengukur tingkat akurasi dan efisiensi performa model.

### 1.3 Batasan Masalah

Ruang lingkup penelitian ini dibatasi oleh beberapa hal sebagai berikut:

1. Sistem yang dibangun menghasilkan keluaran teks Bahasa Indonesia yang mendeskripsikan objek di dalam gambar masukan.
2. Menggunakan arsitektur EfficientNetV2-S dan EfficientNetV2-M sebagai model ekstraksi fitur gambar serta Transformer untuk menghasilkan teks deskripsi.
3. Menggunakan dataset Flickr8k yang setiap gambar disertai dengan lima *caption* dan diterjemahkan ke dalam Bahasa Indonesia. Dataset terdiri dari kumpulan gambar dengan subjek umum tanpa adanya pengkhususan.
4. Metrik evaluasi yang dipakai saat pengujian adalah BLEU (*Bilingual Evaluation Understudy*). Pengujian sistem dengan metrik evaluasi hanya dilakukan terhadap gambar yang berasal dari data uji pada dataset Flickr8k.
5. Model dikembangkan dengan *framework* Pytorch dalam bahasa pemrograman Python.
6. Model diaplikasikan melalui API yang terintegrasi dengan antarmuka *website*. Pengguna *website* dapat mengunggah satu buah gambar untuk mendapatkan satu deskripsi gambar.

### 1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk mengukur seberapa baik tingkat akurasi dan efisiensi performa kecepatan model *image captioning* Bahasa Indonesia dengan mengimplementasikan kombinasi dari arsitektur EfficientNetV2 dan Transformer.

### 1.5 Manfaat Penelitian

Penelitian ini diharapkan mampu menghasilkan sebuah aplikasi *image captioning* Bahasa Indonesia yang mampu secara otomatis menghasilkan deskripsi

gambar dalam Bahasa Indonesia untuk digunakan oleh masyarakat umum. Selain itu, hasil penelitian ini dapat dijadikan referensi bagi penelitian di masa depan yang mengeksplorasi bidang *image captioning*, khususnya yang berfokus pada penggunaan Bahasa Indonesia.

## 1.6 Metodologi Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah sebagai berikut:

### 1. Studi Pustaka

Peneliti akan memulai penelitian dengan mengumpulkan referensi yang diperoleh dari sumber tertulis, seperti artikel jurnal dan *proceeding*. Pencarian referensi dilakukan untuk memperoleh data dan informasi yang berkaitan dengan teknologi *image captioning*, konsep *Convolutional Neural Network* sebagai pengekstraksi fitur gambar, arsitektur EfficientNetV2, dan arsitektur Transformer sebagai model yang menghasilkan deskripsi gambar.

### 2. Analisis dan Perancangan

Tahap analisis dan perancangan dimulai dengan melakukan analisis arsitektur EfficientNetV2 sebagai pengekstraksi fitur visual dari gambar serta Transformer sebagai pemodelan bahasa yang menerjemahkan masukan gambar menjadi teks deskriptif yang relevan. Selain itu, penulis juga melakukan analisis terhadap hal-hal yang dibutuhkan dalam penelitian untuk segera dilakukan perancangan sistem dalam sebuah arsitektur penelitian.

### 3. Implementasi Sistem

Pada tahap ini, dilakukan pembangunan sistem, mengikuti arsitektur penelitian yang telah dirancang sebelumnya. Setelah sistem berhasil dibangun, langkah selanjutnya adalah mengintegrasikan model tersebut ke aplikasi API dan pembuatan antarmuka *website*.

### 4. Pengujian Sistem

Pada tahapan ini, dilaksanakan pengujian terhadap sistem yang telah dibangun dengan tujuan menilai performanya. Fokus pengujian adalah pada tingkat keakuratan model dalam menghasilkan teks, serta pada kecepatan *training* dan *inference*. Pengujian ini bertujuan untuk memastikan bahwa sistem mampu beroperasi secara efisien dan menghasilkan *output* yang akurat. Metode dan

metrik spesifik digunakan untuk mengukur aspek-aspek tersebut, sehingga dapat diidentifikasi area yang memerlukan optimasi atau peningkatan.

#### 5. Dokumentasi Sistem

Pada tahap ini, dilakukan dokumentasi dari seluruh proses, mulai dari analisis hingga pengujian sistem untuk menampilkan hasil-hasil penelitian yang telah dicapai.

### 1.7 Sistematika Penulisan

Sistematika penulisan dari skripsi ini terdiri dari lima bab, yakni:

#### **BAB I PENDAHULUAN**

Bab ini memberikan penjelasan lengkap mengenai latar belakang masalah yang akan diteliti, termasuk rumusan masalah, batasan masalah, tujuan penelitian, manfaat, metode penelitian, dan sistematika penulisan.

#### **BAB II LANDASAN TEORI**

Bab ini berisikan tinjauan teoritis yang berkaitan dengan *image captioning*, *convolutional neural network*, EfficientNetV2, Transformer, dan matrik evaluasi yang digunakan.

#### **BAB III ANALISIS DAN PERANCANGAN**

Bab ini menjelaskan mengenai analisis pada arsitektur penyusun model yang digunakan disertai perancangan diagram yang diperlukan.

#### **BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM**

Bab ini mencakup penjelasan implementasi dan pengujian sistem yang didasarkan dari tahapan pada bab analisis dan perancangan.

#### **BAB V KESIMPULAN DAN SARAN**

Berisikan kesimpulan yang didapat setelah melakukan penelitian dan saran yang diberikan peneliti sebagai masukan bagi penelitian berikutnya.

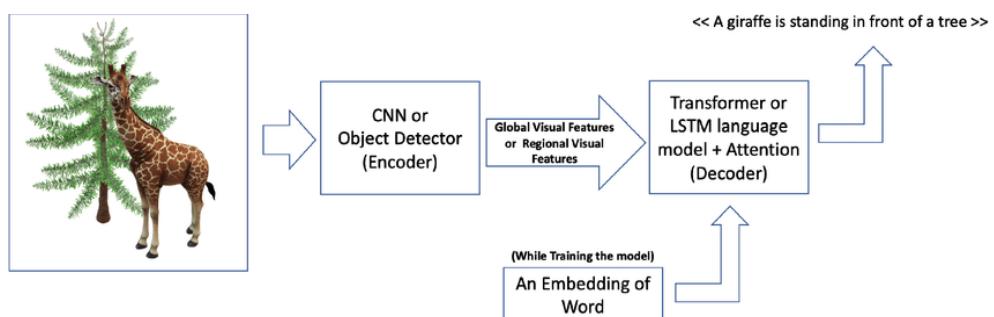
## BAB II

### LANDASAN TEORI

#### 2.1 *Image Captioning*

*Image Captioning* merupakan kemampuan untuk menggambarkan konten dari sebuah gambar melalui kalimat (He et al., 2020). *Image captioning* menggabungkan teknologi dari dua bidang *artificial intelligence*, yaitu *computer vision* untuk memahami konten gambar yang diberikan dan *natural language processing* (NLP) untuk mentransformasi konten gambar tersebut menjadi kalimat yang alami (Anderson et al., 2018). Dalam *image captioning*, model pembangkit deskripsi tidak hanya harus bisa menangkap objek atau pemandangan yang ada dalam sebuah gambar, tetapi juga harus mampu menggambarkan bagaimana objek atau pemandangan tersebut berhubungan satu sama lain (Fudholi, et al., 2022).

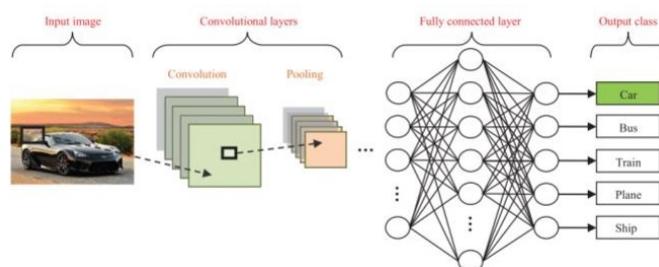
Penelitian terkini menunjukkan bahwa salah satu arsitektur *image captioning* yang efektif adalah memanfaatkan *attention mechanism* dalam kerangka kerja *encoder-decoder*. Pada arsitektur ini, *encoder* bertugas mengekstraksi fitur visual dari gambar, sedangkan *decoder* menghasilkan teks deskriptif berdasarkan fitur visual tersebut. Kerangka kerja ini biasanya menggabungkan antara arsitektur *Convolutional Neural Network* (CNN) sebagai *encoder* dan *Long Short-Term Memory* (LSTM) atau Transformer sebagai *decoder* (Zohourianshahzadi & Kalita, 2022). Gambar 2.1 merupakan ilustrasi arsitektur *image captioning* yang menggunakan kerangka kerja *encoder-decoder*.



**Gambar 2.1** Ilustrasi Arsitektur Image Captioning yang Menggunakan Kerangka Kerja *Encoder-Decoder* (Zohourianshahzadi & Kalita, 2022)

## 2.2 Convolutional Neural Network

*Convolutional Neural Network* (CNN) merupakan jenis model *deep learning* yang sangat efektif dan sering digunakan dalam beragam aplikasi, termasuk pengenalan objek, pengenalan suara, pengolahan citra komputer, klasifikasi gambar, dan analisis bioinformatika. Berbeda dengan metode tradisional, CNN (*Convolutional Neural Networks*) secara otomatis belajar dan mengenali fitur dari data tanpa memerlukan ekstraksi fitur manual oleh manusia (Shiri et al., 2023) CNN biasanya terdiri 3 lapisan jaringan, yaitu *convolutional*, *pooling*, dan *fully-connected* (O’Shea et al., 2015). Gambar 2.2 merupakan contoh penggunaan arsitektur CNN dalam klasifikasi objek.



**Gambar 2.2** Contoh Penggunaan Arsitektur *Convolutional Neural Networks*  
(Shiri et al., 2023)

### 2.2.1 Lapisan *Convolutional*

Lapisan *convolutional* atau konvolusional berperan penting dalam cara CNN bekerja. Lapisan ini terdiri dari satu set kernel konvolusional di mana setiap neuron bertugas sebagai *kernel*. *Kernel* tersebut membagi gambar menjadi irisan kecil, yang umumnya dikenal sebagai bidang reseptif. Pembagian gambar menjadi blok-blok kecil dapat membantu proses mengekstrak motif fitur. Kernel melakukan konvolusi terhadap gambar menggunakan satu set bobot tertentu dengan melakukan perkalian elemen dengan elemen-elemen lainnya yang sesuai dari bidang reseptif (Khan et al., 2020).

### 2.2.2 Lapisan *Pooling*

Lapisan *pooling* mengurangi jumlah koneksi dalam jaringan dengan melakukan *down-sampling* dan mengurangi dimensi pada data masukan. Hal ini bertujuan untuk secara bertahap mengurangi dimensionalitas representasi, sehingga mengurangi jumlah parameter dan kompleksitas komputasi dari model (O’Shea et

al., 2015). Selain itu, lapisan *pooling* memungkinkan CNN untuk mengenali objek meskipun bentuknya terdistorsi atau dilihat dari sudut yang berbeda, dengan menggabungkan berbagai dimensi dari sebuah gambar (Shiri et al., 2023).

### 2.2.3 Lapisan *Fully-Connected*

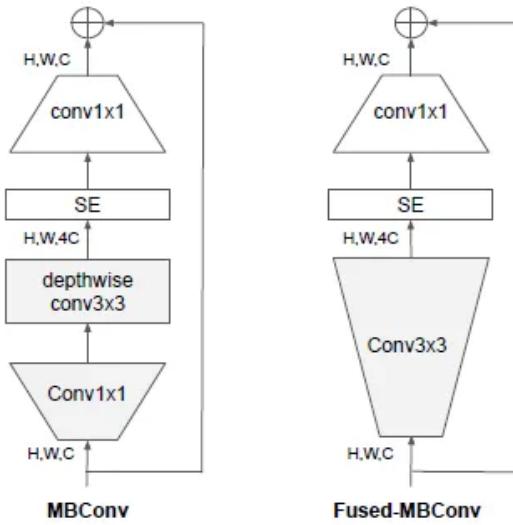
Lapisan *Fully-Connected* (FC) biasanya terletak di akhir arsitektur CNN. Di lapisan ini, setiap neuron terhubung ke semua neuron di lapisan sebelumnya. Lapisan FC menerima masukan dari lapisan *pooling* atau *convolutional* terakhir, yang merupakan vektor yang dibuat dengan meratakan peta fitur. Lapisan ini berfungsi sebagai pengklasifikasi dalam CNN, memungkinkan jaringan untuk menghasilkan prediksi berdasarkan masukan yang diberikan (Shiri et al., 2023).

## 2.3 EfficientNetV2

EfficientNetV2 merupakan seri baru dari *Convolutional Neural Network* yang dirancang untuk meningkatkan kecepatan *training* dan efisiensi parameter dibandingkan model sebelumnya, EfficientNet. Arsitektur ini dikemukakan pada makalah berjudul “EfficientNetV2: Smaller Models and Faster Training” (Tan & Le, 2021) dengan kontribusi utama sebagai berikut:

1. Merancang arsitektur model bernama EfficientNetV2 yang lebih kecil dan lebih cepat dibandingkan model pendahulunya.
2. Menyediakan metode progressive learning yang ditingkatkan, di mana metode ini dapat menyesuaikan regularisasi secara adaptif mengikuti perubahan ukuran gambar. Metode ini terbukti dapat mempercepat proses training dengan secara bersamaan meningkatkan akurasi.

Jaringan EfficientNetV2 terdiri dari blok MBConv (*Mobile Inverted Bottleneck Convolution*) dan variasinya, Fused-MBConv. Jika MBConv menggunakan *depthwise conv3x3* dan *expansion conv1x1*, Fused-MBConv menggabungkan keduanya menjadi satu konvolusi biasa. Pendekatan ini membuat Fused-MBConv menjadi lebih efisien, terutama pada lapisan-lapisan awal yang memiliki *feature maps* berukuran besar (Tan & Le, 2021). Gambar 2.3 menunjukkan perbedaan struktur MBConv dan Fused-MBConv.



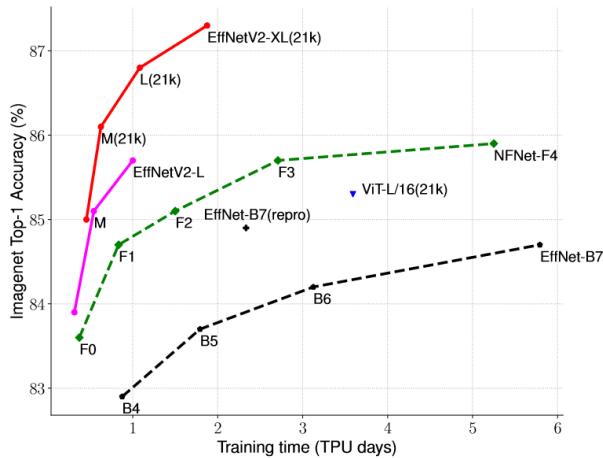
**Gambar 2.3** Struktur MBConv dan Fused-MBConv (Tan & Le, 2021)

Dalam rangka menentukan penempatan yang ideal untuk blok MBConv dan Fused-MBConv, tim peneliti EfficientNetV2 menerapkan *Neural Architecture Search* (NAS) dan penskalaan. Ditemukan bahwa Fused-MBConv memiliki beban komputasi yang lebih ringan daripada MBConv, namun cenderung mengorbankan sedikit akurasi. Oleh karena itu, blok Fused-MBConv merupakan pilihan yang lebih efisien untuk ditempatkan pada lapisan awal jaringan. Sementara itu, blok MBConv memiliki kemampuan yang lebih unggul dalam mengekstrak fitur-fitur kompleks, sehingga dianggap ideal untuk ditempatkan pada tahap lanjutan dari arsitektur (Tan & Le, 2021). Tabel 2.1 menjelaskan susunan arsitektur yang digunakan EfficientNetV2-S (skala kecil).

