



**MODEL *CROSS-ATTENTION VIDEO VISION TRANSFORMER*
UNTUK DESKRIPSI VIDEO**

TUGAS AKHIR

**Miftahul Huda
121450125**

**PROGRAM STUDI SAINS DATA
JURUSAN SAINS
INSTITUT TEKNOLOGI SUMATERA
LAMPUNG SELATAN
2024**



**MODEL *CROSS-ATTENTION VIDEO VISION TRANSFORMER*
UNTUK DESKRIPSI VIDEO**

TUGAS AKHIR

Diajukan sebagai syarat untuk memperoleh gelar sarjana

**Miftahul Huda
121450125**

**PROGRAM STUDI SAINS DATA
JURUSAN SAINS
INSTITUT TEKNOLOGI SUMATERA
LAMPUNG SELATAN
2024**

LEMBAR PENGESAHAN

Tugas Akhir dengan judul MODEL *CROSS-ATTENTION VIDEO VISION TRANSFORMER UNTUK DESKRIPSI VIDEO* adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya, baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Institut Teknologi Sumatera maupun di institusi pendidikan lainnya.

Lampung Selatan, 28 November 2024
Penulis,

Miftahul Huda
NIM 121450125



Pembimbing I

Ahmad Luky Ramdani, S.Kom., M.Kom
NIP. 198804242020121004

Pembimbing II

Luluk Muthoharoh, M.Si
NIP 199504112022032014

Disahkan oleh,
Koordinator Program Studi Sains Data
Jurusan Sains
Institut Teknologi Sumatera

Tirta Setiawan, S.Pd., M.Si
NIP 199008222022031003

MODEL CROSS ATTENTION VIDEO VISION TRANSFORMER UNTUK DESKRIPSI VIDEO

Miftahul Huda (121450125)

Ahmad Luky Ramdani, S.Kom., M.Kom

Luluk Muthoharoh, M.Si

ABSTRAK

Deskripsi video otomatis merupakan tugas penting dalam pemahaman video, namun masih menghadapi tantangan dalam efisiensi komputasi dan kompleksitas arsitektur. *Video Vision Transformer* (ViViT) menunjukkan performa unggul dalam menangkap informasi spasial-temporal, tetapi pendekatannya memerlukan daya komputasi tinggi karena penerapan *self-attention* pada seluruh token video. Penelitian ini mengusulkan arsitektur *Cross-Attention Video Vision Transformer* yang lebih efisien, dengan menekan kompleksitas pada *self-attention* dan tanpa bergantung pada ekstraksi fitur dari pretrained CNN, sehingga memungkinkan pelatihan secara *end-to-end*. Tugas *video captioning* dipilih sebagai benchmark karena menuntut pemahaman spasial-temporal yang mendalam serta kemampuan generatif. Model yang diusulkan dibandingkan dengan empat varian ViViT lainnya menggunakan metrik evaluasi seperti *Sequence Accuracy*, CIDEr, serta analisis kompleksitas komputasi. Hasil eksperimen menunjukkan bahwa arsitektur ini mampu menghasilkan deskripsi video yang relevan, dengan kecepatan inferensi lebih tinggi dan performa yang sebanding dengan varian ViViT lainnya.

Kata kunci: *Cross-Attention*, Deskripsi Video, Efisiensi Komputasi, Pemodelan Spasial-Temporal, *Video Vision Transformer*

CROSS ATTENTION VIDEO VISION TRANSFORMER MODEL FOR VIDEO DESCRIPTION

Miftahul Huda (121450125)

Ahmad Luky Ramdani, S.Kom., M.Kom

Luluk Muthoharoh, M.Si

ABSTRACT

Automatic video description is an important task in video understanding, but it still faces challenges in computational efficiency and architectural complexity. The Video Vision Transformer (ViViT) has shown superior performance in capturing spatial-temporal information, but its approach requires high computational power due to the application of self-attention on all video tokens. This paper proposes a more efficient Cross-Attention Video Vision Transformer architecture, by suppressing the complexity of self-attention and without relying on feature extraction from pretrained CNNs, thus enabling end-to-end training. The video captioning task is chosen as the benchmark because it demands deep spatial-temporal understanding and generative capabilities. The proposed model is compared with four other ViViT variants using evaluation metrics such as Sequence Accuracy, CIDEr, and computational complexity analysis. Experimental results show that this architecture is capable of generating relevant video descriptions, with higher inference speed and comparable performance to other ViViT variants.

Keywords: Computational Efficiency, Cross-Attention, Spatial-Temporal Modeling, Video Captioning, Video Vision Transformer

DAFTAR ISI

LEMBAR PENGESAHAN	i
HALAMAN PERNYATAAN ORISINALITAS	ii
HALAMAN PERSETUJUAN PUBLIKASI	iii
ABSTRAK	iv
ABSTRACT	v
MOTTO	vi
PERSEMPAHAN	vii
KATA PENGANTAR	viii
DAFTAR ISI	ix
DAFTAR GAMBAR	xiii
DAFTAR TABEL	xiv
DAFTAR ALGORITMA	xv
Bab I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan	3
1.4 Manfaat Penelitian	3

1.5	Batasan Masalah	3
Bab II	KAJIAN PUSTAKA	4
2.1	Penelitian Terkait	4
2.2	Deskripsi Video	5
2.3	Tokenisasi dan <i>Word Embedding</i>	5
2.4	Transformer	5
2.5	<i>Video Vision Transformer</i>	8
2.5.1	Proses Pembentukan <i>Embedding</i>	8
2.5.2	Model 1: <i>Spatio-Temporal Attention</i>	8
2.5.3	Model 2: <i>Factorised Encoder</i>	8
2.5.4	Model 3: <i>Factorised Self-Attention</i>	9
2.5.5	Model 4: <i>Factorised Dot Product Attention</i>	10
2.6	3D <i>Convolutional Neural Network</i>	10
2.7	Metrik CIDEr	11
Bab III	METODOLOGI PENELITIAN	13
3.1	Deskripsi Penelitian	13
3.2	Deskripsi Data	13
3.3	Tahapan Penelitian	14
3.3.1	Persiapan Penelitian	14
3.3.2	Persiapan Data	14
3.3.3	Pemrosesan Teks	15
3.3.4	Pemrosesan Video	15
3.3.5	Rancangan Arsitektur Model	15
3.3.6	Pelatihan Model	20
3.3.7	Evaluasi dan Analisis Hasil	20

Daftar Isi

Bab IV HASIL DAN PEMBAHASAN	21
4.1 Perbandingan Performa Variasi Model <i>Video Vison Transformer</i>	21
4.1.1 Eksplorasi Dataset	21
4.1.2 <i>Network Architecture</i>	22
4.1.3 Variasi Model <i>Video Vison Transformer</i>	23
4.1.4 Evaluasi Model	23
4.1.5 Analisis Kompleksitas	24
4.2 Evaluasi Performa <i>Cross-Attention</i>	26
4.2.1 Studi Ablasi <i>Cross-Attention</i>	26
4.2.2 Generalisasi Model	28
Bab V KESIMPULAN DAN SARAN	32
5.1 Kesimpulan	32
5.2 Saran	32