**Tabel 2.1** Arsitektur EfficientNetV2-S

Stage	Operator	Stride	Number of Layers	Channels
0	Conv3x3	2	1	24
1	Fused-MBConv1, k3x3	1	2	24
2	Fused-MBConv4, k3x3	2	4	48
3	Fused-MBConv4, k3x3	2	4	64
4	MBConv4, k3x3, SE0.25	2	6	128
5	MBConv6, k3x3, SE0.25	1	9	160
6	MBConv6, k3x3, SE0.25	2	15	256
7	Conv1x1 & Pooling & FC	-	1	1280

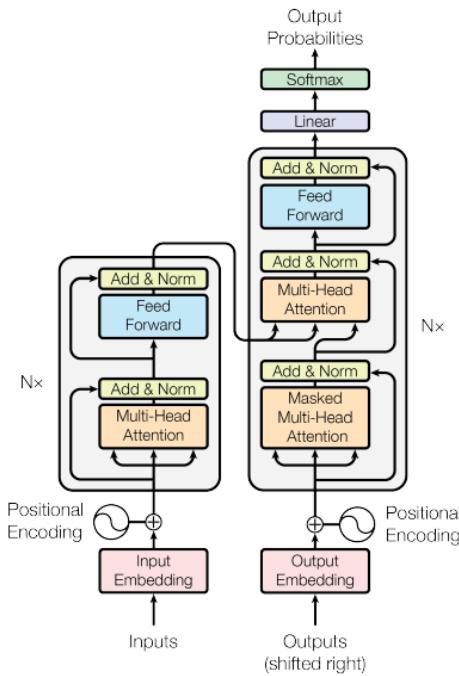
Gambar 2.4 menunjukkan perbandingan performa model CNN, meliputi EfficientNet, Vision Transformer (ViT), Normalizer-Free Networks (NFNet), dan EfficientNetV2 yang dilatih pada dataset ILSVRC2012. Pada gambar terlihat bahwa seri EfficientNetV2 mendapatkan akurasi tertinggi dan durasi waktu *training* yang paling singkat di antara ketiga model tersebut.



**Gambar 2.4** Perbandingan Model CNN (Tan & Le, 2021)

## 2.4 Transformer

Transformer dikemukakan pertama kali pada makalah “Attention is All You Need” (Vaswani et al., 2017). Transformer merupakan sebuah arsitektur jaringan sederhana yang sepenuhnya dibangun di atas mekanisme *self-attention* dan memastikan tidak adanya penggunaan jaringan *recurrent* dan *convolution*. Transformer telah menunjukkan kinerja yang sangat baik dalam berbagai tugas pada bidang NLP, seperti klasifikasi teks, penerjemahan otomatis, dan menjawab kuesioner (Khan et al., 2022). Gambar 2.4 menunjukkan gambaran arsitektur Transformer.



**Gambar 2.5** Arsitektur Model Transformer (Vasnawi et al., 2017)

Secara umum, arsitektur Transformer dibangun menggunakan tumpukan *self-attention* dan lapisan *fully connected* yang berbasis *point-wise* untuk *encoder* maupun *decoder*. Masing-masing ditunjukkan pada bagian kiri dan kanan dari Gambar 2.5.

## 2.5 Tumpukan *Encoder-Decoder*

Bagian *encoder* dan *decoder* dalam arsitektur Transformer terdiri dari  $N$  lapisan yang identik, masing-masing mengandung dua sub-lapisan. Sub-lapisan pertama adalah mekanisme *multi-head attention*, diikuti oleh lapisan kedua, yaitu jaringan *position-wise fully connected feed-forward network*. Di antara kedua lapisan tersebut ditambahkan koneksi *residual* diikuti dengan lapisan normalisasi.

Lapisan identik pada bagian *decoder* terdiri dari tiga sub-lapisan yang juga merupakan mekanisme *multi-head attention*. Serupa dengan *encoder*, di antara setiap lapisan juga terdapat koneksi *residual* yang diikuti lapisan normalisasi. Perbedaan antara *decoder* dan *encoder* terlihat pada lapisan *self-attention*. Dalam *decoder*, sub-lapisan yang pertama dimodifikasi dengan teknik *masking*. Teknik ini berfungsi untuk mencegah setiap posisi mengakses informasi dari posisi berikutnya. Dengan demikian, setiap representasi kata dalam kalimat yang berasal dari *output embedding* akan melalui proses *masking*. Hal ini penting dilakukan

untuk memastikan bahwa prediksi yang dihasilkan pada posisi ke-*i* hanya akan bergantung pada keluaran yang sudah diketahui sebelumnya.

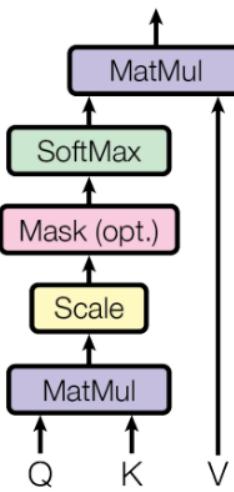
*Output* setiap komponen baik pada bagian *encoder* dan *decoder* adalah  $\text{LayerNorm}(x + \text{Sublayer}(x))$ , di mana  $\text{Sublayer}(x)$  merupakan fungsi yang diaplikasikan pada komponen itu sendiri. Dimensi keluaran dari semua komponen termasuk lapisan *embedding* adalah  $d_{\text{model}} = 512$  (Vasnawi et al., 2017).

### 2.5.1 Scaled Dot-Product Attention

*Scaled Dot-Product Attention* menerima masukan yang terdiri dari *query* dan kunci-kunci dari dimensi  $d_k$ , dan nilai dimensi  $d_v$ . Fungsi ini menghitung *dot product* dari *query* dengan seluruh *keys*, membaginya dengan  $\sqrt{d_k}$ , lalu mengaplikasikan fungsi *softmax* untuk mendapatkan nilai beratnya. Dalam praktiknya, komputasi *attention* dilakukan secara bersamaan pada seluruh rangkaian *query* yang dikumpulkan menjadi sebuah matriks Q. *Keys* dan *values* juga dikumpulkan dalam bentuk matriks K dan V secara berurut. Persamaan 1 merepresentasikan fungsi *attention*:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

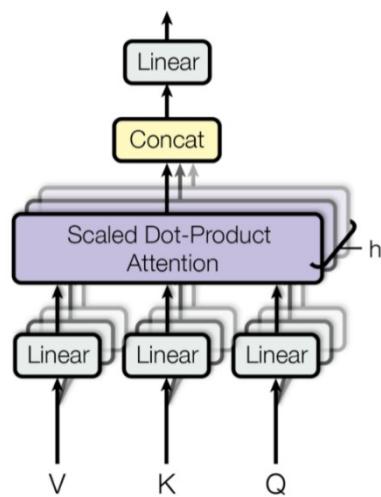
Pembagian hasil *dot product* dengan  $\sqrt{d_k}$  berfungsi sebagai penskalaan. Penskalaan ini dilakukan untuk membantu dalam stabilisasi *gradien* selama proses pelatihan, karena proses *dot product* dengan nilai  $d_k$  yang besar dapat mendorong fungsi *softmax* ke daerah dengan gradien yang sangat kecil (Vasnawi et al., 2017). Gambar 2.6 menunjukkan ilustrasi *scaled dot-product attention*.



**Gambar 2.6** Scaled Dot-Product Attention pada Transformer (Vasnawi et al., 2017)

### 2.5.2 Multi-Head Attention

Mekanisme *multi-head attention* memproyeksikan kumpulan *query*, *keys*, dan *values* sebanyak  $h$  kali dengan proyeksi linier yang berbeda. *Multi-head attention* ini memungkinkan model untuk secara bersamaan memberi atensi terhadap informasi dari berbagai sub-ruang representasi pada posisi yang berbeda. Mekanisme ini penting untuk menggunakan lebih dari satu *head attention*, karena jika hanya menggunakan satu *head attention*, proses rata-rata (*averaging*) dapat menghambat kemampuan ini (Vasnawi et al., 2017). Gambar 2.7 menunjukkan gambaran *multi-head attention*.



**Gambar 2.7** Multi-Head Attention pada Transformer (Vasnawi et al., 2017)

Fungsi *multi-head attention* dapat dituliskan dengan persamaan matematis berikut:

$$\begin{aligned} & \mathbf{MultiHead}(Q, K, V) \\ &= \mathbf{Concat}(\mathbf{head}_1, \mathbf{head}_2, \dots, \mathbf{head}_h)W^O \end{aligned} \quad (2)$$

di mana setiap  $\mathbf{head}_i$  dihitung sebagai berikut:

$$\mathbf{head}_i = \mathbf{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

Berikut adalah penjelasan dari persamaan diatas:

1. Fungsi *Attention* yang merepresentasikan nilai  $\mathbf{head}_i$  adalah implementasi dari *scaled dot-product attention* yang telah dijelaskan sebelumnya.
2. Q, K, V adalah matriks untuk *queries*, *keys*, dan *values*. Q dihitung dari *input* dengan dikalikan dengan matriks bobot  $W^Q$ , K dengan  $W^K$ , dan V dengan  $W^V$ , di mana  $W^Q$ ,  $W^K$ , dan  $W^V$  adalah parameter yang dipelajari.
3. Fungsi  $\mathbf{Concat}(\mathbf{head}_1, \mathbf{head}_2, \dots, \mathbf{head}_h)$  adalah proses penggabungan *output* dari *single-head attention*
4.  $W^O$  digunakan untuk melakukan perkalian terhadap hasil *concatenation* dari semua *head attention*, sehingga menghasilkan keluaran akhir dari *multi-head attention*.

### 2.5.3 Position-wise Feed-Forward Networks

Tambahan untuk sub-lapisan *attention*, setiap lapisan baik di dalam *encoder* maupun *decoder* terdapat sebuah *fully connected feed-forward network* (FFN) yang terdiri dari dua transformasi linear dengan fungsi aktivasi ReLU di antaranya. Persamaan 4 merepresentasikan jaringan tersebut, di mana  $W_1$  dan  $W_2$  adalah matriks bobot dan  $b_1$  dan  $b_2$  adalah bias.

$$\mathbf{FFN}(x) = \max(\mathbf{0}, xW_1 + b_1)W_2 + b_2 \quad (4)$$

FFN beroperasi secara independen pada setiap posisi dari urutan *input*, tetapi dengan cara yang identik di semua posisi. Jadi, meskipun setiap posisi diolah secara terpisah, parameter yang sama digunakan untuk setiap posisi dalam urutan (Vasnawi et al., 2017).

#### 2.5.4 Positional Encoding

*Positional encoding* adalah mekanisme yang digunakan untuk menyertakan informasi tentang posisi relatif token dalam urutan *input*. Karena model Transformer tidak memiliki rekurensi atau konvolusi yang secara intrinsik memodelkan urutan atau posisi, diperlukan cara untuk memasukkan informasi posisi agar model dapat memanfaatkan urutan input yang sebenarnya. Seperti yang terlihat pada Gambar 2.4, *Positional encoding* ditambahkan ke vektor *embedding input* pada *encoder* dan *embedding output* pada *decoder*. Persamaan 5 dan 6 merepresentasikan *Positional Encoding* yang merupakan fungsi *sinus* dan *cosinus* di mana *pos* adalah posisi dan *i* adalah dimensi.

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}}) \quad (5)$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}}) \quad (6)$$

## 2.6 Metrik Evaluasi

### 2.6.1 BLEU

BLEU atau *Bilingual Evaluation Understudy* (Papineni et al., 2002) adalah sebuah metrik umum digunakan untuk mengevaluasi sistem *Natural Language Processing* (NLP) yang memproduksi teks bahasa, terutama sistem *machine translation* (MT) and *Natural Language Generation* (NLG). Metrik ini mengukur seberapa banyak *n-gram* dalam teks terjemahan mesin yang cocok dengan *n-gram* terjemahan referensi (biasanya terjemahan manusia) dan sifatnya tidak memperhatikan posisi. Dalam konteks evaluasi BLEU, *n-gram* merujuk pada urutan *n-item* (biasanya kata-kata) yang berurutan dari sebuah teks.

Dalam praktiknya, BLEU memodifikasi cara pengukuran presisi *n-gram* dengan menghindari pemberian skor berlebihan kepada terjemahan yang terlalu sering menggunakan kata-kata tertentu. Dengan kata lain, jumlah tiap *n-gram* pada teks terjemahan dipangkas agar tidak melebihi jumlah maksimum yang ditemukan dalam satu terjemahan referensi. Modifikasi ini disebut *modified n-gram precision* yang proses komputasinya ditunjukkan pada persamaan berikut.

$$p_n = \frac{\sum C \epsilon \{candidates\} \sum n - gram \in C^{Count_{clip}(n-gram)}}{\sum C' \epsilon \{candidates\} \sum n - gram' \in C'^{Count_{clip}(n-gram)}} \quad (7)$$

Setelah melalui proses komputasi  $p_n$ , skor akhir BLEU dikalkulasikan dengan mengambil rata-rata geometris dari fungsi tersebut, lalu dikalikan dengan *brevity penalty factor* (BP). BP memastikan bahwa panjang teks terjemahan tidak terlalu pendek atau terlalu panjang dibandingkan dengan teks referensi (Papineni et al., 2002). Persamaan berikut menunjukkan komputasi BP dan skor akhir BLEU:

$$BP = f(x) = \begin{cases} 1, & \text{jika } c > r \\ e^{(1-r/c)}, & \text{jika } c \leq r \end{cases} \quad (8)$$

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^N w_n \log p_n\right) \quad (9)$$

Di mana  $c$  adalah panjang dari teks terjemahan mesin,  $r$  adalah panjang dari referensi,  $w_n$  adalah bobot yang berikan untuk setiap presisi  $n$ -gram yang dimodifikasi, dan  $N$  biasanya adalah 4, sehingga  $w_n$  bernilai  $\frac{1}{4} = 0.25$ . Metrik BLEU memiliki skala antara 0 hingga 1. Skor yang semakin mendekati 1, semakin serupa dengan terjemahan referensi manusia maka semakin baik penilaianya.

## **BAB III**

### **ANALISIS DAN PERANCANGAN SISTEM**

#### **3.1 Analisis Sistem**

Tahapan analisis sistem bertujuan untuk mengidentifikasi kebutuhan sistem agar dapat beroperasi secara optimal sesuai harapan. Dalam penelitian ini, tahapan analisis terdiri dari analisis masalah dan analisis kebutuhan. Analisis masalah difokuskan untuk mengidentifikasi penyebab dan dampak dari masalah yang ada, sedangkan analisis kebutuhan diarahkan untuk menentukan komponen-komponen esensial yang diperlukan dalam pembangunan sistem secara keseluruhan..