DAFTAR GAMBAR

2.1	Arsitektur Transformer dengan <i>multi-head attention</i> , <i>Layer Normalization</i> , dan <i>Feed Forward Neural Network</i> disetiap blocknya	6
2.2	Proses <i>self-attention</i> dengan satu <i>batch</i> input.	7
2.3	<i>Multi-head attention</i>	7
2.4	<i>Factorised Encoder</i>	9
2.5	<i>Factorised Self-Attention</i>	9
2.6	<i>Factorised Dot Product Attention</i>	10
2.7	Ilustrasi proses konvolusi yang akan bergerak ke samping, ke bawah, dan ke belakang (3D).	11
2.8	3D <i>Convolutional Neural Network</i>	11
3.1	Flowchart rancangan penelitian.	14
3.2	Beberapa variasi model ViViT dan <i>Cross-Attention</i> yang memproses token spasial dan temporal.	16
3.3	<i>Spatial Embedding</i>	16
3.4	<i>Temporal Embedding</i>	16
3.5	Linear layer untuk mentransformasi $\mathbf{z}_i^{(t)} \in \mathbb{R}^{P_d \times (W/r) \times (H/r) \times C}$ menjadi $\mathbf{z}_i^{(t)} \in \mathbb{R}^d$ dan $\mathbf{z}_i^{(s)} \in \mathbb{R}^{(D/r) \times P_w \times P_h \times C}$ menjadi $\mathbf{z}_i^{(s)} \in \mathbb{R}^d$	17
3.6	Encoder arsitektur <i>Cross-Attention</i>	17
3.7	<i>Decoder</i> Transformer.	18
3.8	<i>Matriks Embedding</i>	18
3.9	Output <i>sequence</i> menghasilkan teks deskripsi.	19
4.1	Distribusi jumlah <i>caption</i> setiap video.	21
4.2	Perbandingan <i>caption</i> hasil generate dengan beberapa <i>Ground Truth</i> . . .	29

Daftar Gambar

4.3	Perbandingan <i>caption</i> hasil generate dengan beberapa <i>Ground Truth</i> . . .	29
4.4	Perbandingan <i>caption</i> hasil generate dengan beberapa <i>Ground Truth</i> . . .	30
4.5	Distribusi panjang <i>sequence</i> pada dataset	31
4.6	Caption hasil generasi dengan panjang mencapai sekitar 14 kata	31

DAFTAR TABEL

2.1	Penelitian terkait	4
3.1	Contoh video dengan captionnya dalam dataset.	13
4.1	Konfigurasi 3D Convolutional Projection	22
4.2	Variasi model	23
4.3	Perbandingan akurasi dan skor CIDEr model <i>Cross-Attention</i> dengan empat variasi model ViViT	24
4.4	Perbandingan kompleksitas waktu	25
4.5	Jumlah parameter, GFLOPs, dan <i>Runtime</i>	26
4.6	Efek dan perbandingan akurasi serta skor CIDEr model <i>Cross-Attention</i> Video Vision Transformer dengan Variasi Ukuran <i>Patch</i> pada <i>Temporal Embedding</i> (\mathbf{Z}_t) yang menghasilkan $n_d = (D/2)$ dan $n_d = D$	26
4.7	Efek dan perbandingan akurasi serta skor CIDEr model <i>Cross-Attention</i> Video Vision Transformer dengan rasio <i>resize r</i> yang berbeda.	27
4.8	Efek dan perbandingan akurasi serta skor CIDEr model <i>Cross-Attention</i> Video Vision Transformer dengan jumlah fitur 3D <i>Convolutional Projection</i> yang berbeda.	28
4.9	Perbandingan performa model berdasarkan urutan <i>query</i> pada <i>Cross-Attention</i>	28

DAFTAR ALGORITMA

1	Pelatihan model	20
---	---------------------------	----

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada Januari 2024, jumlah pengguna internet di dunia mencapai 5,35 miliar orang [1, 2]. Laporan dari *Sandvine Global Internet Phenomena Report* menunjukkan bahwa platform video seperti YouTube, Netflix, dan TikTok terus mendominasi penggunaan *bandwidth* internet global [3], yang mencerminkan peningkatan signifikan dalam jumlah dan durasi video yang diakses oleh pengguna. Hal ini menunjukkan bahwa video telah menjadi salah satu format utama dalam konsumsi informasi di internet. Seiring dengan meningkatnya jumlah video ini, muncul kebutuhan baru dalam pengelolaan, pencarian, pengindeksan, dan pemantauan konten video secara efektif [4, 5]. Peningkatan volume data video ini menuntut inovasi dalam teknologi untuk memudahkan akses dan pengelolaan konten. Selain itu, aksesibilitas bagi individu dengan gangguan penglihatan masih sangat terbatas karena belum banyak tersedia sistem yang secara otomatis dapat memberikan penjelasan naratif tentang apa yang terjadi dalam video, seperti adegan, aksi, dan konteks visual yang tidak bisa didengar [4, 6].

Banyak peneliti di bidang *Computer Vision* dan *Natural Language Processing* yang telah mendeskripsikan gambar dan video ke dalam teks berbahasa Inggris [5–7]. Di Indonesia, beberapa peneliti juga telah berhasil menghasilkan deskripsi gambar dalam bahasa Indonesia [8–10] untuk mengambil informasi visual secara otomatis karena deskripsi visual dalam bahasa lokal sangat penting untuk meningkatkan aksesibilitas teknologi bagi masyarakat luas, termasuk pengguna dengan keterbatasan bahasa atau kemampuan membaca gambar. Namun, mendeskripsikan video merupakan tantangan yang lebih kompleks dibandingkan dengan gambar. Hal ini disebabkan oleh adanya informasi temporal yang terkandung dalam urutan frame video, sehingga tidak hanya diperlukan ekstraksi fitur spasial dan channel, tetapi juga fitur temporal dari dimensi waktu [5, 6]. Selain itu, tantangan lain yang kerap muncul adalah adanya *trade-off* antara akurasi dan kompleksitas, yang menjadi pertimbangan penting dalam pengembangan sistem deskripsi video yang efisien dan akurat.

Penelitian deskripsi video konvensional umumnya menggunakan pendekatan berbasis *Encoder-Decoder Recurrent Neural Network* seperti LSTM dan GRU pada output *Convolutional Neural Network* (CNN) untuk memodelkan hubungan spasial dan temporal dalam video [5, 7, 11, 12] karena arsitektur ini mampu menangkap fitur visual dari setiap frame (melalui CNN) dan mengurutkan informasi tersebut secara temporal (melalui RNN), sehingga menghasilkan deskripsi yang runtut dan kontekstual. Misalnya, Venugopalan et al. (2015) yang menggunakan pendekatan *sequence-to-sequence* dengan LSTM untuk menghasilkan deskripsi video [11], sementara Pan et al. (2016) memperkenalkan model yang menggabungkan Semantic Attributes dengan *Convolutional Neural Network* (CNN) untuk ekstraksi fitur visual dan *Recurrent Neural*

Network (RNN) seperti LSTM untuk pemodelan temporal [12]. Namun, *Recurrent Neural Network* (RNN) sering kali mengalami kendala saat harus memproses urutan data yang sangat panjang [13]. Karena itu muncul arsitektur Transformer yang dibuat dengan alasan untuk mengatasi kekurangan dari RNN oleh Vaswani et al. (2017) [13]. Arsitektur ini juga digunakan untuk data video, seperti yang diusulkan oleh Sun et al. (2019) dengan VideoBERT [14] dan Arnab et al. (2021) dengan *Video Vision Transformer* (ViViT) [15].

Video Vision Transformer (ViViT) dirancang untuk memahami isi video tanpa perlu menggunakan jaringan saraf berulang (*Recurrent Neural Network*) atau konvolusi 3D (*3D Convolutional Neural Network*) yang sebelumnya umum digunakan. Model ini bekerja dengan membagi video menjadi bagian-bagian kecil (*patch*) dari setiap frame, lalu memprosesnya menggunakan arsitektur Transformer untuk mengenali pola visual dan urutan waktu dalam video. Untuk mengatasi tantangan dalam memproses data video yang besar dan kompleks, ViViT mengembangkan empat varian arsitektur, yaitu *Spatio-Temporal Attention*, *Factorised Encoder*, *Factorised Self-Attention*, dan *Factorised Dot Product Attention*. Masing-masing varian ini menggunakan cara yang berbeda dalam memisahkan dan menggabungkan informasi ruang (spasial) dan waktu (temporal), dengan tujuan untuk meningkatkan efisiensi komputasi. Pendekatan ini telah menunjukkan performa yang baik dalam berbagai tugas seperti pengenalan aksi, klasifikasi video, dan pembuatan deskripsi video [15, 16].

Meskipun ViViT menunjukkan kinerja unggul dalam pemrosesan video berkat kemampuannya menangkap informasi spasial dan temporal secara simultan melalui mekanisme *self-attention*, arsitektur ini memiliki kompleksitas komputasi yang sangat tinggi. Hal ini disebabkan oleh kebutuhan untuk menerapkan *self-attention* secara menyeluruh terhadap seluruh token spasial-temporal yang diekstraksi dari setiap frame video, yang jumlahnya dapat meningkat secara eksponensial [13]. Kebutuhan memori dan daya komputasi yang besar menjadi kendala signifikan, terutama pada perangkat dengan sumber daya terbatas. Walaupun ViViT menjanjikan akurasi tinggi serta pemahaman spasial-temporal yang baik, efisiensi waktu inferensi dan skalabilitasnya masih menjadi tantangan utama yang perlu diatasi agar dapat diimplementasikan secara luas dan efektif. Oleh karena itu penelitian ini mengusulkan pendekatan dengan *Cross-Attention* yang lebih efisien.

Penelitian ini mengadopsi konsep ViViT dengan *Cross-Attention* untuk menangkap konteks spasial 3D dan temporal berdasarkan spasial-temporal untuk deskripsi video dan tidak menggunakan ekstraksi fitur dari *pretrained CNN*, sehingga menghasilkan model *end-to-end* yang diharapkan memiliki kecepatan inferensi yang tinggi dan jumlah FLOPs (*Floating-point Operations Per Second*) yang rendah. Penelitian ini berfokus pada mengoptimalkan arsitektur ViViT agar memiliki kecepatan inferensi yang lebih tinggi dan jumlah FLOPs yang rendah. Walaupun dalam makalah asli eksperimen ViViT menggunakan tugas *video recognition*, tugas *video captioning* dipilih dalam penelitian ini karena menuntut pemahaman spasial-temporal yang lebih dalam serta kemampuan generatif, sehingga penggunaan *video captioning* sebagai benchmark memungkinkan evaluasi yang lebih menyeluruh. Dataset dengan pola umum digunakan sebagai benchmark untuk mengevaluasi performa model, sebagaimana yang umum dilakukan

dalam penelitian sejenis lainnya [5, 7, 11, 12]. Oleh karena itu, penelitian ini tidak berfokus pada analisis atau batasan kategori aksi dalam video, melainkan pada pengembangan metode yang dapat meningkatkan kecepatan inferensi dalam menghasilkan deskripsi video dan mengurangi kompleksitas model.

1.2 Rumusan Masalah

Berdasarkan latar belakang, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana performa dan kompleksitas model *Cross-Attention Video Vision Transformer* dibandingkan dengan empat varian pendekatan ViViT lainnya dalam tugas deskripsi video?
2. Bagaimana pengaruh bagian-bagian arsitektur model *Cross-Attention Video Vision Transformer* berdasarkan performa?

1.3 Tujuan

Berdasarkan rumusan masalah, tujuan dari penelitian ini adalah sebagai berikut:

1. Membandingkan performa dan kompleksitas model *Cross-Attention Video Vision Transformer* dengan empat varian pendekatan ViViT lainnya dalam menghasilkan deskripsi video.
2. Mengevaluasi dan menganalisis pengaruh bagian-bagian arsitektur model *Cross-Attention Video Vision Transformer* melalui studi aborsi.

1.4 Manfaat Penelitian

1. Model ini dapat digunakan untuk aplikasi yang membutuhkan interpretasi video otomatis dan *real-time*, seperti pencarian video, sistem pemantauan, dan multimedia dengan dataset yang lebih spesifik.
2. Penelitian ini merupakan eksplorasi awal dalam penerapan model video understanding yang mungkin dapat berkembang lebih lanjut dalam penelitian berikutnya.

1.5 Batasan Masalah

1. Teks yang digunakan umum, tidak menyertakan nama orang atau lainnya yang spesifik dan hanya mencakup kosakata yang terdapat dalam dataset.
2. Penelitian ini tidak berfokus pada klasifikasi atau batasan aksi dalam video, melainkan pada evaluasi metode *Cross-Attention Video Vision Transformer* dalam menghasilkan deskripsi video.

BAB II

KAJIAN PUSTAKA

2.1 Penelitian Terkait

Tabel 2.1: Penelitian terkait

Author(s)	Judul	Metode dan Dataset	Hasil Penelitian
S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko (2015)	Sequence to Sequence – Video to Text	CNN + LSTM Encoder-Decoder dan MSVD [17], MPII-MD [18], MVAD [19] dataset	Menggunakan CNN untuk ekstraksi fitur dari video dan LSTM sebagai encoder-decoder untuk menghasilkan deskripsi video.
C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid (2019)	VideoBERT: A Joint Model for Video and Language Representation Learning	VideoBERT (Transformer-based) + S3D pretrained dan YouCook2 [20] dataset	Mengadaptasi BERT untuk video dan bahasa dengan pre-trained S3D. Memahami hubungan video dan teks secara kontekstual.
A. Dosovitskiy et al. (2020)	An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale	Vision Transformer (ViT) dan ImageNet [21], ImageNet-21k [22], JFT-300M [23] dataset	Memperkenalkan konsep patching gambar 2D untuk memproses data visual menggunakan Transformer.
A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lucic, and C. Schmid (2021)	ViViT: A Video Vision Transformer	Video Vision Transformer (ViViT) dan Kinetics [24], Epic Kitchens [25], Moments in Time [26], SSv2 [27] dataset	Menggunakan patching 2D dan 3D untuk ekstraksi fitur spasial-temporal secara langsung dari video.

Penelitian sebelumnya (konvensional) umumnya menggunakan CNN/3D CNN [28], dan LSTM [5, 7, 11, 12], seperti Venugopalan et al. (2015) [11] yang mengusulkan model *Encoder-Decoder* LSTM. Seiring berkembangnya arsitektur Transformer, fokus penelitian mulai bergeser. VideoBERT (Sun et al., 2019) [14] mengadaptasi konsep BERT [29] dan *pretrained* S3D [30] untuk memahami video dan bahasa. *Vision Transformer* (ViT) oleh Dosovitskiy et al. (2020) [31] juga diterapkan ke video oleh Arnab et al. (2021) [15] dengan teknik *patching* 2D dan 3D untuk ekstraksi fitur secara

langsung, terbukti efektif dalam *video recognition* [15]. Penelitian ini menggunakan pendekatan *Cross-Attention* yang menggunakan *attention* antara fitur spasial 3D dengan fitur spasial-temporal dan fitur temporal dengan fitur spasial-temporal. Penggunaan fitur spasial 3D karena fitur spasial 3D memiliki informasi spasial-temporal sekaligus dan *patch* yang dihasilkan menjadi tidak terlalu banyak.

2.2 Deskripsi Video

Deskripsi video (*video captioning*) merupakan tugas menghasilkan teks secara otomatis berdasarkan isi video dengan memahami aksi dan peristiwa yang terjadi di dalamnya. Model *video captioning* dilatih menggunakan data multimodal berupa pasangan video dan teks. *Video captioning* dapat membantu dalam mengambil informasi dari video secara lebih efisien melalui representasi teks [32].