##### **3.1.1 Analisis Masalah**

Pada penelitian ini fokus permasalahan yang dikaji adalah bagaimana kombinasi dari arsitektur EfficientNetV2 dan Transformer dapat diimplementasikan untuk membangun sebuah aplikasi Image Captioning Bahasa Indonesia yang berfungsi dengan baik. Aplikasi *Image Captioning* harus dapat menghasilkan teks deskripsi yang mampu menggambarkan hubungan antar objek di dalam gambar dengan baik. Selain itu, pada saat ini belum ditemukan adanya penelitian pengembangan model *Image Captioning* dalam Bahasa Indonesia yang memanfaatkan kombinasi dari arsitektur EfficientNetV2 dan Transformer.

##### **3.1.2 Analisis Kebutuhan**

Dalam rangka menyelesaikan permasalahan di atas, dirumuskan syarat-syarat atau kebutuhan yang harus dipenuhi oleh sistem yang dibangun. Kebutuhan tersebut terbagi menjadi dua jenis, yakni kebutuhan fungsional dan kebutuhan non-fungsional.

###### **3.1.2.1 Kebutuhan Fungsional**

Kebutuhan fungsional mencakup proses-proses yang dilakukan oleh sistem yang dibangun. Kebutuhan fungsional sistem adalah sebagai berikut.

1. Sistem dapat berjalan di lingkungan web.
2. Sistem dapat menerima masukan berupa gambar yang diunggah oleh pengguna.

3. Sistem mampu mengidentifikasi gambar masukan dan membangkitkan teks deskriptif secara otomatis dalam Bahasa Indonesia untuk menggambarkan objek sesuai dengan gambar yang diberikan.
4. Sistem dapat menampilkan teks deskriptif yang telah dibangkitkan pada halaman web yang dapat dilihat oleh pengguna sistem.

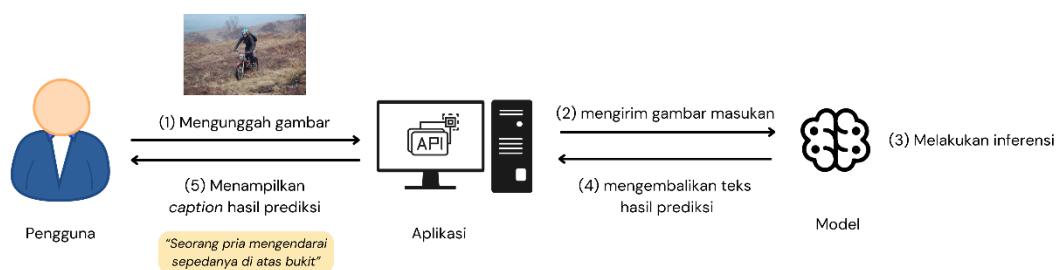
### 3.1.2.2 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional mencakup batasan-batasan kemampuan sistem yang dikembangkan. Kebutuhan non-fungsional sistem adalah sebagai berikut.

1. Teks deskripsi yang dibangkitkan menggunakan Bahasa Indonesia.
2. Sistem memiliki tampilan antarmuka yang baik sehingga mudah untuk dioperasikan.
3. Objek pada gambar masukan yang dapat diidentifikasi adalah objek yang telah dikenali sesuai yang tersedia pada dataset yang digunakan.
4. Sistem mampu memberikan respons yang cepat dalam menghasilkan teks deskripsi.

## 3.2 Diagram Umum Sistem

Diagram umum sistem menggambarkan alur penggunaan dan interaksi yang terdapat pada sistem secara umum. Rancangan sistem yang berupa aplikasi interaktif dijelaskan pada Gambar 3.1.



**Gambar 3.1** Diagram Umum Sistem

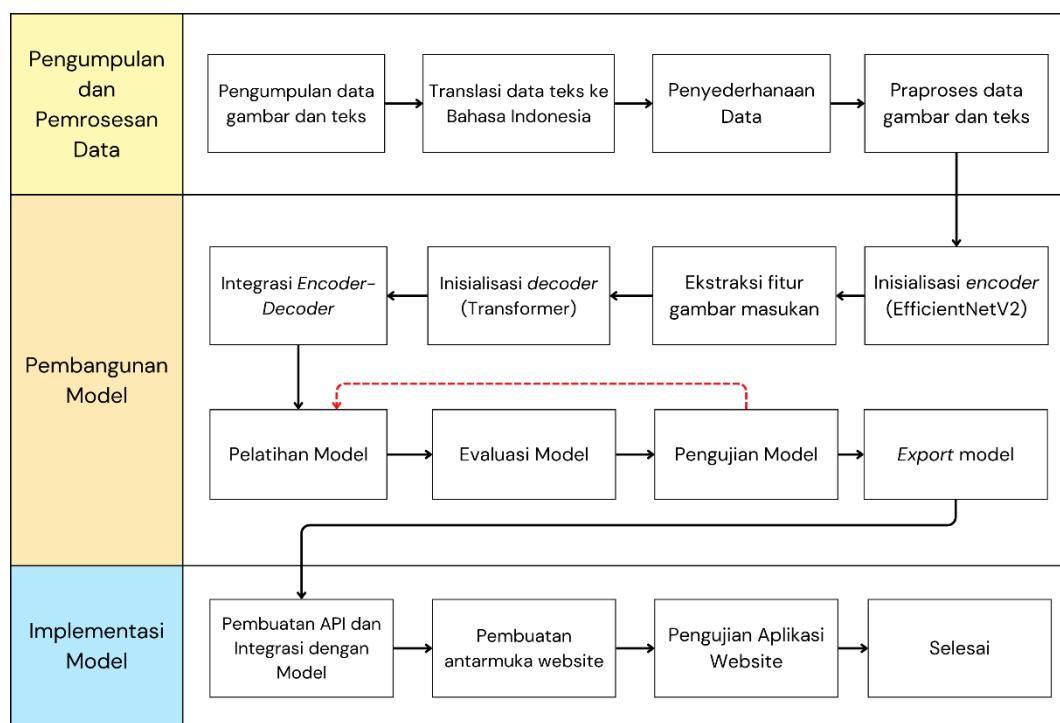
Berikut adalah penjelasan alur penggunaan aplikasi:

1. Pengguna mengakses aplikasi web dan mengunggah gambar sebagai masukan.
2. Laman web mengirimkan data gambar yang diunggah ke API, lalu API meneruskan gambar tersebut ke model.

3. Model melakukan inferensi yaitu proses pembangkitan teks *caption*.
4. Hasil inferensi dikembalikan ke API.
5. Aplikasi web kemudian menampilkan kalimat *caption* yang dapat dilihat oleh pengguna.

### 3.3 Arsitektur Penelitian

Arsitektur penelitian merupakan gambaran proses atau langkah-langkah yang dilakukan untuk membangun keseluruhan sistem. Secara umum, proses pengembangan sistem terbagi ke dalam tiga tahap utama, yaitu pengumpulan dan pemrosesan data, pembangunan model, dan implementasi model. Gambar 3.2 menjelaskan arsitektur penelitian.



**Gambar 3.2** Arsitektur Penelitian

Pada tahapan pertama, dilakukan pengumpulan data gambar dan teks yang berasal dari dataset Flickr8k. *Caption* pada data kemudian diterjemahkan ke dalam Bahasa Indonesia. Selanjutnya, dilakukan proses penyederhanaan data dengan memastikan konsistensi setiap data yang tersedia. Langkah berikutnya adalah praproses data gambar dan teks sehingga data kompatibel sebagai masukan pada jaringan *neural network* yang digunakan.

Pada tahapan kedua, dilakukan pembangunan model yang dimulai dari inisialisasi EfficientNetV2 sebagai *encoder* untuk mengekstrak fitur visual dari data gambar masukan. Setelah itu, inisialisasi Transformer sebagai *decoder* dan dikonfigurasi untuk memproses data teks. *Encoder* dan *decoder* diintegrasikan untuk melakukan *training* model yang mampu mengeluarkan *caption* sesuai gambar masukan. Proses *training* diikuti dengan evaluasi untuk menilai performa *training* model. Selanjutnya, dilakukan pengujian model. Jika hasil pengujian belum memuaskan, proses *training* berpotensi diulang dengan konfigurasi berbeda hingga model dapat lebih dioptimalkan. Model kemudian diekspor untuk diimplementasikan ke dalam API.

Tahapan terakhir, dilakukan pembuatan API dan antarmuka *website*, yang kemudian diintegrasikan dengan model yang telah dibangun sebelumnya. Selanjutnya, dilakukan pengujian aplikasi untuk mengukur fungsionalitas aplikasi berjalan dengan baik. Proses akhirnya menghasilkan sebuah aplikasi yang siap digunakan untuk menerjemahkan gambar menjadi teks *caption*.

### 3.4 Pengumpulan dan Pemrosesan Data

#### 3.4.1 Pengumpulan Data

Proses pengumpulan data gambar dan teks dilakukan dengan mengunduh dataset Flickr8k. Dataset Flickr8k pertama kali dikemukakan pada makalah yang berjudul "Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics" (M. Hodosh et al., 2013). Dataset ini berisikan kumpulan gambar yang di mana setiap gambar disertai dengan lima kalimat yang mendeskripsikan objek-objek pada gambar. Dataset ini terdiri dari total 8.091 gambar dan 40.455 kalimat.

Peneliti mengumpulkan data gambar dan teks Flickr8k dari tautan berikut: <https://github.com/jbrownlee/Datasets/releases/tag/Flickr8k>. Gambar yang tersedia pada dataset ini merepresentasikan objek yang umum ditemui pada aktivitas sehari-hari, seperti kegiatan manusia, hewan peliharaan, dan pemandangan dan sebagainya. Data teks yang ada pada dataset ini disusun dalam Bahasa Inggris. Sehingga, pada penelitian ini, seluruh data teks diterjemahkan ke dalam Bahasa Indonesia.

Setelah berhasil mengumpulkan data, tahap berikutnya adalah melakukan pemrosesan data. Langkah-langkah dalam pemrosesan data meliputi penyederhanaan data, prapemrosesan gambar, dan prapemrosesan kalimat. Tujuan dari pemrosesan data adalah untuk mempersiapkan data agar lebih mudah digunakan dalam proses pembangunan sistem pada tahap berikutnya.

### 3.4.2 Penyederhanaan Data

Tujuan dari penyederhanaan data adalah untuk menjaga konsistensi data serta proses pengolahan data di tahap selanjutnya. Penyederhanaan data dilakukan dengan memastikan bahwa setiap gambar pada dataset memiliki tepat lima kalimat yang mendeskripsikannya. Selain itu, konten pada data teks diubah sedemikian rupa sehingga memiliki bentuk teks yang sederhana. Rincian tahap penyederhanaan data yang dilakukan adalah sebagai berikut:

- Menghapus beberapa baris dalam data teks yang berisi nama *file* gambar yang tidak ada dalam dataset gambar.
- Menambahkan baris *header* (nama kolom) pada dataset
- Menghilangkan indeks *integer* pada kolom nama *file* gambar

### 3.4.3 Prapemrosesan Gambar

Tujuan dari prapemrosesan gambar adalah untuk menyiapkan data gambar yang kompatibel dengan arsitektur jaringan EfficientNetV2. Tahapan praproses gambar dijelaskan pada langkah-langkah sebagai berikut:

#### 1. Penyesuaian Ukuran Gambar

Dataset terdiri dari gambar-gambar yang memiliki ukuran dan orientasi yang beragam. Oleh karena itu, perlu dilakukan proses penyeragaman dimensi yang sesuai dengan spesifikasi gambar masukan yang dibutuhkan oleh arsitektur EfficientNetV2.

#### 2. Pemangkasan Gambar

Setelah ukuran data gambar diubah, langkah selanjutnya memangkas gambar dengan teknik *center cropping*. Tujuannya untuk menjamin ukuran gambar sesuai persis dengan yang dibutuhkan oleh model dan meminimalisir terjadinya kesalahan.

#### 3. Pembentukan Tensor

Berikutnya, data gambar pada dataset ditransformasi ke dalam bentuk tensor agar dapat diproses oleh jaringan *neural network* ketika diterima sebagai masukan oleh model.

#### 4. Normalisasi Gambar

Normalisasi gambar dilakukan dengan tujuan mengubah nilai piksel pada gambar sehingga memiliki jarak dari 0 hingga 1. Tahapan proses ini adalah dengan mengurangi nilai rata-rata piksel, lalu setiap nilai piksel dibagi dengan nilai standar deviasi. Pada implementasinya, nilai *mean* yang digunakan adalah [0.485, 0.456, 0.406], sedangkan nilai standar deviasi yang digunakan adalah [0.229, 0.224, 0.225] untuk setiap *channel Red, Green, Blue* (RGB) secara berurutan.

##### 3.4.4 Prapemrosesan Kalimat

Data teks dalam dataset merupakan kalimat yang berfungsi memberikan keterangan atas objek-objek dalam gambar. Kalimat-kalimat ini masih dalam bentuk mentah dan memerlukan prapemrosesan sebelum dapat diolah lebih lanjut. Tahap prapemrosesan ini melibatkan beberapa langkah sebagai berikut:

###### 1. Standarisasi Kalimat

Standarisasi kalimat dilakukan pada awal pengolahan data guna menyeragamkan format penulisan caption pada dataset. Tahapan ini meliputi pengubahan semua huruf dalam kalimat menjadi huruf kecil serta memisahkan setiap kata dengan tanda baca.

###### 2. Penambahan Token Khusus

Pada dataset, *caption* yang tersedia masih berbentuk data mentah yang hanya berisikan kata-kata deskriptif. Dalam rangka untuk memudahkan model dalam melakukan penyusunan kalimat, ditambahkan dua token khusus, yaitu “<start>” dan “<end>”. Token “<start>” ditambahkan pada bagian awal kalimat, sedangkan token “<end>” ditambahkan pada akhir kalimat. Proses penambahan dilakukan secara menyeluruh terhadap seluruh data *caption* pada dataset.

### 3. Tokenisasi

Tokenisasi adalah proses pemisahan kata dalam kalimat menjadi satu kata tersendiri. Setiap kata tersebut diubah menjadi bentuk *integer* unik.

### 4. *Text to Sequence*

Token atau kata diubah ke dalam bentuk indeks *integer* yang berurutan. Tahap ini akan menghasilkan sebuah vektor yang masing-masing *integer* merepresentasikan satu token.

### 5. *Pad Sequence*

Tahap ini terdiri dari dua proses, yaitu menambah *integer* 0 yang berperan sebagai *padding* pada vektor dengan jumlah token yang lebih kecil dari jumlah yang ditentukan, serta pengurangan atau pemotongan pada bagian akhir vektor jika panjangnya melebihi jumlah yang ditentukan.

### 6. Pembangunan Kosakata

Kosakata atau *vocabulary* adalah data yang dibangun dengan mengumpulkan kata-kata pada saat proses tokenisasi di tahap sebelumnya. Data ini berisikan kata-kata unik yang urutannya ditentukan berdasarkan frekuensi kemunculan.

#### 3.4.5 Pembagian Data

Total data yang terdapat pada dataset berjumlah 8.091 gambar dan 40.455 kalimat. Mengikuti pemisahan data oleh Hodosh et al., (2013), penelitian ini hanya menggunakan sejumlah 8.000 data gambar dan 40.000 kalimat. Data tersebut dibagi ke dalam tiga bagian: data latih, data validasi, dan data uji dengan proporsi pembagian data ditunjukkan dapat dilihat pada Tabel 3.1.