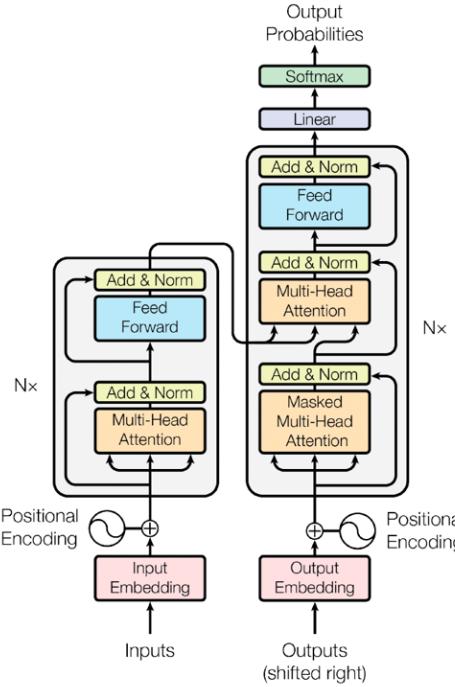
2.3 Tokenisasi dan *Word Embedding*

Tokenisasi adalah proses memecah teks menjadi unit kecil yang disebut token, seperti kata atau sub-kata, yang kemudian digunakan sebagai input untuk model [33]. Setelah ditokenisasi, setiap token direpresentasikan dalam bentuk vektor berdimensi tetap menggunakan teknik *word embedding*, yang memungkinkan model memahami hubungan semantik antar kata. Representasi ini disimpan dalam matriks berukuran *Embedding Dimension* \times *Vocabulary Size* [34].

Terdapat dua jenis embedding utama, yaitu embedding statis seperti *Word2Vec* dan *GloVe*, yang memberikan satu vektor tetap untuk setiap kata, serta embedding kontekstual seperti BERT, yang menghasilkan vektor berbeda tergantung konteks kalimat (hasil dari pelatihan BERT) [29]. Selain itu, embedding dapat berupa hasil pelatihan sebelumnya (*pre-trained*) atau dilatih langsung selama proses pembelajaran model (*trainable*) [35]. Pemilihan metode embedding yang tepat penting untuk meningkatkan kualitas hasil pada tugas-tugas seperti klasifikasi teks, *machine translate*, teks generasi, dan deskripsi gambar atau video.

2.4 Transformer

Transformer pertama kali diperkenalkan oleh Vaswani et al. pada tahun 2017 melalui *paper* berjudul "*Attention is All You Need*". Arsitektur Transformer terdiri dari dua komponen utama, yaitu *Encoder* dan *Decoder*. Transformer menghasilkan output berdasarkan representasi *Encoder* sebagai *Key* dan *Value* lalu output dari *masked multi-head attention* sebagai *Query* [13, 36].



Gambar 2.1: Arsitektur Transformer dengan *multi-head attention*, *Layer Normalization*, dan *Feed Forward Neural Network* disetiap blocknya [13].

Transformer tidak otomatis mengenal urutan seperti RNN, Oleh karena itu, harus menambahkan informasi urutan yaitu *Positional Encoding* sebagai penanda urutan. *Positional Encoding* memiliki dimensi yang sama dengan *embedding* input, sehingga keduanya dapat dijumlahkan [13]. *Positional Encoding* didefinisikan sebagai berikut:

$$PE(pos, 2i) = \sin(pos/10000^{2i/d_{embed}}) \quad (2.1)$$

$$PE(pos, 2i + 1) = \cos(pos/10000^{2i/d_{embed}}) \quad (2.2)$$

Di mana *pos* adalah posisi, $2i$ dan $2i + 1$ adalah indeks genap dan ganjil dalam dimensi *embedding*. *Positional Encoding* membentuk panjang gelombang dengan frekuensi berbeda disetiap dimensi untuk menangkap posisi yang berdekatan dan berjauhan. Komponen utama dalam Transformer adalah *self-attention* yang terdiri dari tiga vektor utama: *Query* (\mathbf{Q}), *Key* (\mathbf{K}), dan *Value* (\mathbf{V}) yang didapat dari lapisan *linear*. Proses *attention* dilakukan dengan menghitung hasil perkalian *dot product* antara \mathbf{Q} dan \mathbf{K} , kemudian dinormalisasi dengan akar dari dimensi *head* dan diteruskan ke fungsi *softmax*, sebelum akhirnya dikalikan dengan (\mathbf{V}) [13].

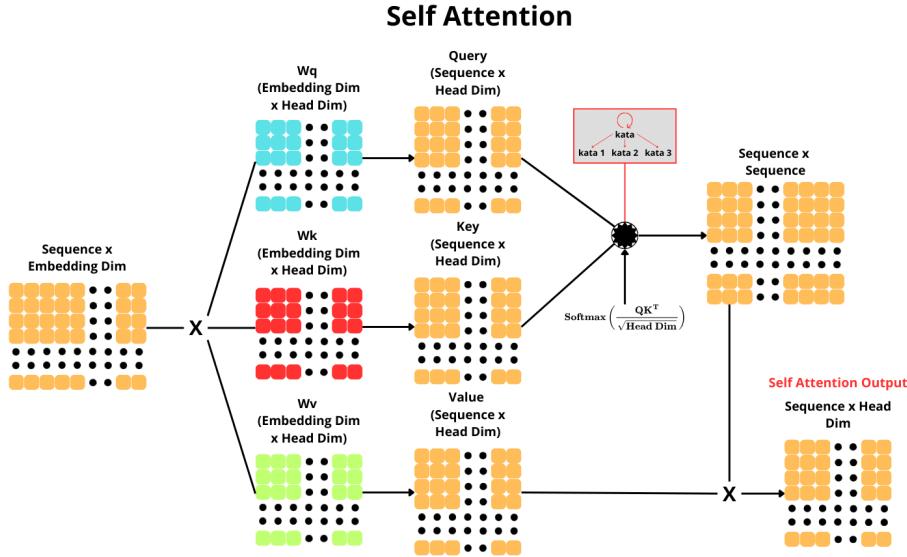
$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax} \left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{\text{Head Dim}}} \right) \mathbf{V} \quad (2.3)$$

dengan

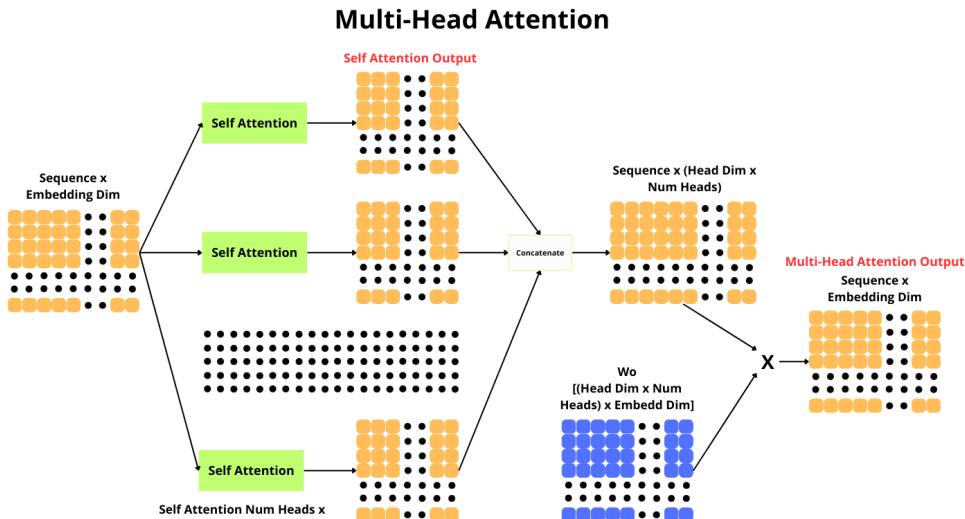
$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{x=1}^n e^{x_j}} \quad (2.4)$$

Transformer menggunakan *multi self-attention* atau *multi-head attention* dengan parame-

ter yang berbeda-beda [13]. Setiap output *attention* digabung dan melewati lapisan *linear* sehingga memiliki dimensi yang sama dengan input.



Gambar 2.2: Proses *self-attention* dengan satu *batch* input.



Gambar 2.3: *Multi-head attention*.

Pada bagian *Decoder*, lapisan awalnya menggunakan *masked multi-head attention* mencegah *attention* ke kata berikutnya. Hal ini penting karena saat proses inferensi, prediksi untuk token saat ini hanya boleh bergantung pada token-token sebelumnya. Setiap blok Transformer memiliki *Feed Forward Network* yang terdiri dari lapisan *non-linear* dan *linear* [13].

$$FFN(x) = \text{Activation}(xW_1 + b_1)W_2 + b_2 \quad (2.5)$$

Di mana W adalah bobot dan b adalah bias. Untuk menjaga kestabilan distribusi nilai selama proses pelatihan, Transformer menggunakan *Layer Normalization* untuk

menormalkan output dari *network*, sehingga distribusi nilainya lebih stabil selama *training*. Proses normalisasi ini dirumuskan sebagai berikut:

$$\text{LayerNorm}(x) = \frac{x - \mu}{\sigma + \epsilon} \cdot \gamma + \beta \quad (2.6)$$

Di mana γ dan β adalah parameter *trainable* dan ϵ adalah konstanta kecil (10^{-6}) [37].

2.5 Video Vision Transformer

Video Vision Transformer (ViViT) merupakan pendekatan berbasis *Transformer* yang dirancang untuk memproses data video. Berbeda dengan metode konvensional yang menggunakan jaringan konvolusional 3D atau jaringan saraf berulang (RNN), ViViT menggunakan arsitektur *Transformer* untuk memahami informasi spasial dan temporal dalam video secara langsung. ViViT telah menunjukkan performa yang menjanjikan dalam berbagai tugas pemrosesan video seperti pengenalan aksi, klasifikasi video, dan pembuatan deskripsi video [15, 16].

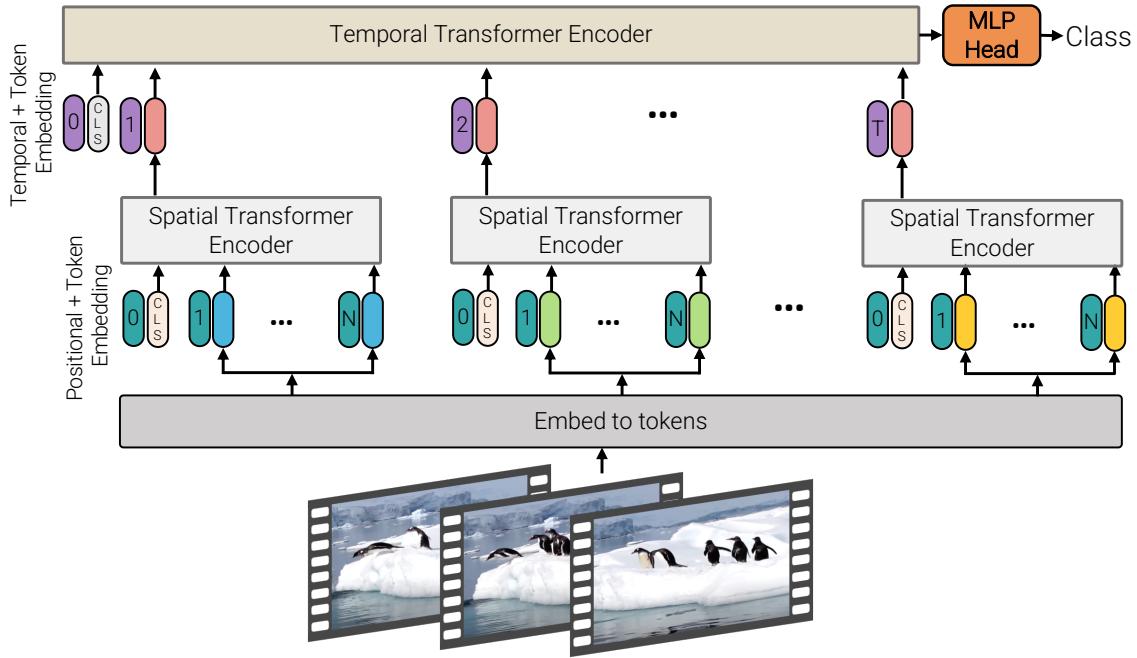
2.5.1 Proses Pembentukan *Embedding*

Video Vision Transformer (ViViT) menggunakan konsep yang sama seperti *Vision Transformer* yang memecah data menjadi *patch-patch* kecil. Video $V \in \mathbb{R}^{D \times W \times H \times C}$ dipecah menjadi $patch P \in \mathbb{R}^{P_d \times P_w \times P_h \times C}$, sehingga akan menghasilkan *sequence* token $\mathbf{Z} \in \mathbb{R}^{(D/P_d) \times (W/P_w) \times (H/P_h) \times d}$ dan \mathbf{Z} *reshaped* menjadi $\mathbf{Z} \in \mathbb{R}^{N \times d}$ di mana $N = (D/P_d) \times (W/P_w) \times (H/P_h)$ dan d adalah dimensi *embedding* yang didapat setelah melewati lapisan linear, di sini \mathbf{Z} adalah *Patch Embedding*. Arnab et al. memperkenalkan dua metode untuk menghasilkan *Patch Embedding* yaitu *Uniform Frame Sampling* yang menghasilkan *embedding* 2D ($P_d = 1$) di dimensi spasial secara independen untuk setiap *frame* dan *Tubelet Embedding* yang menghasilkan *embedding* 3D di dimensi spasial dan temporal [15]. Kemudian \mathbf{Z} akan ditambahkan dengan *Positional Encoding* dan akan menjadi input untuk *Encoder* *Transformer* [15]. Arnab et al. mengusulkan empat pendekatan dalam ViViT yaitu *Spatio-Temporal Attention*, *Factorised Encoder*, *Factorised Self-Attention*, dan *Factorised Dot Product Attention* [15]

2.5.2 Model 1: Spatio-Temporal Attention

Model ini secara langsung memasukkan *embedding* $\mathbf{Z} \in \mathbb{R}^{N \times d}$ ke dalam *multi-head attention*, sehingga jumlah *patch* atau panjang *sequence* yang diproses dalam *multi-head attention* menjadi sangat besar. Kondisi tersebut meningkatkan beban komputasi dan memperlambat proses *Dot Product Attention*, terutama saat memproses video dengan dimensi spasial-temporal yang kompleks. Setiap *patch* berkontribusi penuh dalam perhitungan *attention*, sehingga berdampak signifikan pada kompleksitas perhitungan. Panjang *sequence* yang besar menyebabkan kompleksitas waktu *self-attention* meningkat secara kuadratik, yaitu $O(N^2 \cdot d)$, dengan N sebagai panjang *sequence* dan d sebagai dimensi *embedding* [15].