**Tabel 3.1** Proporsi Pembagian Data

Bagian	Gambar	Kalimat
Data latih	6.000	30.000
Data validasi	1.000	5.000
Data uji	1.000	5.000

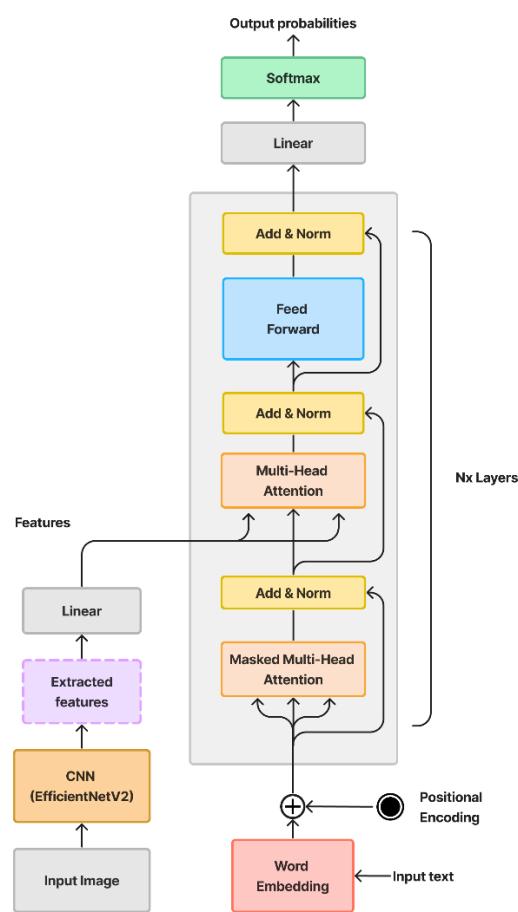
### 3.5 Pembangunan Model

Tahap berikutnya adalah pembuatan model *deep learning* berbasis CNN-Transformer. Model CNN bertugas untuk melakukan ekstraksi fitur visual dari

gambar, sedangkan Transformer berfungsi untuk memproses fitur visual tersebut untuk menghasilkan sebuah teks *caption*.

### 3.5.1 Inisialisasi Model

Model yang dibangun merupakan kombinasi CNN dengan Transformer. Arsitektur Transformer yang terdiri dari dua bagian, yaitu *encoder* dan *decoder*. Pada penelitian ini, CNN menggantikan peran dari *encoder* pada Transformer sebagai model yang mengekstraksi fitur gambar. Gambar 3.3 menunjukkan ilustrasi dari rancangan arsitektur model yang digunakan pada penelitian ini.



**Gambar 3.3** Ilustrasi Arsitektur *encoder* (kiri) dan *decoder* (kanan)

Pada bagian *encoder*, model CNN dirancang untuk menerima masukan berupa gambar dari dataset. Setelah menerima masukan, CNN kemudian melakukan inferensi guna mendapatkan fitur visual dari gambar. Berikutnya, fitur visual tersebut diteruskan ke *decoder* sehingga dapat dilakukan proses *decoding* untuk memprediksi kalimat deskriptif.

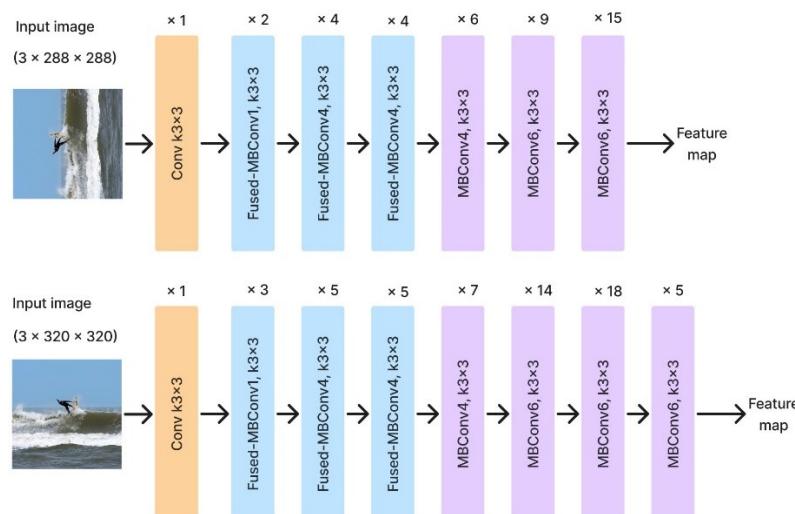
### 3.5.1.1 Encoder

Pada sisi *encoder*, EfficientNetV2 adalah arsitektur CNN yang digunakan untuk ekstraksi fitur gambar. Berdasarkan makalah “EfficientNetV2: Smaller Models and Faster Training” (Tan & Le, 2021), arsitektur ini dikembangkan dengan tiga variasi berdasarkan jumlah parameternya, yaitu variasi skala kecil, skala medium, dan skala besar. Dalam penelitian ini, model yang digunakan adalah model skala kecil (EfficientNetV2-S) dan skala medium (EfficientNetV2-M). Kedua model ini telah dilatih dengan dataset ImageNet dan spesifikasi dari hasil pelatihan ditunjukkan oleh Tabel 3.2.

**Tabel 3.2** Spesifikasi EfficientNetV2-S dan EfficientNetV2-M

<b>Spesifikasi</b>	<b>EfficientNetV2-S</b>	<b>EfficientNetV2-M</b>
<i>Parameters</i>	23.9M	53.2M
<i>Input size</i>	288 x 288 piksel	320 x 320 piksel

Pada Tabel 3.2, *Parameters* menggambarkan jumlah bobot yang dapat dipelajari oleh model, sedangkan *input size* menunjukkan ukuran gambar masukan yang direkomendasikan untuk melakukan ekstraksi fitur gambar, karena sesuai dengan ukuran gambar yang digunakan pada masa pelatihan model. EfficientNetV2 terdiri dari kumpulan blok MBConv (*Mobile Inverted Bottleneck Convolution*) dan Fused-MBConv.



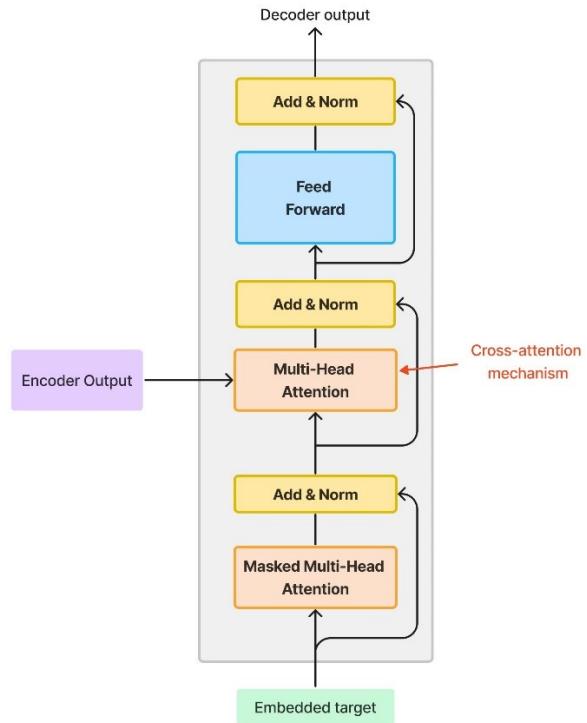
**Gambar 3.4** Ilustrasi Arsitektur EfficientNetV2-S (atas) dan EfficientNetV2-M (bawah)

Gambar 3.4 menunjukkan gambaran lapisan-lapisan penyusun EfficientNetV2-S dan EfficientNetV2-M dalam menghasilkan fitur visual dari gambar. Arsitektur EfficientNetV2-S terdiri dari 1 blok konvolusi, 3 blok Fused-MBConv, dan 3 blok MBConv. EfficientNetV2-M memiliki arsitektur yang sama, namun dengan tambahan 1 blok MBConv di lapisan terakhir. Masing-masing blok tersebut disusun berulang kali dengan jumlah pengulangan yang bervariasi. Setelah melalui operasi di dalam lapisan-lapisan tersebut, model akan menghasilkan *feature map* sebagai hasil pengkodean gambar menjadi bentuk vektor. Dimensi *feature map* terdiri dari *batch\_size*, *number of channels*, *height*, dan *width*. Berikut adalah penjelasan dari masing-masingnya:

1. *Batch size* menunjukkan jumlah gambar yang diproses secara bersamaan dalam satu *batch*.
2. *Number of channels* merujuk pada jumlah filter atau fitur yang diekstraksi oleh model pada lapisan terakhir. Setiap *channel* mewakili satu set fitur yang merepresentasikan informasi tertentu dari gambar.
3. *Height* dan *width* adalah dimensi spasial dari *feature map*. Dimensi ini bergantung pada ukuran *input* yang diberikan dan transformasi yang diterapkan selama proses pelatihan dan inferensi.

### 3.5.1.2 Decoder

Penelitian ini menggunakan arsitektur *decoder* dari Transformer. Masukan yang diterima oleh *decoder* merupakan fitur visual gambar yang dihasilkan oleh *encoder* CNN. Selain itu, *decoder* juga membutuhkan *input* berupa sebuah kalimat yang disebut sebagai *target sequence*. Pada masa pelatihan, *target sequence* adalah kalimat referensi dari dataset yang merupakan deskripsi dari gambar yang sedang dipelajari model. Sementara pada inferensi, *target sequence* awalnya hanya berisikan token '<start>' dan tugas *decoder* adalah memprediksi kata-kata berikutnya sesuai dengan gambar masukan yang diberikan. Gambar 3.5 mengilustrasikan arsitektur *decoder* dari Transformer.



**Gambar 3.5** Ilustrasi Arsitektur *Decoder* dari Transformer

Transformer *decoder* terdiri dari lapisan *multi head-attention*, lapisan normalisasi, dan lapisan *feed-forward*. Mekanisme *attention* yang terdapat pada *multi-head attention* terdiri dari dua jenis, yaitu *self-attention* dan *cross-attention*. Berikut adalah penjelasan dari masing-masing lapisan:

1. *Masked Multi-Head Attention*, merupakan lapisan yang melakukan mekanisme *self-attention*, namun hanya fokus pada token sebelumnya dalam urutan *caption* saat memprediksi token berikutnya. Tujuannya adalah untuk menjaga keakuratan dan logika dalam urutan teks yang dihasilkan.
2. Normalisasi, berfungsi untuk menstabilkan dan mempercepat proses *training* dengan menormalkan *output tensor* yang dihasilkan lapisan sebelumnya.
3. *Cross-attention*, merupakan lapisan yang menggabungkan informasi fitur visual gambar dengan urutan *caption* yang sedang dibangun. Mekanisme ini memungkinkan *decoder* untuk “memperhatikan” bagian-bagian tertentu dari gambar saat proses pembentukan *caption*.
4. *Feed-forward*, terdiri dari lapisan linear (*fully-connected*) dan fungsi aktivasi ReLU. Masukan dari lapisan ini adalah nilai *tensor* dari lapisan

sebelumnya dan berfungsi untuk memperkaya representasi fitur dengan menerapkan transformasi non-linear.

### 3.5.2 Pelatihan dan Evaluasi Model

Tahap selanjutnya adalah melakukan pelatihan (*training*) model terhadap dataset yang telah disiapkan sebelumnya. Pada penelitian ini, ada dua arsitektur jaringan yang saling diintegrasikan dalam kerangka *encoder-decoder*. Pada sisi *encoder*, digunakan arsitektur EfficientNetV2 dengan skala kecil dan skala medium. Model EfficientNetV2 tersebut telah dilatih dengan dataset ImageNet, sehingga sudah memiliki bobot yang cukup untuk mengekstrak fitur visual dari gambar. Pada sisi *decoder*, model Transformer akan dilatih dengan memanfaatkan fitur visual gambar yang diekstrak oleh CNN. Pada waktu yang bersamaan, Transformer juga diberikan data teks yang menjadi *target sequence*, sehingga model dapat mengenali fitur-fitur gambar dan bagaimana penerjemahannya ke dalam kalimat.

Proses pelatihan model berlangsung secara iteratif dan terlaksana secara bergantian dengan evaluasi model. Proses evaluasi bertujuan untuk menilai seberapa baik model dalam mempelajari fitur dan melakukan prediksi. Penilaian dilakukan dengan menghitung nilai bobot dan parameter pelatihan, serta menggunakan metrik validasi seperti *loss function* untuk menghitung perbandingan antara *training loss* dan *validation loss*. Selain itu, untuk mendapatkan gambaran yang lebih baik tentang kinerja model dalam kondisi yang berbeda, digunakan teknik validasi silang dengan bagian data yang lain. Hal ini berguna untuk menilai apakah model dapat bekerja dengan efektif dalam menangani data baru yang belum dikenali sebelumnya. Validasi silang ini dilakukan terhadap data validasi yang proporsinya sebagaimana dijelaskan pada Tabel 3.1 mengenai proporsi pembagian data.

### 3.5.3 Pengujian Model

Setelah model dikembangkan, tahap berikutnya adalah mengevaluasi kinerja model dalam menghasilkan teks *caption* berdasarkan gambar yang diberikan. Evaluasi model dilakukan dengan menggunakan metrik evaluasi yang umum digunakan, yaitu BLEU. Metrik evaluasi BLEU dihitung dengan membandingkan

kata-kata yang dihasilkan model dengan target sebenarnya. Skor BLEU memiliki skala antara 0 hingga 1. Semakin mendekati 1, maka semakin serupa dengan kalimat referensi manusia maka semakin baik penilaianya. Proses evaluasi ini dilakukan terhadap 1000 data uji yang merupakan bagian dari dataset Flickr8k, namun belum pernah dikenali sebelumnya oleh model pada masa pelatihan.

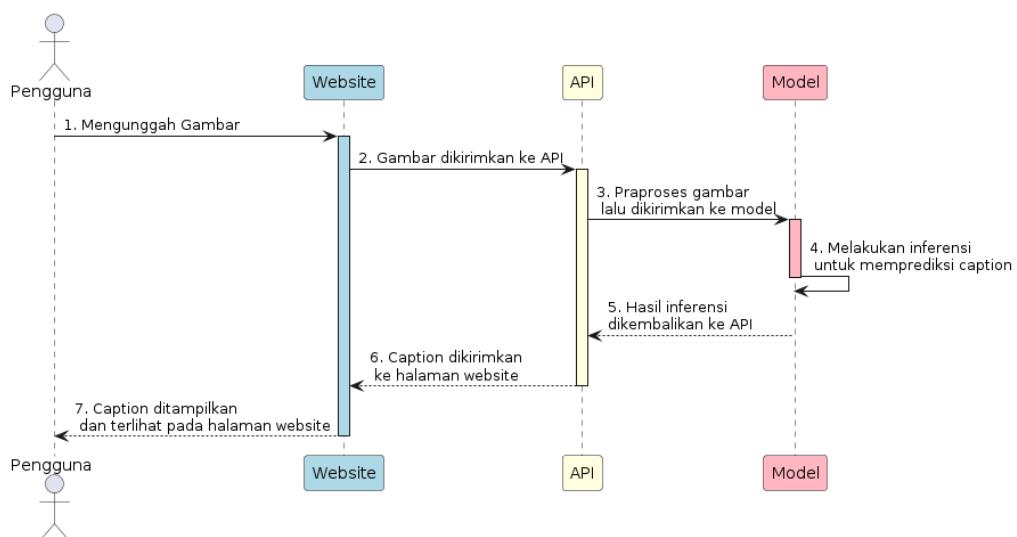
### 3.5.4 Ekspor Model

Tahap ekspor model merupakan tahapan terakhir dalam proses pembangunan model. Model yang telah selesai dilatih akan diekspor ke dalam *file* dengan format ‘.pth’. *File* tersebut berisi *state dictionary* yang mencakup semua parameter yang telah dipelajari oleh model, seperti bobot dan bias. *State dictionary* ini merupakan representasi dari kondisi internal model pada saat tertentu. Di sini, *state dictionary* yang disimpan adalah kondisi terbaik di mana nilai *validation loss* adalah yang paling kecil.