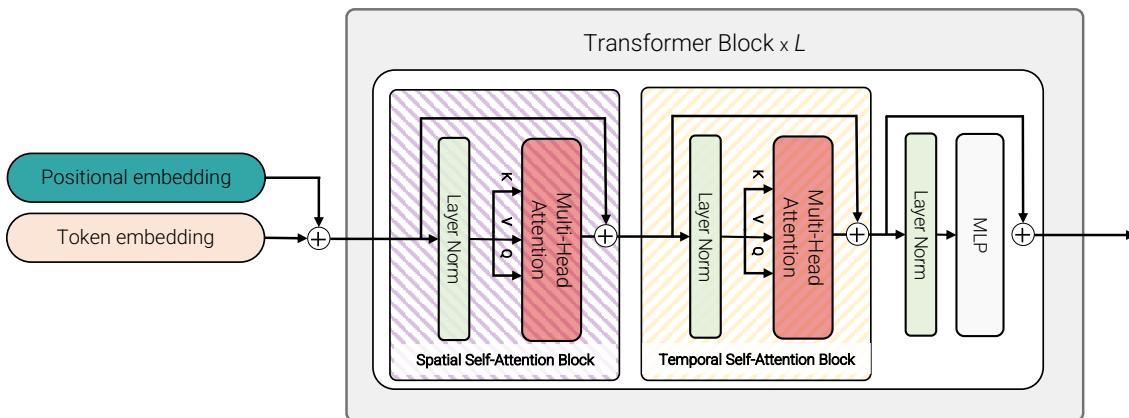
2.5.3 Model 2: Factorised Encoder



Gambar 2.4: *Factorised Encoder* [15]

Pada model ini *embedding* $\mathbf{Z} \in \mathbb{R}^{N \times d}$ dipisahkan untuk setiap frame $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_n]$, untuk setiap $\mathbf{Z}_i \in \mathbb{R}^{(W/P_w) \cdot (H/P_h) \times d}$ dilakukan *attention* terpisah dalam *Spatial Encoder*. Hasil dari *Spatial Encoder* memiliki dua pilihan yaitu output gabungan dari token [cls] (hanya untuk tugas klasifikasi) atau dilakukan *Global Average Pooling*, jika menggunakan *Global Average Pooling* akan memiliki jumlah parameter yang lebih sedikit karena tidak menggunakan token [cls] tetapi pada percobaan yang dilakukan pada dataset Kinetics 400 dan Epic Kitchens memiliki Top-1 akurasi lebih rendah dibanding menggunakan token [cls]. *Spatial Encoder* menghasilkan output \mathbf{h}_i yang kemudian di *concatenate* menjadi $\mathbf{H} \in \mathbb{R}^{(D/P_d) \times d}$. Selanjutnya \mathbf{H} digunakan sebagai input untuk *Temporal Encoder* [15].

2.5.4 Model 3: *Factorised Self-Attention*

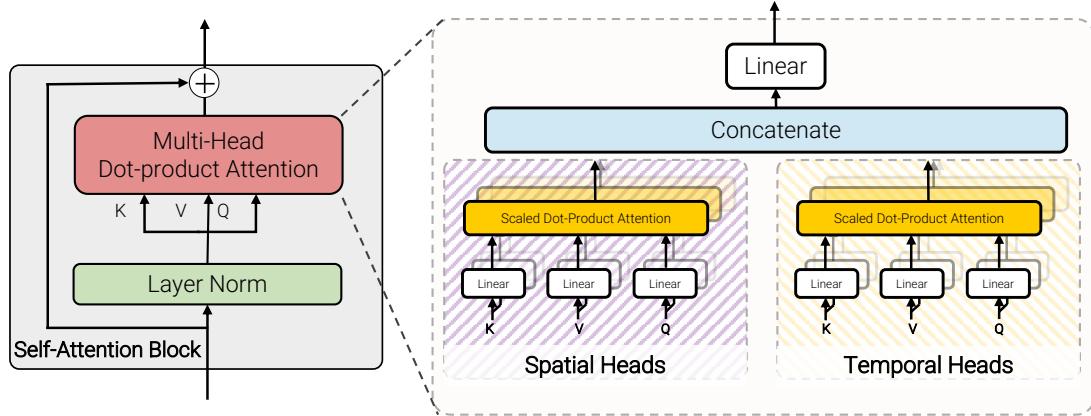


Gambar 2.5: *Factorised Self-Attention* [15]

Factorised Self-Attention menggunakan satu *Encoder* dengan dua *multi-head attention* (spasial dan temporal *attention*). Input *embedding* \mathbf{Z} dipisahkan untuk setiap frame

$\mathbb{R}^{(W/P_w) \cdot (H/P_h) \times d}$ sebagai input *Spatial Attention* dan dilakukan *attention* terpisah untuk setiap dimensi temporal seperti *Factorised Encoder* lalu output dari *Spatial Attention* dipisahkan untuk setiap spasial $\mathbb{R}^{(D/P_d) \times d}$ sebagai input *Temporal Attention* dan dilakukan *attention* terpisah untuk setiap dimensi spasial [15].

2.5.5 Model 4: Factorised Dot Product Attention



Gambar 2.6: Factorised Dot Product Attention [15]

Factorised Dot Product Attention menggunakan *Key* dan *Value* terpisah untuk spasial dan temporal dengan *embedding* $\mathbf{Z} \in \mathbb{R}^{N \times d}$ sebagai input *Query*, jika \mathbf{Q} didapat dari $\mathbf{Z} \in \mathbb{R}^{N \times d}$, dan $\mathbf{K}_s, \mathbf{V}_s$ didapat dari $\mathbf{Z} \in \mathbb{R}^{(W/P_w) \cdot (H/P_h) \times d}$, lalu $\mathbf{K}_t, \mathbf{V}_t$ didapat dari $\mathbf{Z} \in \mathbb{R}^{(D/P_d) \times d}$, maka $\mathbf{Y}_{s_i} = \text{Attention}(\mathbf{Q}, \mathbf{K}_s, \mathbf{V}_s)$ dan $\mathbf{Y}_{t_i} = \text{Attention}(\mathbf{Q}, \mathbf{K}_t, \mathbf{V}_t)$ yang masing-masing sebanyak setengah dari jumlah *heads attention*. Sehingga output dari *multi-head attention* yaitu $\mathbf{Y} = \text{Concat}(\mathbf{Y}_s, \mathbf{Y}_t)\mathbf{W}_o$ [15].

2.6 3D Convolutional Neural Network

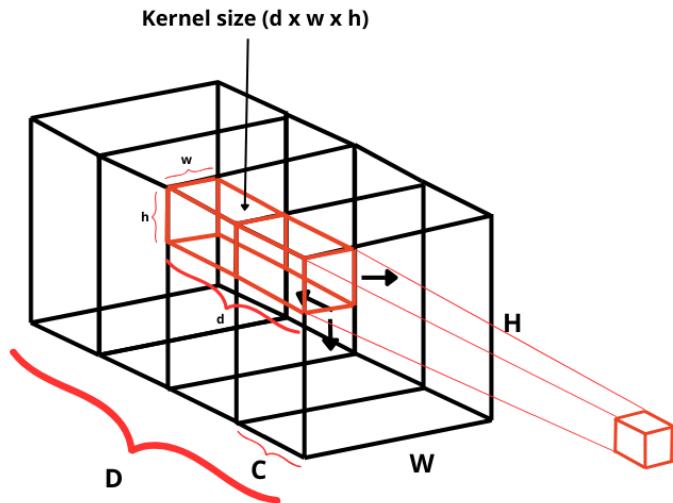
Pada proses konvolusi 2D, kernel melintasi dimensi spasial dan melakukan konvolusi antara kernel dengan bagian input [38, 39]. Misal terdapat gambar input I dengan satu channel berukuran $H \times W$, kernel K dengan ukuran $h \times w$, dan output hasil konvolusi (*feature map*) O berukuran $(H - h/S + 1) \times (W - w/S + 1)$ dengan S adalah *stride*. Output O pada posisi (i, j) didefinisikan sebagai [40]:

$$O(i, j) = \sum_{m=0}^{h-1} \sum_{n=0}^{w-1} I(iS + m, jS + n)K(m, n) \quad (2.7)$$