## 3.6 Implementasi Model

### 3.6.1 Pembuatan API dan Integrasi dengan Model

Setelah model berhasil diekspor, langkah berikutnya adalah mengimplementasikan model tersebut dengan mengintegrasikannya ke sebuah API. API dibuat sebagai jembatan antara aplikasi *website* dengan model yang telah disimpan. Hal ini memungkinkan model ini untuk dapat diakses dan melakukan inferensi secara interaktif.

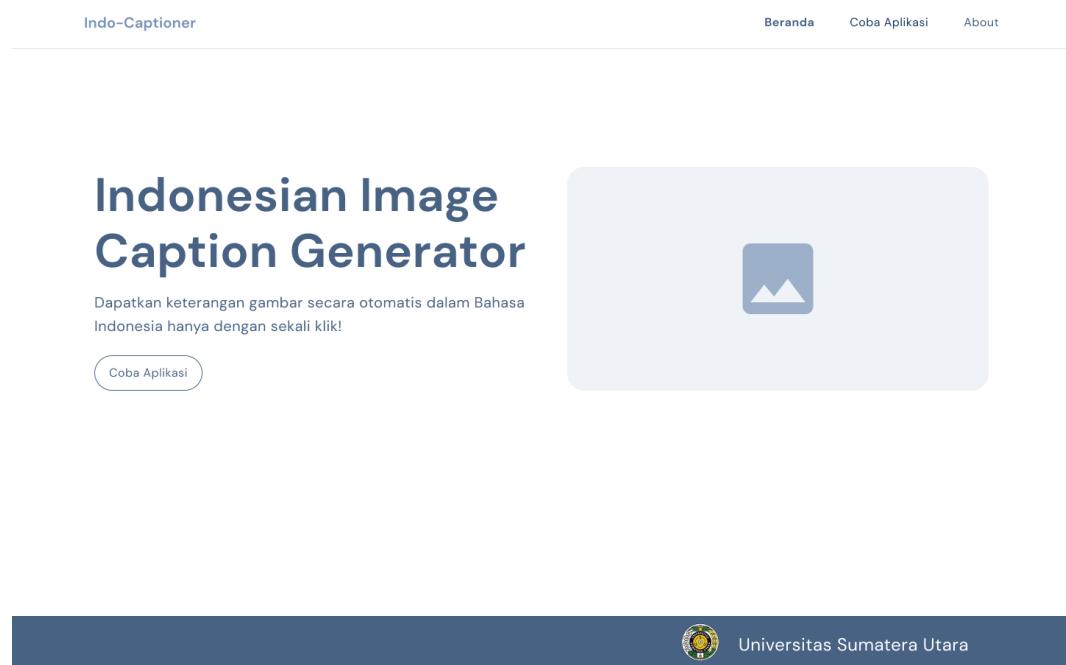


**Gambar 3.6 Sequence Diagram**

Gambar 3.6 adalah *sequence diagram* yang menjelaskan alur interaksi antara pengguna, antarmuka *website*, API, dan model *deep learning* dalam proses pengunggahan gambar hingga hasil inferensi ditampilkan kepada pengguna. Proses ini dimulai ketika pengguna mengunggah gambar melalui *website*. *Website* kemudian menerima gambar tersebut dan mengirimkannya ke API di *server*. Setelah itu, API melakukan praproses terhadap gambar, lalu dikirimkan ke model. Model kemudian melakukan inferensi untuk memprediksi *caption* atau keterangan yang sesuai dengan gambar tersebut. Setelah inferensi selesai, model mengembalikan hasil berupa *caption* ke API. API kemudian mengirimkan *caption* tersebut ke antarmuka *website* untuk ditampilkan kepada pengguna.

### 3.6.2 Perancangan Antarmuka *Website*

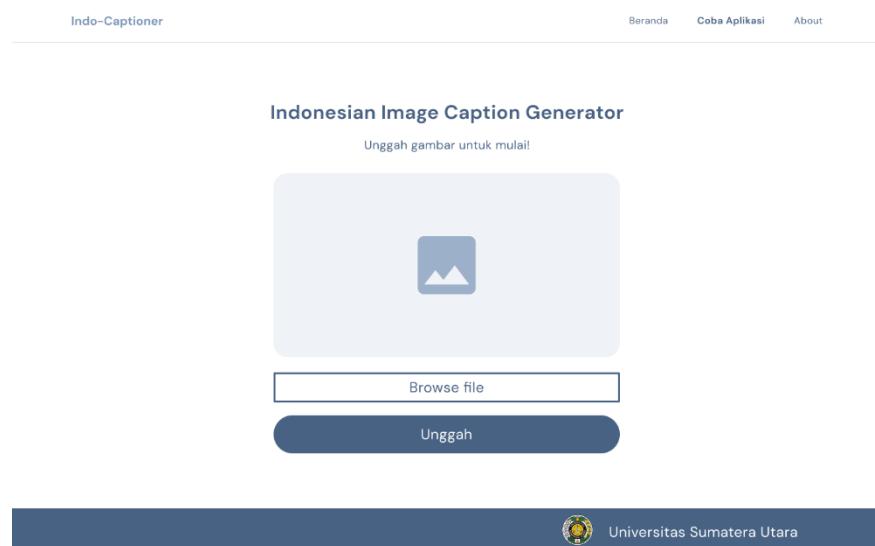
Perancangan antarmuka *website* dilakukan untuk mendapatkan gambaran dari implementasi sistem yang dibangun. *Website* dirancang untuk memiliki tiga halaman utama, yakni halaman beranda, halaman uji aplikasi, dan halaman *about*.



**Gambar 3.7** Rancangan Antarmuka Halaman Beranda

Gambar 3.7 adalah rancangan antarmuka pada halaman beranda. Halaman beranda adalah halaman pertama yang ditampilkan ketika *website* diakses. Halaman

ini berisikan informasi singkat dan perkenalan terhadap pengguna mengenai kegunaan *website*.



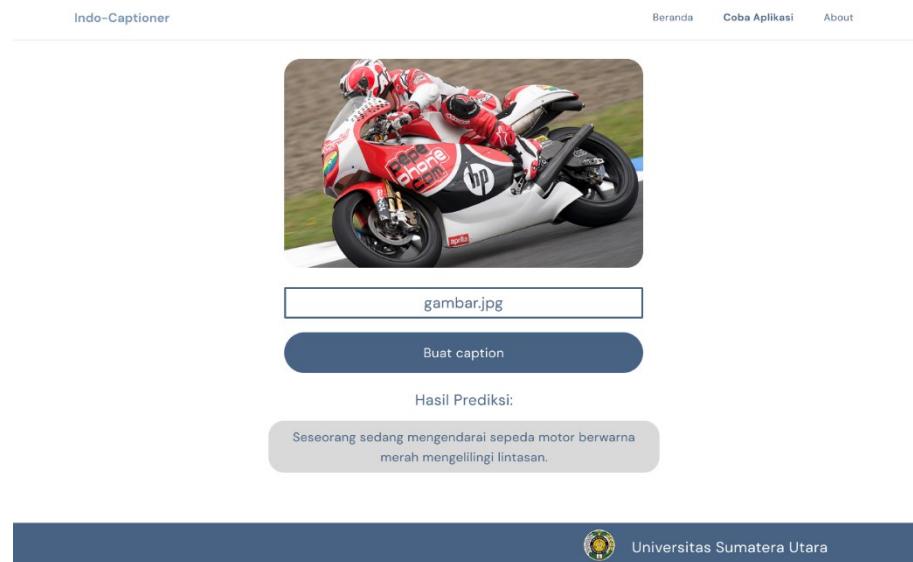
**Gambar 3.8** Rancangan Antarmuka untuk *Image Caption Generator*



**Gambar 3.9** Rancangan Antarmuka untuk Halaman Image Caption Generator (Lanjutan)

Gambar 3.8 dan 3.9 menampilkan halaman prediksi *caption* otomatis yang merupakan fitur utama pada *website*. Pada halaman ini, pengguna dapat memilih dan mengunggah sebuah gambar. Setelah memilih gambar, pengguna dapat memulai proses prediksi *caption* atau mengganti gambar yang dipilih. Jika pengguna memilih untuk membuat *caption*, aplikasi akan mengeluarkan kalimat

hasil inferensi model yang telah diintegrasikan seperti yang dapat dilihat pada Gambar 3.10.



**Gambar 3.10** Rancangan Antarmuka Halaman Image Caption Generator (Lanjutan)

Halaman selanjutnya adalah halaman *about* yang berisikan informasi singkat mengenai kegunaan *website*, teknologi yang digunakan pada pengembangan aplikasi, dan profil peneliti. Desain halaman ini ditunjukkan Gambar 3.9.



**Gambar 3.11** Rancangan Antarmuka Halaman *About*

## **BAB IV**

### **IMPLEMENTASI DAN PENGUJIAN SISTEM**

#### **4.1 Lingkungan Implementasi**

Pada penelitian ini, sistem dibangun secara terstruktur sesuai dengan perancangan yang telah dijelaskan sebelumnya. Dalam rangka membantu proses pembangunan sistem, dibutuhkan beberapa perangkat, baik perangkat keras maupun perangkat lunak.

##### **4.1.1 Perangkat Keras**

Komponen perangkat keras yang digunakan pada proses pengembangan model *deep learning* umumnya memiliki pengaruh yang cukup signifikan terhadap durasi pelatihan. Hal ini dikarenakan komponen seperti GPU dapat memproses data tensor dengan lebih efisien dibandingkan hanya menggunakan komputasi CPU. Adapun perangkat keras digunakan pada penelitian ini terdiri dari sebuah personal komputer dan *cloud-based computing* (Google Colab). Spesifikasi masing-masing perangkat adalah sebagai berikut:

###### **1. Personal Komputer**

- a. CPU : Intel Core i3-12100F
- b. GPU : NVIDIA GeForce GTX 1660 Ti
- c. RAM : 32 GB
- d. Sistem Operasi : Microsoft Windows 11 Home

###### **2. Google Colab**

- a. GPU : T4 GPU
- b. *System* RAM : 51.0 GB
- c. GPU RAM : 15.0 GB
- d. *Disk* : 201.2 GB

##### **4.1.2 Perangkat Lunak**

Dalam pembangunan sistem digunakan perangkat lunak yang terdiri dari beberapa hal dengan rincian sebagai berikut:

- *Text Editor* dan IDE : Visual Studio Code dan Jupyter Notebook

- Bahasa pemrograman : Python
- Pustaka utama :
  - Modul Pytorch untuk inisialisasi, pengembangan, pelatihan, dan pengujian model.
  - Modul torchtext dan pandas membantu proses pengolahan dan prapemrosesan data teks
  - Modul timm untuk inisialisasi model CNN EfficientNetV2
  - Modul sacrebleu (Post, 2018) untuk perhitungan skor BLEU
  - Modul matplotlib untuk visualisasi hasil pengujian sistem
  - Modul Flask untuk mengimplementasikan model ke dalam aplikasi API
- *Tech Stack Web* :
  - HTML – digunakan untuk struktur dasar halaman web
  - JavaScript – digunakan sebagai integrasi halaman web dengan API
  - Bootstrap – *framework* CSS yang digunakan untuk desain responsif dan modern

## 4.2 Implementasi Pengumpulan dan Pemrosesan Data

### 4.2.1 Pengumpulan Data

Data pada penelitian ini dikumpulkan dari dataset Flickr8k (Hodosh, M. et al., 2013) yang terdiri dari 8.000 gambar. Setiap gambar dalam kumpulan data ini disertai dengan lima deskripsi gambar berbeda yang diperoleh melalui *crowdsourcing*. Kalimat yang terdapat pada dataset ini disusun dengan Bahasa Inggris yang kemudian diterjemahkan ke dalam Bahasa Indonesia. Tabel 4.1 menampilkan contoh data gambar dan kalimat yang berhasil dikumpulkan.

**Tabel 4.1** Contoh Data Gambar dan Kalimat

Gambar	Kalimat
	Seorang pria berdiri sendirian di atas tebing berbatu dan memandang ke arah hutan belantara
	Seorang lelaki bercelana pendek dan berjaket hitam berdiri di atas batu besar memandang ke bawah bukit dan lembah

	Seorang pria sedang berdiri di atas batu besar yang menghadap ke lembah
	Seorang pria berdiri di atas gunung batu yang besar
	Manusia berdiri di tepi batu besar menghadap pemandangan indah
	Seorang pria memainkan lagu dengan gitar untuk kucingnya
	Seorang pria memainkan gitar kuning sementara seekor kucing mengawasinya
	Seorang pria memainkan gitar kuningnya sambil menatap kucingnya
	Seorang pria, duduk di depan komputernya, bermain gitar dengan kucingnya
	Pria itu sedang bermain gitar dan duduk dengan seekor kucing

Adapun struktur pada data teks terdiri dari dua kolom. Kolom pertama merupakan nama berkas gambar dan kolom kedua adalah kalimat yang mendeskripsikan gambar tersebut. Gambaran konten yang terdapat pada data teks ditunjukkan oleh Gambar 4.1.

0	1
0 1000268201_693b08cb0e.jpg#0	Seorang anak berpakaian merah muda sedang menaiki tangga di jalan masuk.
1 1000268201_693b08cb0e.jpg#1	Seorang gadis memasuki sebuah bangunan kayu.
2 1000268201_693b08cb0e.jpg#2	Seorang gadis kecil naik ke rumah bermain kayu.
3 1000268201_693b08cb0e.jpg#3	Seorang gadis kecil menaiki tangga menuju rumah bermainnya.
4 1000268201_693b08cb0e.jpg#4	Seorang gadis kecil berpakaian merah muda masuk ke kabin kayu.
...	...
40450 997722733_0cb5439472.jpg#0	Seorang pria berkemeja merah muda memanjat permukaan batu
40451 997722733_0cb5439472.jpg#1	Seorang pria sedang memanjat tebing tinggi di udara.
40452 997722733_0cb5439472.jpg#2	Seseorang dengan kemeja merah memanjat permukaan batu yang ditutupi pegangan bantuan.
40453 997722733_0cb5439472.jpg#3	Seorang pemanjat tebing berbaju merah.
40454 997722733_0cb5439472.jpg#4	Seorang pemanjat tebing berlatih di dinding panjat tebing.

**Gambar 4.1** Gambaran Konten pada Data Teks

#### 4.2.2 Penyederhanaan Data

Penyederhanaan data adalah tahapan pertama dalam proses pemrosesan data. Data yang disederhanakan adalah data teks yang memiliki struktur tabel dengan dua kolom, yaitu kolom nama *file* gambar dan kalimat *caption*. Gambar 4.2 merupakan perbandingan sebagian data sebelum dan sesudah dilakukan penyederhanaan data.



	filename	captions
0	1000268201_693b08cb0e.jpg#0	Seorang anak dengan gaun merah muda sedang menaiki satu set tangga di pintu masuk.
1	1000268201_693b08cb0e.jpg#1	Seorang anak perempuan masuk ke dalam sebuah bangunan kayu.
2	1000268201_693b08cb0e.jpg#2	Seorang gadis kecil memanjat ke dalam rumah bermain kayu.
3	1000268201_693b08cb0e.jpg#3	Seorang gadis kecil menaiki tangga menuju rumah bermainnya.
4	1000268201_693b08cb0e.jpg#4	Seorang gadis kecil dengan gaun merah muda masuk ke dalam kabin kayu.
...	...	...
40444	997722733_0cb5439472.jpg#0	Seorang pria dengan kemeja merah muda memanjat tebing
40445	997722733_0cb5439472.jpg#1	Seorang pria memanjat tebing tinggi di udara.
40446	997722733_0cb5439472.jpg#2	Seseorang dengan kemeja merah memanjat permukaan batu yang ditutupi dengan pegangan bantuan.
40447	997722733_0cb5439472.jpg#3	Seorang pemanjat tebing dengan kemeja merah.
40448	997722733_0cb5439472.jpg#4	Seorang pemanjat tebing sedang berlatih di dinding panjat tebing.
40449 rows × 2 columns		

	filename	captions
0	1000268201_693b08cb0e.jpg	Seorang anak dengan gaun merah muda sedang menaiki satu set tangga di pintu masuk.
1	1000268201_693b08cb0e.jpg	Seorang anak perempuan masuk ke dalam sebuah bangunan kayu.
2	1000268201_693b08cb0e.jpg	Seorang gadis kecil memanjat ke dalam rumah bermain kayu.
3	1000268201_693b08cb0e.jpg	Seorang gadis kecil menaiki tangga menuju rumah bermainnya.
4	1000268201_693b08cb0e.jpg	Seorang gadis kecil dengan gaun merah muda masuk ke dalam kabin kayu.
...	...	...
40439	997722733_0cb5439472.jpg	Seorang pria dengan kemeja merah muda memanjat tebing
40440	997722733_0cb5439472.jpg	Seorang pria memanjat tebing tinggi di udara.
40441	997722733_0cb5439472.jpg	Seseorang dengan kemeja merah memanjat permukaan batu yang ditutupi dengan pegangan bantuan.
40442	997722733_0cb5439472.jpg	Seorang pemanjat tebing dengan kemeja merah.
40443	997722733_0cb5439472.jpg	Seorang pemanjat tebing sedang berlatih di dinding panjat tebing.
40444 rows × 2 columns		

**Gambar 4.2** Ilustrasi Perbandingan Data Setelah Penyederhanaan Data

Perubahan terlihat pada Gambar 4.2 yakni ditambahkannya nama kolom *filename* dan *captions*, hilangnya penomoran indeks gambar pada kolom *filename*, dan berkurangnya ukuran data dari sebelumnya berjumlah 40.450 menjadi 40.445 baris. Pengurangan ukuran data ini disebabkan ditemukannya lima baris pada dataset, di mana *file* gambar yang dituliskan di baris tersebut tidak ditemukan pada direktori gambar. Dengan demikian, lima baris tersebut dihapus dari dataset.

#### 4.2.3 Prapemrosesan Gambar

Prapemrosesan gambar terdiri dari beberapa tahapan, seperti penyesuaian ukuran gambar, pemangkasan gambar, pembentukan tensor, dan normalisasi gambar. Tahapan tersebut diberlakukan terhadap semua data gambar yang ada pada dataset sebelum dilakukan pelatihan dan evaluasi. Gambar 4.3 menunjukkan

perbandingan antara data gambar sebelum dan sesudah dilakukan tahap-tahap prapemrosesan.

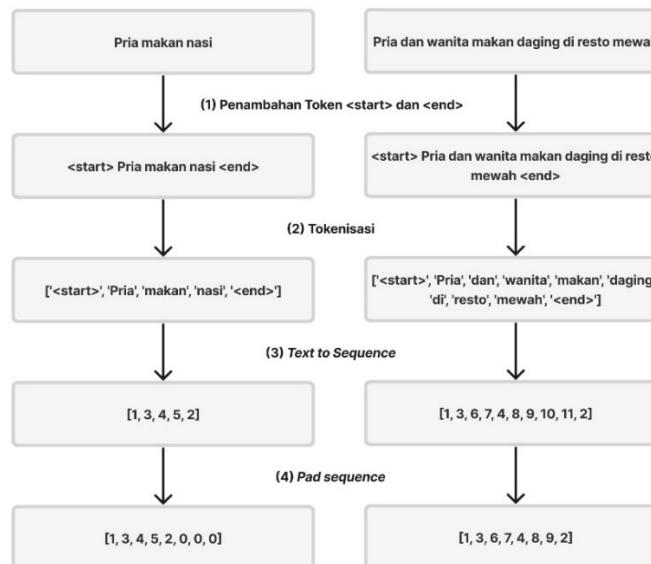


**Gambar 4.3** Ilustrasi Hasil Prapemrosesan Terhadap Data Gambar

Hasil akhir dari tahapan ini adalah seluruh gambar ditransformasi menjadi bentuk *tensor*. *Tensor* yang dihasilkan merepresentasikan gambar sebelum dilakukan proses ekstraksi fitur gambar.

#### 4.2.4 Prapemrosesan Kalimat

Tahapan prapemrosesan kalimat terdiri dari standarisasi kalimat, penambahan token khusus, tokenisasi, *text to sequence*, *pad sequence*, dan pembangunan kosakata. Ilustrasi hasil prapemrosesan kalimat hingga tahap *pad sequence* ditunjukkan pada Gambar 4.4.

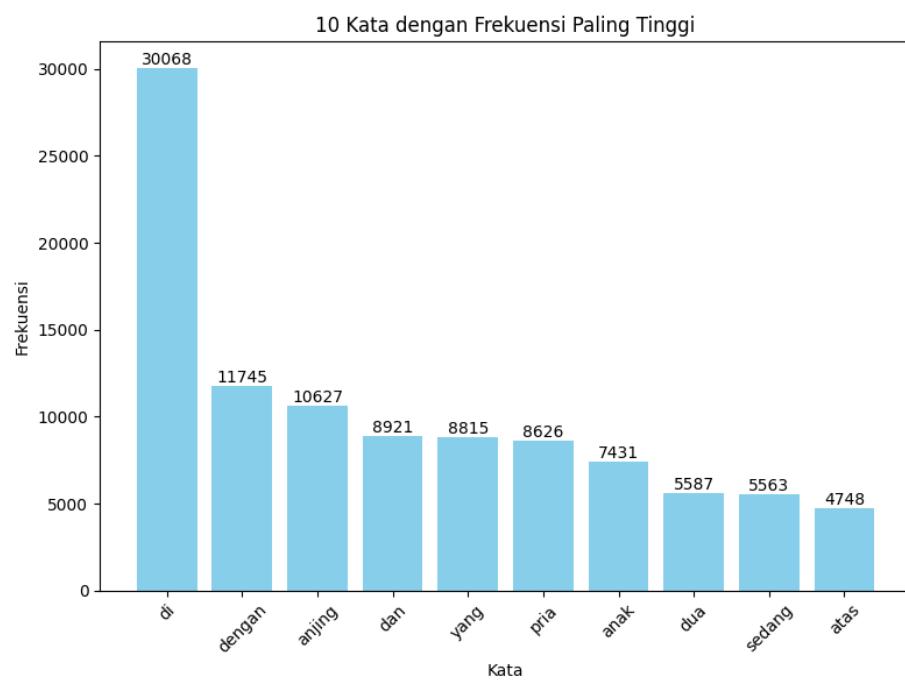


**Gambar 4.4** Ilustrasi Hasil Prapemrosesan Kalimat

Pada Gambar 4.4, diasumsikan indeks token <pad>, <start> dan <end> adalah 0, 1, dan 2 secara berturut serta *max\_seq\_length* = 8. Pada kalimat “Pria makan

nasi”, panjang kalimat bertambah menjadi 5 setelah disisipkan token `<start>` dan `<end>`. Untuk mencapai `max_seq_length` perlu ditambahkan 3 indeks token `<pad>` sehingga ukurannya menjadi 8. Sedangkan pada kalimat “Pria dan wanita makan daging di resto mewah”, panjang kalimat sudah melebihi ukuran maksimum yang diperbolehkan. Oleh karena itu, dilakukan pemangkasan kalimat, namun membiarkan token `<end>` tetap berada di akhir kalimat.

Setelah melalui tahap *pad sequence*, kalimat dikonversi menjadi rangkaian *integer*, di mana tiap *integer* merepresentasikan indeks kata yang sesuai. Pada kalimat yang lebih pendek daripada panjang maksimum yang ditetapkan, atau *max length*, token *padding* ditambahkan untuk memastikan setiap kalimat memiliki panjang yang sama. Di sisi lain, selama prapemrosesan berlangsung, dilakukan pengumpulan kata-kata yang ditemukan dan disimpan menjadi sebuah kosakata yang mencakup pasangan kata dengan nomor indeks yang mewakili. Gambar 4.5 memperlihatkan 10 kata dalam kosakata yang memiliki frekuensi kemunculan paling tinggi dengan pengecualian tanda baca.



**Gambar 4.5** Kosakata dengan Frekuensi Kemunculan Paling Tinggi

### 4.3 Implementasi Pembangunan Model

#### 4.3.1 Inisialisasi Model

Arsitektur yang diinisialisasi saat membangun model adalah EfficientNetV2 dan Transformer. Variasi dari model EfficientNetV2 yang dipakai adalah variasi dengan skala kecil (EfficientNetV2-S) dan skala medium (EfficientNetV2-M) telah dilatih sebelumnya terhadap dataset ImageNet. Saat inisialisasi, kedua model memiliki tiga lapisan utama, yakni lapisan *features*, *avgpool*, dan *classifier*. Pada implementasi ini, lapisan yang dibutuhkan hanya lapisan *features*. Dalam arti lain, penggunaan model CNN ini hanya untuk mendapatkan fitur visual dari gambar masukan.

Selanjutnya, tahapan inisialisasi lapisan jaringan Transformer adalah sebagai berikut:

1. Lapisan token *embedding*, berfungsi untuk konversi data token atau kata menjadi vektor sehingga dapat diproses sistem
2. Lapisan *positional embedding*, berfungsi untuk memberikan informasi tentang urutan atau posisi token dalam *sequence*
3. Blok Transformer *decoder* yang mengandung lapisan *multi-head attention*, normalisasi, dan *feed forward*. Kombinasi dari ketiga lapisan ini berfungsi untuk mempelajari fitur gambar dan menghasilkan prediksi kalimat.
4. Lapisan *linear*, yaitu lapisan yang menerima masukan nilai *tensor* dari blok Transformer *decoder*. Lapisan ini mengubah dimensi dari setiap *tensor* menjadi skor logit yang dimensinya sesuai dengan ukuran kosakata (*vocabulary*). Misalnya, jika ukuran kosakata adalah 10.000, maka lapisan ini akan mengubah setiap *tensor* keluaran menjadi *tensor* dengan 10.000 elemen, di mana setiap elemen mewakili skor logit untuk kata yang bersesuaian dalam *vocabulary*.
5. Lapisan *softmax*, berperan sebagai fungsi aktivasi yang diaplikasikan terhadap skor logit untuk mengubahnya menjadi distribusi probabilitas. Dengan distribusi tersebut, lapisan *softmax* dapat menentukan pemilihan kata yang tepat sebagai keluaran dari model.

Terdapat dua model dari kombinasi yang dikembangkan oleh peneliti. Untuk memudahkan penyebutannya, istilah untuk kedua model adalah EffNetV2S-Trans dan EffNetV2M-Trans. Rincian dari kombinasi model tersebut dijabarkan pada Tabel 4.2.

**Tabel 4.2** Rincian Model yang Dibangun

Istilah	Encoder	Decoder
<b>EffNetV2S-Trans</b>	EfficientNetV2-S (skala kecil)	Transformer
<b>EffNetV2M-Trans</b>	EfficientNetV2-M (skala medium)	

Kedua model tersebut menggunakan konfigurasi *hyperparameter* yang sama untuk penentuan arsitektur Transformer sebagaimana dijelaskan pada subbab 4.4.2.1 mengenai konfigurasi *hyperparameter*. Perbedaan terletak pada ukuran transformasi gambar yang diberlakukan terhadap data masukan. Pada EffNetV2S-Trans, ukuran gambar ditransformasi menjadi 288x288 piksel, sedangkan EffNetV2M-Trans diubah menjadi 320x320 piksel. Ukuran ini telah disesuaikan dengan ukuran gambar yang dibutuhkan oleh kedua arsitektur CNN agar proses ekstraksi fitur gambar menjadi lebih optimal.

#### 4.3.2 Implementasi Pelatihan Model

Tahap pelatihan model dilaksanakan setelah arsitektur model telah selesai diinisialisasi. Sebelum proses pelatihan dimulai, peneliti melakukan penetapan konfigurasi *hyperparameter*, *optimizer*, dan *loss function* yang dibutuhkan.

##### 4.3.2.1 Konfigurasi *Hyperparameter*

*Hyperparameter* adalah kumpulan parameter yang digunakan untuk mengatur proses pelatihan model. Pada penelitian ini, *hyperparameter* ditetapkan sebelum dimulainya proses pelatihan. Sebagian besar konfigurasi *Hyperparameter* di dalam arsitektur Transformer ditetapkan dengan mengikuti konfigurasi yang ditetapkan oleh Vasnawi et al. (2017) pada makalah “Attention is All You Need”. Perbedaannya yaitu pada *num\_layers*, peneliti hanya menggunakan satu lapisan. Konfigurasi *Hyperparameter* dapat dilihat pada Tabel 4.2.

**Tabel 4.3** Konfigurasi *Hyperparameter*

<b>Hyperparameter</b>	<b>Nilai</b>	<b>Fungsi</b>
<i>d_model</i>	512	Ukuran input <i>embedding</i> pada Transformer
<i>fc_dim</i>	2048	Ukuran lapisan <i>feed forward</i>
<i>num_layers</i>	1	Jumlah lapisan <i>decoder</i> yang dipakai
<i>n_head</i>	4	Jumlah <i>multi-head attention</i> pada blok Transformer
<i>vocab_size</i>	6070	Ukuran kosakata (bergantung pada dataset)
<i>max_seq_length</i>	25	Tetapan panjang maksimal kalimat
<i>num_epochs</i>	25	Jumlah iterasi pelatihan model (dapat berbeda jika terjadi <i>early stopping</i> )
<i>dropout</i>	0.1	Nilai <i>dropout rate</i> pada setiap lapisan di dalam blok <i>decoder</i>
<i>batch_size</i>	32	Jumlah sampel data yang dipakai pada setiap iterasi pelatihan
<i>num_workers</i>	4	Menentukan bagaimana utilisasi CPU saat <i>data loading</i> .

#### 4.3.2.2 Optimizer

*Optimizer* bekerja dengan memperbarui parameter model seperti bobot dan bias untuk meminimalkan nilai *loss* pada masa pelatihan. Hal ini berdampak pada peningkatan performa model dalam mempelajari fitur dan menghasilkan prediksi. Dalam konteks pengembangan model ini, *Optimizer* yang digunakan adalah Adam (*Adaptive Moment Estimation*). Adam memiliki beberapa parameter yang umum dimodifikasi seperti *learning rate* (*lr*), *beta\_1* ( $\beta^1$ ), *beta\_2* ( $\beta^2$ ) dan *epsilon*  $\epsilon$ . Nilai yang ditetapkan untuk parameter tersebut adalah  $lr = 1e-3$ ,  $\beta^1 = 0.9$ ,  $\beta^2 = 0.999$  dan  $\epsilon = 1e-8$ .