Jika ditinjau dari Persamaan 2.7 konvolusi 3D melibatkan dimensi tambahan yaitu depth (kedalaman), di mana satu gambar mewakili satu frame [41]. Misal input I dengan satu channel memiliki ukuran $D \times W \times H$, kernel K berukuran $d \times w \times h$, dan output O berukuran $((D - d)/S + 1) \times ((W - w)/S + 1) \times ((H - h)/S + 1)$ dengan S adalah *stride*. Output O pada posisi (i, j, k) didefinisikan sebagai:

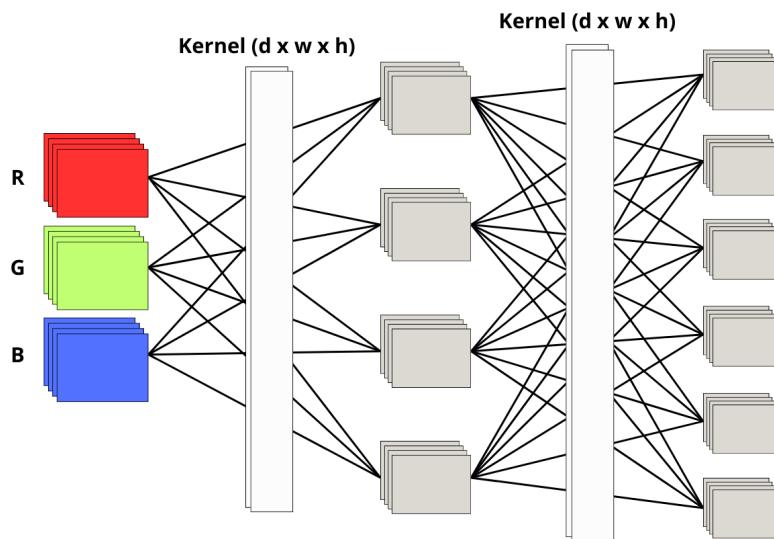
$$O(i, j, k) = \sum_{m=0}^{d-1} \sum_{n=0}^{w-1} \sum_{p=0}^{h-1} I(iS + m, jS + n, kS + p)K(m, n, p) \quad (2.8)$$

Konvolusi 3D bergerak tidak hanya ke samping dan ke bawah, tetapi juga ke belakang [41] yang dapat dilustrasikan pada Gambar 2.7.



Gambar 2.7: Ilustrasi proses konvolusi yang akan bergerak ke samping, ke bawah, dan ke belakang (3D).

Pada CNN, satu filter terdiri dari beberapa kernel yang dipelajari selama pelatihan [38–40]. Setiap kernel melakukan konvolusi pada satu channel, dan hasilnya dijumlahkan dan melewati fungsi aktivasi. Sehingga 3D CNN dengan 3 channel input dapat diilustrasikan seperti Gambar 2.8



Gambar 2.8: 3D Convolutional Neural Network.

2.7 Metrik CIDEr

CIDEr merupakan metrik evaluasi yang dirancang secara khusus untuk tugas *captioning*. CIDEr menghitung kesamaan antara caption hasil prediksi dan caption referensi dengan

memanfaatkan skor TF-IDF dari setiap n -gram yang muncul. Skor TF-IDF dari n -gram k pada kalimat referensi s_{ij} dihitung sebagai berikut:

$$g_k(s_{ij}) = \frac{h_k(s_{ij})}{\sum_{\omega_l \in \Omega} h_l(s_{ij})} \log \left(\frac{|I|}{\sum_{I_p \in I} \min(1, \sum_q h_k(s_{pq}))} \right) \quad (2.9)$$

Di mana $h_k(s_{ij})$ menunjukkan frekuensi kemunculan n -gram k dalam kalimat s_{ij} , dan $|I|$ adalah jumlah total gambar atau video dalam dataset. Setelah menghitung representasi TF-IDF, kesamaan antara caption prediksi c_i dan himpunan caption referensi S_i dihitung menggunakan cosine similarity sebagai berikut:

$$\text{CIDErn}(c_i, S_i) = \frac{10}{m} \sum_j \frac{\mathbf{g}^n(c_i) \cdot \mathbf{g}^n(s_{ij})}{|\mathbf{g}^n(c_i)| |\mathbf{g}^n(s_{ij})|} \quad (2.10)$$

Perhitungan ini dilakukan untuk setiap n -gram dengan n mulai dari 1 hingga N , dan skor akhir CIDEr diperoleh dengan merata-ratakan hasil keseluruhan n -gram:

$$\text{CIDEr}(c_i, S_i) = \frac{1}{N} \sum_{n=1}^N \text{CIDEr}_n(c_i, S_i) \quad (2.11)$$

Metrik ini menangkap kualitas caption dengan mempertimbangkan pentingnya setiap n -gram dalam konteks dataset, sekaligus memperhatikan keseimbangan antara frekuensi dan relevansi. Dalam implementasinya, CIDEr umumnya menggabungkan skor dari berbagai n -gram dengan $n = 1, 2, 3, 4$ [42].

BAB III

METODOLOGI PENELITIAN

3.1 Deskripsi Penelitian

Penelitian ini memodifikasi arsitektur *Video Vision Transformer* sebagai salah satu pendekatan model berbasis Transformer yang banyak digunakan dalam pemrosesan data video untuk meningkatkan efisiensi dalam proses ekstraksi fitur spasial-temporal dari video berdimensi tinggi. Modifikasi yang dilakukan bertujuan untuk mengurangi kompleksitas komputasi sekaligus mempertahankan kualitas representasi fitur yang dihasilkan oleh model. Fokus utama penelitian ini terletak pada proses perancangan dan evaluasi arsitektur model yang dioptimalkan secara khusus guna mempercepat proses inferensi pada tugas deskripsi video, sehingga mampu menghasilkan output teks yang merepresentasikan isi video secara lebih cepat dan akurat.

Model yang diusulkan dalam penelitian ini diharapkan tidak hanya mampu meningkatkan kecepatan pemrosesan video secara signifikan, tetapi juga tetap menjaga performa model dalam memahami konteks dan alur kejadian dalam video. Dengan demikian, model hasil modifikasi ini diharapkan menjadi lebih efektif dan efisien untuk diterapkan dalam berbagai aplikasi berbasis kecerdasan buatan yang membutuhkan proses otomatisasi pengenalan dan deskripsi video secara *real-time*, seperti sistem pencarian video berbasis teks, pemantauan video cerdas, serta aplikasi multimedia lainnya yang membutuhkan interpretasi video secara otomatis dan cepat.

3.2 Deskripsi Data

Dataset yang digunakan adalah *Microsoft Video Description* (MSVD), yang diterjemahkan ke dalam bahasa Indonesia oleh Hendria dengan bantuan API *Google Translate* [43]. Dataset ini terdiri dari sekitar 80000 pasangan video-teks aktivitas sehari-hari manusia dan hewan, dengan total 1970 video, di mana setiap video memiliki beberapa pasangan teks untuk tugas-tugas multimodal video-teks, termasuk pembuatan deskripsi video seperti pada penelitian ini [17, 43].