#### 4.3.2.3 Loss Function

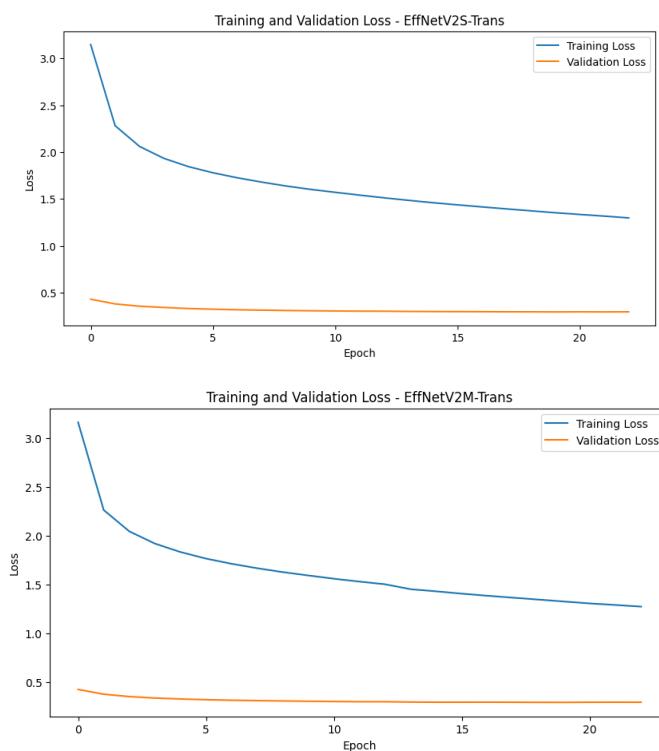
*Loss function* berperan untuk mengukur seberapa baik model dalam membuat prediksi terhadap data target. Nilai *loss* menunjukkan “kerugian” yang terjadi dengan membandingkan antara nilai prediksi dengan nilai target yang hendak

dicapai. Adapun fungsi *loss* yang dipakai pada penelitian ini adalah fungsi *Cross-Entropy*.

Fungsi *Cross-Entropy* mengukur perbedaan antara distribusi probabilitas yang diprediksi oleh model dan distribusi probabilitas dari target. Setiap *output* dari *decoder* dihitung sebagai distribusi probabilitas atas semua kata yang mungkin, dan *cross-entropy loss* digunakan untuk mengukur seberapa efektif model memprediksi urutan kata yang benar. Nilai *loss* ini kemudian digunakan oleh *optimizer* memperbarui bobot dalam model selama masa pelatihan.

#### 4.3.3 Implementasi Evaluasi Model

Evaluasi model diimplementasikan secara bersamaan dengan proses pelatihan model. Proses evaluasi dilakukan dengan melakukan prediksi terhadap data validasi. Data validasi merupakan hasil dari pemisahan data yang menghasilkan 1.000 data gambar dan 5.000 data kalimat. Proses evaluasi dilakukan dengan menghitung dan membandingkan jumlah *training loss* dan *validation loss*. Gambar 4.6 menampilkan perbandingan antara *training loss* dan *validation loss* pada kedua model yang dibangun setelah proses pelatihan dengan 23 *epoch*.



Gambar 4.6 Perbandingan *Training* dan *Validation Loss*

Pada grafik, nilai *validation loss* terlihat lebih tinggi dibandingkan nilai *training loss* sejak *epoch* pertama hingga terakhir. Hal ini menunjukkan bahwa tidak terjadinya *overfitting* selama proses pelatihan. Lebih jelasnya, perbandingan antara *training loss* dan *validation loss* serta durasi training yang dibutuhkan untuk setiap *epoch* ditampilkan oleh Tabel 4.4.

**Tabel 4.4** Perbandingan *Train Loss*, *Validation Loss*, dan Durasi Pelatihan

<b>Epoch</b>	<b>Model EffNetV2S-Trans</b>			<b>Model EffNetV2M-Trans</b>		
	<b>Train Loss</b>	<b>Val Loss</b>	<b>Durasi (detik)</b>	<b>Train Loss</b>	<b>Val Loss</b>	<b>Durasi (detik)</b>
2	2,2653	0,3779	303,21	2,2810	0,3796	500,43
4	1,9214	0,3386	302,40	1,9321	0,3417	510,16
6	1,7676	0,3217	319,95	1,7790	0,3229	504,33
8	1,6689	0,3121	361,61	1,6791	0,3131	485,75
10	1,5945	0,3056	354,91	1,6015	0,3065	483,73
12	1,5316	0,3010	311,93	1,5404	0,3024	487,89
14	1,4538	0,2977	330,02	1,4848	0,2995	509,87
16	1,4081	0,2960	328,70	1,4366	0,2973	504,86
18	1,3669	0,2955	331,56	1,3935	0,2956	507,33
20	1,3277	<b>0,2941</b>	333,38	1,3524	<b>0,2940</b>	508,23
22	<b>1,2924</b>	0,2959	310,16	<b>1,3166</b>	0,2942	504,66

Berdasarkan data tersebut terlihat bahwa kedua model memiliki tingkat *loss* yang tidak terlalu berbeda. Pada tabel, nilai yang bercetak tebal menunjukkan nilai *loss* yang paling baik. Ditemukan bahwa pada kedua model, nilai *validation loss* setelah *epoch* 20 tidak lagi menunjukkan peningkatan, sedangkan nilai *training loss* tetap mengalami peningkatan sampai *epoch* terakhir. Di sisi lain, rata-rata durasi pelatihan dalam 1 *epoch* pada EfficientNetV2S-Trans dan EfficientNetV2M-Trans adalah 326,16 detik dan 500,65 detik secara berurut. Dapat disimpulkan bahwa model EfficientNetV2-M dengan jumlah *parameter* lebih besar memiliki pengaruh yang cukup signifikan terhadap durasi pelatihan.

#### 4.3.4 Implementasi Pengujian Model

Pengujian model dilakukan dengan melakukan prediksi *caption* terhadap gambar masukan pada data yang belum pernah dikenali oleh model. Proses pembentukan *caption* memanfaatkan algoritma *beam search* untuk menentukan kata prediksi yang diambil. Pengujian model berlangsung dengan melakukan iterasi terhadap gambar data uji, lalu dilakukan perhitungan rata-rata skor BLEU berdasarkan hasil prediksi dari setiap gambar. Tabel 4.5 Menampilkan pasangan gambar dan kalimat yang berhasil diprediksi oleh EffNetV2S-Trans dan EffNetV2M-Trans.

**Tabel 4.5** Hasil Prediksi EffNetV2S-Trans dan EffNetV2M-Trans

Gambar	EffNetV2S-Trans	EffNetV2M-Trans
	Pengendara sepeda motor melakukan trik di atas sepeda motor .	Pria mengendarai sepeda motor.
	Anak laki-laki berlari di lapangan.	Anak laki-laki yang mengenakan kemeja biru sedang bermain dengan bola.
	Pemain bola basket melompat untuk menangkap bola.	Pemain bola basket bersiap-siap untuk menangkap bola.

	Anjing cokelat dan putih berjalan di atas rumput.	Anjing hitam dan putih melompati rintangan.
	Matahari terbenam.	Tiga orang berdiri di pantai.

Sebagai perbandingan, berikut adalah lima kalimat referensi dari dataset untuk gambar yang diuji disajikan secara berurut pada Tabel 4.6.

**Tabel 4.6** Perbandingan Kalimat Prediksi dan Kalimat Referensi

Gambar	Kalimat Referensi
	<ol style="list-style-type: none"> <li>1. Dua pengendara sepeda motor berboncengan di atas sepeda di jalan.</li> <li>2. Dua orang mengendarai sepeda motor hitam yang melaju dengan sangat cepat</li> <li>3. Dua orang di atas sepeda motor.</li> <li>4. Dua orang berboncengan di atas sepeda motor.</li> <li>5. Dua orang mengendarai sepeda motor.</li> </ol>
	<ol style="list-style-type: none"> <li>1. Anak laki-laki menendang tumpukan daun.</li> <li>2. Anak laki-laki dengan kemeja biru menendang daun-daun mati dari tumpukan daun.</li> <li>3. Anak laki-laki dengan atasan biru melompat ke tumpukan daun yang disapu di depan pepohonan.</li> </ol>

	<p>4. Anak laki-laki berbaju biru berlari melewati tumpukan daun berwarna cokelat.</p> <p>5. Anak laki-laki berdiri dengan kaki terangkat di tumpukan besar daun.</p>
	<p>1. Pemain bola basket bersiap untuk menembak bola.</p> <p>2. Pemain bola basket sedang melakukan tembakan.</p> <p>3. Pria kulit hitam bermain basket dengan sepatu oranye.</p> <p>4. Pemain bola basket dengan nomor punggung 32 melompat untuk melakukan tembakan, kerumunan orang di latar belakang.</p> <p>5. Pemain bola basket Miami sedang menembak.</p>
	<p>1. Anjing hitam dan cokelat melompati dua tiang berwarna putih dan ungu.</p> <p>2. Anjing melompati rintangan di rumput.</p> <p>3. Anjing melompati rintangan.</p> <p>4. Anjing hitam dan cokelat melompati rintangan dengan penyanga putih.</p> <p>5. Pertunjukan anjing selama pertunjukan anjing di luar ruangan.</p>
	<p>1. Sekelompok orang di pantai saat matahari terbenam.</p> <p>2. Dua orang berjalan menuju sekelompok orang di pantai saat matahari terbenam.</p> <p>3. Dua orang berjalan di sepanjang pantai saat matahari terbenam menuju sekelompok orang.</p>

	<p>4. Dua orang berjalan di sepanjang pantai saat matahari terbenam.</p> <p>5. Dua orang berjalan di pantai saat matahari terbenam.</p>
--	---

Pada hasil yang didapatkan, ditemukan bahwa model sudah dapat memprediksi objek dengan benar dan sesuai dengan ada di kalimat referensi. Namun, untuk keterangan lain seperti keterangan kegiatan atau aktivitas, prediksi model masih kurang tepat pada beberapa gambar pengujian.

#### 4.3.4.1 Hasil Skor BLEU

Metrik BLEU adalah metrik evaluasi yang umum ditemukan pada pengujian sistem di bidang NLP, termasuk pada tugas *image captioning*. Perhitungan skor ini dilakukan dengan membandingkan kalimat hasil prediksi dengan lima kalimat referensi secara kumulatif. Semakin dekat kalimat prediksi dengan kalimat referensi, maka skor akan semakin membaik. Pembobotan dari Skor BLEU yang dihitung pada penelitian ini adalah BLEU-1, BLEU-2, BLEU-3, dan BLEU-4. Tabel 4.7 menunjukkan rata-rata skor BLEU pada pengujian model saat proses prediksi data uji.

**Tabel 4.7** Hasil Evaluasi Skor BLEU Kedua Model

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
EffNetV2S-Trans	<b>0,6028</b>	<b>0,3547</b>	<b>0,2247</b>	<b>0,1572</b>
EffNetV2M-Trans	0,5756	0,3393	0,2188	0,1431

Pada skor yang tampak pada tabel, ditemukan bahwa model EffNetV2S-Trans memiliki hasil skor BLEU yang lebih baik dibandingkan EffNetV2M-Trans, namun dengan selisih yang tidak terlalu berbeda. Sebagai perbandingan, Tabel 4.8 menunjukkan perbandingan antara performa rerata skor BLEU yang dihasilkan pada penelitian ini dengan penelitian terdahulu dengan arsitektur berbeda pada aplikasi Image Captioning bahasa Indonesia.

**Tabel 4.8** Perbandingan Evaluasi Skor BLEU dengan Penelitian Terdahulu

Arsitektur	Dataset	Rerata Skor BLEU			
		1	2	3	4
InceptionV3 dan GRU (Nugraha et al., 2019)	Flickr30k Bahasa	0,363	0,170	0,060	0,014
CNN dan LSTM (Mulyanto, E. et al., 2019)	FEEH-ID	0,500	0,314	0,239	0,131
ResNet50 dan Transformer (Mulyawan et al., 2022)	Flickr8k Bahasa	0,560	0,4117	0,2942	0,2057
EfficientNetB0 V1 dan Transformer (Mulyawan et al., 2023)	Flickr8k Bahasa	0,5773	0,4254	0,3024	0,2124
EffNetV2S-Transformer (penelitian ini)	Flickr8k Bahasa	0,6028	0,3547	0,2247	0,1572

Pada Tabel 4.8 ditunjukkan bahwa model yang dihasilkan pada penelitian ini mampu mendapatkan skor BLEU-1 paling tinggi dibandingkan penelitian terdahulu dengan dataset yang sama. Namun, pada BLEU-2 hingga BLEU-4, skor yang didapatkan belum lebih baik dibandingkan pada model yang menggunakan CNN EfficientNetV1 dan ResNet50 pada lapisan *encoder*.

#### 4.4 Penerapan Model

##### 4.4.1 Implementasi Pembuatan API

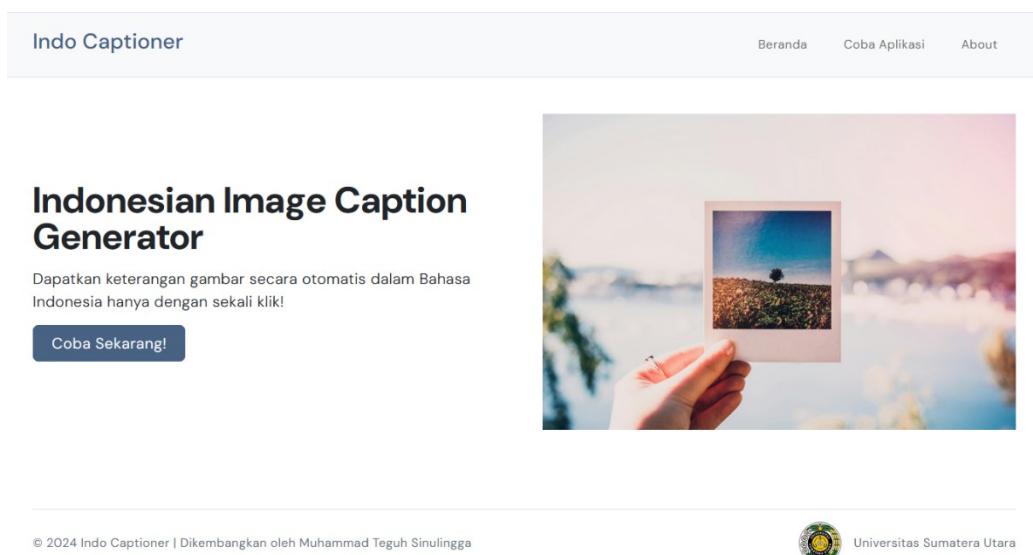
Model yang telah dikembangkan sebelumnya diintegrasikan ke dalam sebuah API atau *Application Programming Interface* pada *web server* yang berjalan di perangkat lokal. API ini dirancang untuk menjalankan halaman antarmuka *website* dan fungsi inferensi model yang berjalan secara interaktif. Tahapan-tahapan interaksi yang terjadi pada API adalah sebagai berikut:

1. Pada halaman inferensi, pengguna dapat mengunggah gambar melalui *form* yang disediakan.
2. Setelah gambar berhasil diunggah, API akan berinteraksi dengan model dan mengirimkan gambar tersebut sebagai masukan bagi model.

3. Selanjutnya, model melakukan inferensi untuk memprediksi *caption* berdasarkan gambar yang telah diterima.
4. Setelah itu, model mengirimkan kembali sebuah *request* ke API dengan format JSON, berisikan data *caption* yang telah dibentuk.
5. Data *caption* tersebut kemudian ditampilkan pada halaman *website* sehingga pengguna dapat melihat hasilnya.

#### 4.4.2 Hasil Pembuatan Antarmuka *Website*

Setelah API berhasil dikembangkan, langkah berikutnya adalah pembuatan antarmuka pada *website*. Antarmuka ini dibangun mengikuti rancangan yang telah diperlihatkan pada subbab 3.7. Halaman *website* dibuat dengan menggunakan bahasa HTML dan *framework Bootstrap* yang membantu proses desain halaman. Jumlah antarmuka yang dibuat adalah sejumlah tiga halaman, yaitu halaman beranda, halaman *caption generator*, dan halaman *about*. Gambar 4.7 sampai 4.11 menunjukkan hasil pembuatan antarmuka pada ketiga halaman tersebut.



The screenshot shows the homepage of the Indo Captioner website. At the top, there is a navigation bar with links for 'Beranda', 'Coba Aplikasi', and 'About'. The main content area features a large image of a hand holding a small photograph of a landscape. To the left of the image, the text 'Indonesian Image Caption Generator' is displayed, along with a subtext 'Dapatkan keterangan gambar secara otomatis dalam Bahasa Indonesia hanya dengan sekali klik!' and a blue button labeled 'Coba Sekarang!'. At the bottom of the page, there is a footer with the text '© 2024 Indo Captioner | Dikembangkan oleh Muhammad Teguh Sirulingga' and the logo of Universitas Sumatera Utara.

**Gambar 4.7** Implementasi Halaman Beranda

Indo Captioner

Beranda Coba Aplikasi About

## Indonesian Image Caption Generator

Unggah gambar untuk mulai!



[Browse...](#) No file selected.

[Unggah](#)

Hasil Prediksi:

---

© 2024 Indo Captioner | Dikembangkan oleh Muhammad Teguh Sinulingga

 Universitas Sumatera Utara

**Gambar 4.8** Implementasi Halaman Prediksi

Indo Captioner

Beranda Coba Aplikasi About

## Indonesian Image Caption Generator

Unggah gambar untuk mulai!



[Browse...](#) 161669933\_3e7d8c7e2c.jpg

[Buat caption](#)

Hasil Prediksi:

---

© 2024 Indo Captioner | Dikembangkan oleh Muhammad Teguh Sinulingga

 Universitas Sumatera Utara

**Gambar 4.9** Implementasi Halaman Prediksi Setelah Gambar Dipilih



© 2024 Indo Captioner | Dikembangkan oleh Muhammad Teguh Sinulingga  Universitas Sumatera Utara

**Gambar 4.10** Halaman Prediksi Setelah Melakukan Pembuatan *Caption*

The screenshot shows the 'About' page of the 'Indo Captioner' website. The header includes the project name 'Indo Captioner // Indo Captioner', developer information 'Developed by Muhammad Teguh Sinulingga', and a note 'Dikembangkan untuk memenuhi tugas akhir dan sebagai syarat memperoleh ijazah Sarjana Ilmu Komputer'. Below this is a 'Tech stack' table:

Tech stack		
Role	Purpose	Tech
Deep Learning Model	Encoder (CNN)	EfficientNetV2-S, EfficientNetV2-M
	Decoder	Transformer (decoder only)
Frontend	Markup Language	HTML
	Styling	CSS and Bootstrap
	Scripting	JavaScript
Web Deployment	Backend Language	Python
	API Framework	Flask

At the bottom are two social media links: 'in LinkedIn' and 'GitHub'.

**Gambar 4.11** Implementasi Halaman *About*

#### 4.4.3 Pengujian Aplikasi *Website*

Pengujian pada aplikasi *website* dilaksanakan dengan melakukan pengunggahan gambar secara interaktif untuk mendapatkan hasil prediksi *caption*. Dalam pengujian ini, peneliti telah mengumpulkan sejumlah 6 gambar dari internet dengan beragam objek dan latar belakang pada gambar tersebut. Peneliti juga melakukan perbandingan antara performa yang dihasilkan oleh kedua model dari sisi kalimat serta waktu yang dibutuhkan dalam satu kali prediksi. Gambar 4.12

merupakan kumpulan foto beserta ID gambar berupa nama *file* yang dijadikan objek uji aplikasi *website*.



**Gambar 4.12** Kumpulan Gambar sebagai Objek Pengujian pada *Website*

Adapun hasil prediksi dari gambar-gambar tersebut ditunjukkan oleh Tabel 4.9 dan Tabel 4.10.

**Tabel 4.9** Pengujian pada *Website* untuk EffNetV2S-Trans

Model EffNetV2S-Trans			
ID Gambar	Hasil Prediksi	Kesesuaian	Durasi (detik)
dog.jpg	Anjing berlari melalui air.	Sesuai	10,16
children.jpg	Sekelompok orang berdiri di atas air.	Sesuai	14,44
motocross.jpg	Pria melakukan trik di udara.	Sesuai	12,11
ski.jpg	Pemain ski menuruni bukit bersalju.	Sesuai	11,51
cricket.jpg	Pemain bisbol sedang melempar bola.	Kurang sesuai	10,91
lake.jpg	Sebuah danau.	Kurang sesuai	10,87
<b>Rata-Rata Durasi</b>			11.66

**Tabel 4.10** Pengujian pada *Website* untuk EffNetV2M-Trans

Model EffNetV2M-Trans			
ID Gambar	Hasil Prediksi	Kesesuaian	Durasi (detik)
dog.jpg	Anjing hitam dan putih berlari melalui air.	Sesuai	13,44
children.jpg	Anak laki-laki melompat ke dalam air.	Sesuai	19,94
motocross.jpg	Pengendara sepeda motor trail di udara.	Sesuai	13,54
ski.jpg	Pemain ski menuruni lereng.	Sesuai	12,57
cricket.jpg	Anak laki-laki bermain bisbol.	Kurang sesuai	10,49
lake.jpg	Pria di dalam air.	Tidak sesuai	10,18
<b>Rata-Rata Durasi</b>			13,36

Pada Tabel 4.9 dan 4.10 ditampilkan hasil prediksi, penilaian kesesuaian, dan durasi yang dibutuhkan oleh sistem untuk membangkitkan kalimat. Penilaian kesesuaian yang diberikan terbagi menjadi tiga kategori, dengan penjelasan sebagai berikut:

1. Sesuai: kalimat mampu mendeskripsikan objek dan keterangan dengan baik.
2. Kurang sesuai: kalimat mampu mendeskripsikan objek dengan baik, namun keterangan masih belum baik.
3. Tidak sesuai: kalimat tidak mendeskripsikan objek dengan baik.

Pada sebagian besar pengujian, kalimat yang dihasilkan telah mampu memprediksi objek dan keterangan aktivitas yang terjadi di dalam gambar dengan baik. Namun, pada sebagian lain, model belum mampu memprediksi objek atau keterangan secara tepat. Hal ini disebabkan oleh model yang belum pernah menemukan dan mempelajari ciri-ciri dari objek tersebut selama masa pelatihan.

Selain itu, berdasarkan hasil pengujian pada *website*, dapat disimpulkan bahwa model dapat berjalan dan terintegrasi dengan baik. Rata-rata durasi yang

dibutuhkan oleh sistem untuk melakukan prediksi pada lingkungan implementasi yang digunakan adalah 11,66 detik pada EffNetV2S-Trans dan 13,36 detik pada EffNetV2M-Trans. Rata-rata durasi tersebut adalah durasi yang diperlukan oleh sistem dari mulai pengunggahan gambar, proses inferensi, hingga memunculkan kalimat ke tampilan website. Hal ini menunjukkan bahwa model CNN yang memiliki jumlah parameter lebih besar, yaitu EfficientNetV2-M membutuhkan waktu lebih lama dalam melakukan prediksi.

## BAB V

### PENUTUP

#### 5.1 KESIMPULAN

Kesimpulan berdasarkan analisis, perancangan, implementasi, dan pengujian yang dilakukan pada penelitian adalah sebagai berikut:

1. Penelitian ini berhasil mengembangkan model *image captioning* untuk Bahasa Indonesia menggunakan dua varian arsitektur: EfficientNetV2S-Transformer dan EfficientNetV2M-Transformer dengan dataset Flickr8k yang diterjemahkan ke Bahasa Indonesia.
2. Berdasarkan hasil pelatihan kedua model, varian EfficientNetV2M-Transformer mencatatkan *validation loss* paling baik, yaitu 0,2940.
3. Pengujian inferensi menunjukkan bahwa kedua model menghasilkan keterangan yang berbeda untuk gambar yang sama.
4. Model EfficientNetV2S-Transformer mendapatkan skor BLEU-1 hingga BLEU-4 yang lebih baik dibandingkan EfficientNetV2M-Transformer, yaitu sebesar 0,6028, 0,3547, 0,2247, dan 0,1572.
5. Penggunaan CNN EfficientNetV2-S, yang memiliki jumlah parameter lebih rendah dibandingkan EfficientNetV2-M, menunjukkan durasi pelatihan dan inferensi yang tercepat, dengan selisih waktu rata-rata masing-masing adalah 174,49 detik dan 1,7 detik.
6. Pengujian sistem menunjukkan bahwa model efektif dalam melakukan prediksi untuk objek yang telah dikenal dari dataset Flickr8k. Namun, untuk objek yang belum dikenali, sistem sering menghasilkan deskripsi yang tidak akurat.
7. Aplikasi *website* yang dikembangkan telah mampu menghasilkan prediksi berupa kalimat deskriptif ketika menerima masukan data gambar.

## 5.2 SARAN

Saran yang dapat penulis berikan untuk penelitian selanjutnya adalah sebagai berikut:

1. Penelitian ini menggabungkan EfficientNetV2 dengan Transformer. Penelitian berikutnya dapat mengaplikasikan arsitektur lain seperti CPTR (Caption Transformer) dalam mengembangkan model untuk menghasilkan keterangan gambar dalam Bahasa Indonesia.
2. Dikarenakan keterbatasan kemampuan komputasi, penulis melatih model menggunakan dataset Flickr8k yang hanya terdiri dari 8.000 gambar. Untuk memperluas keragaman *vocabulary* dan fitur visual yang dapat dipelajari, penelitian berikutnya dapat memanfaatkan dataset dengan jumlah data lebih besar seperti Flickr30k atau MS COCO (Microsoft Common Objects in Context)
3. Berdasarkan hasil penelitian, sistem hanya dapat menghasilkan kalimat yang cenderung pendek dan hanya berjumlah satu kalimat. Penelitian selanjutnya dapat mengembangkan sistem yang dapat memprediksi kalimat yang lebih panjang atau menghasilkan lebih dari satu kalimat prediksi.
4. Penelitian di masa mendatang dapat mengembangkan model *image captioning* yang hanya berfokus pada objek atau latar belakang tertentu. Studi ini dapat dimulai dengan mengumpulkan dataset yang terbatas hanya pada objek tertentu, seperti objek hewan atau kumpulan gambar aktivitas olahraga.

**BAB VI**  
**DAFTAR PUSTAKA**

- Al-Faruq, U. a. A., & Fudholi, D. H. (2021). Implementasi Arsitektur Transformer pada Image Captioning dengan Bahasa Indonesia. *AUTOMATA*, 2(2).
- Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., & Darrell, T. (2017). Long-Term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 677–691. <https://doi.org/10.1109/tpami.2016.2599174>.
- Fudholi, D. H., Zahra, A., & Nayoan, R. A. N. (2022). A Study on Visual Understanding Image Captioning using Different Word Embeddings and CNN-Based Feature Extractions. *Kinetik : Game Technology, Information System, Computer Network, Computing, Electronics, and Control*. <https://doi.org/10.22219/kinetik.v7i1.1394>.
- Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., ... & Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern recognition*, 77, 354-377.
- He, S., Liao, W., Tavakoli, H. R., Yang, M., Rosenhahn, B., & Pugeault, N. (2020). Image captioning through image transformer. In *Proceedings of the Asian conference on computer vision*.
- 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(8), 5455–5516. <https://doi.org/10.1007/s10462-020-09825-6>
- Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s), 1-41.

- Kiros, R., Salakhutdinov, R., & Zemel, R. S. (2014). Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*.
- Liu, S., Bai, L., Hu, Y., & Wang, H. (2018). Image captioning based on deep neural networks. In *MATEC web of conferences* (Vol. 232, p. 01052). EDP Sciences.
- M. Hodosh, P. Young and J. Hockenmaier (2013) "Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics", Journal of Artificial Intelligence Research, Volume 47, pages 853-899.
- Mulyanto, E., Setiawan, E. I., Yuniarso, E. M., & Purnomo, M. H. (2019). Automatic Indonesian Image Caption Generation using CNN-LSTM Model and FEEH-ID Dataset. In *2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications, CIVEMSA 2019 - Proceedings Article 9071632* (2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications, CIVEMSA 2019 - Proceedings). Institute of Electrical and Electronics Engineers Inc.. <https://doi.org/10.1109/CIVEMSA45640.2019.9071632>.
- Mulyawan, R., Sunyoto, A., & Muhammad, A. H. M. (2023). Pre-Trained CNN Architecture Analysis for Transformer-Based Indonesian Image Caption Generation Model. *JOIV: International Journal on Informatics Visualization*, 7(2), 487-493.
- Nugraha, A. A., & Arifianto, A. (2019, July). Generating image description on Indonesian language using convolutional neural network and gated recurrent unit. In *2019 7th International Conference on Information and Communication Technology (ICoICT)* (pp. 1-6). IEEE.
- Nursikuwagus, A., Munir, R., & Khodra, M. L. (2020). Image Captioning menurut Scientific Revolution Kuhn dan Popper. *Jurnal Manajemen Informatika* (JAMIKA), 10(2), 110-121.

- O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.
- Papineni, K., Roukos, S., Ward, T., dan Zhu, W.-J., 2001, BLEU, *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Philadelphia.
- Post, M. (2018). A Call for Clarity in Reporting BLEU Scores. In *WMT 2018 - 3rd Conference on Machine Translation, Proceedings of the Conference* (Vol. 1, pp. 186–191). Association for Computational Linguistics (ACL). <https://doi.org/10.18653/v1/w18-6319>.
- Pratiwi, V. R., & PARDEE, J. (2022). Image Captioning Menggunakan Metode Inception-V3 dan Transformer. FTI.
- Shiri, F. M., Perumal, T., Mustapha, N., & Mohamed, R. (2023). A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU. *arXiv preprint arXiv:2305.17473*.
- Tan, M. & Le, Q. (2021). EfficientNetV2: Smaller Models and Faster Training. Proceedings of the 38th International Conference on Machine Learning, in *Proceedings of Machine Learning Research* 139:10096-10106 Available from <https://proceedings.mlr.press/v139/tan21a.html>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Zohourianshahzadi, Z., & Kalita, J. K. (2022). Neural attention for image captioning: review of outstanding methods. *Artificial Intelligence Review*, 55(5), 3833–3862. <https://doi.org/10.1007/s10462-021-10092-2>