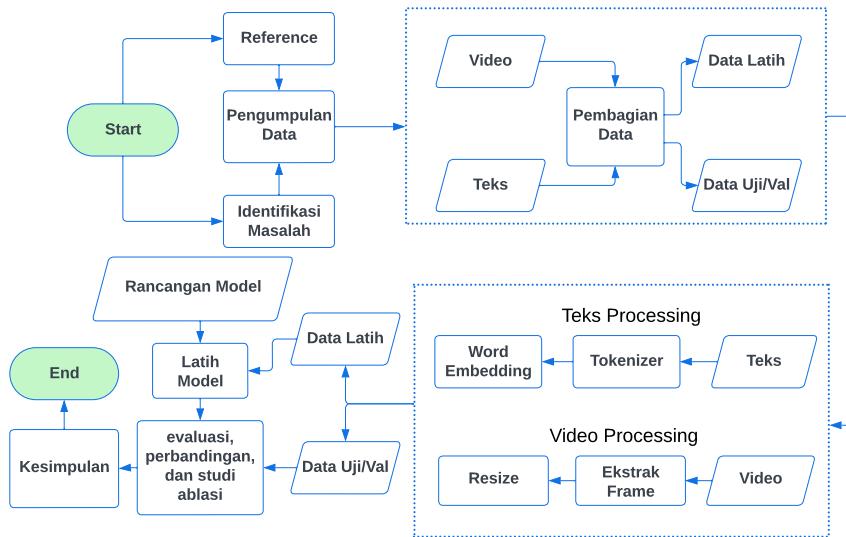
Tabel 3.1: Contoh video dengan captionnya dalam dataset.

Video	Caption
	Ground Truth 1: tupai sedang makan kacang. Ground Truth 2: chipmunk sedang makan. Ground Truth 3: tupai makan buah kacang.

Dataset MSVD digunakan sebagai benchmark standar, seperti penelitian pengembangan metode lainnya [5, 7, 11, 12]. Penelitian ini tidak membatasi kategori aksi dalam video, karena fokus utama adalah mengevaluasi metode *Cross-Attention ViViT* yang lebih cepat.

Dataset ini digunakan untuk mengevaluasi performa model secara umum terhadap video dengan variasi aksi yang beragam.

3.3 Tahapan Penelitian



Gambar 3.1: Flowchart rancangan penelitian.

3.3.1 Persiapan Penelitian

Tahapan awal penelitian ini adalah mengidentifikasi masalah terkait pengembangan pendekatan untuk menghasilkan deskripsi video otomatis yang mampu menangkap pola spasial dan temporal serta konteks yang kuat antar frame, dengan mempertimbangkan komputasi yang masih terjangkau oleh pengguna gratisan. Oleh karena itu dilakukan pencarian referensi literatur tentang perkembangan model *Deep Learning* yang relevan dengan penelitian ini [5–7, 11–14, 28]. Untuk memastikan keberhasilan eksperiment, penelitian ini menggunakan data dengan pola yang beragam seperti aktivitas sehari-hari.

3.3.2 Persiapan Data

Pada tahap ini, setiap pasangan video dan teks dikemas ke dalam struktur *dictionary* untuk mempermudah proses pemetaan antara fitur video dan teks deskriptif. Setiap teks dilengkapi dengan token khusus <start> di awal kalimat sebagai penanda mulai dan token <end> di akhir kalimat sebagai penanda akhir. Token <start> berfungsi sebagai sinyal bagi *Decoder* untuk memulai proses generasi kalimat selama inferensi, memungkinkan model memproduksi *caption* dengan panjang yang bervariasi. Proses *decoder* akan berlanjut hingga token <end> terdeteksi, yang secara otomatis menghentikan prediksi kata berikutnya dan menandakan bahwa *caption* yang dihasilkan telah selesai. Sebanyak 20% dari keseluruhan data dialokasikan sebagai data uji/validasi yang digunakan untuk mengevaluasi kinerja model selama proses pelatihan serta menguji kemampuan model dalam menghasilkan *caption* yang sesuai dengan konten video.

3.3.3 Pemrosesan Teks

Pemrosesan teks diawali dengan proses tokenisasi. Token dalam penelitian ini direpresentasikan sebagai kata-kata tunggal yang membentuk kalimat deskripsi video. Proses tokenisasi bertujuan untuk memudahkan konversi teks ke dalam representasi numerik yang dapat diproses oleh model. Setelah proses tokenisasi, setiap token diubah menjadi bilangan bulat sebagai indeks yang mengacu pada posisi token tersebut dalam *vocabulary* yang telah ditentukan. Indeks-indeks ini kemudian dipetakan ke dalam ruang vektor menggunakan *Word Embedding*. Representasi vektorial ini memungkinkan model untuk memahami hubungan semantik antar-kata berdasarkan kedekatan vektor dalam ruang berdimensi tinggi. Vektor-vektor hasil embedding inilah yang digunakan sebagai input ke dalam *decoder* pada tahap pelatihan dan inferensi.

3.3.4 Pemrosesan Video

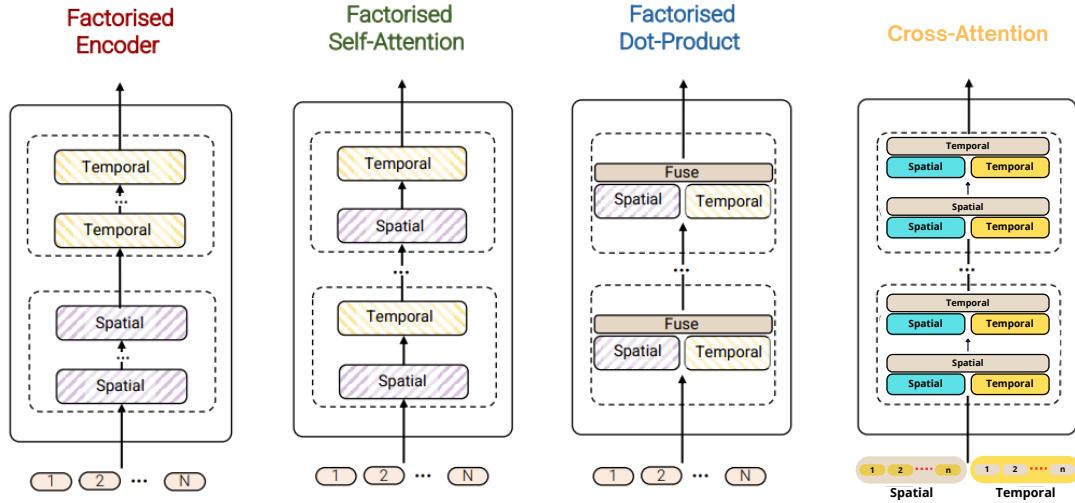
Pemrosesan video dilakukan melalui beberapa tahapan untuk menyiapkan data dalam format yang sesuai untuk model. Tahap pertama adalah ekstraksi frame, di mana video diubah menjadi serangkaian gambar yang berurutan [44]. Setiap frame merepresentasikan satu cuplikan waktu dalam video. Setelah frame diekstraksi, dilakukan proses *resize* pada setiap frame agar seluruh frame memiliki dimensi spasial yang seragam. Penyesuaian ukuran ini penting untuk memastikan konsistensi dalam pemrosesan batch data serta kompatibilitas dengan arsitektur model yang digunakan.

Selain normalisasi ukuran, penyesuaian jumlah frame juga dilakukan untuk memastikan setiap video memiliki jumlah frame yang sama. Hal ini bertujuan agar video dapat diproses secara paralel dalam satu batch tanpa menyebabkan ketidaksesuaian dimensi. Penyamaan jumlah frame dilakukan dengan dua kondisi utama: jika total frame dalam video kurang dari atau sama dengan jumlah maksimum frame yang ditentukan, maka seluruh frame digunakan; namun jika melebihi batas tersebut, dilakukan proses *uniform sampling*.

Teknik *uniform sampling* secara merata memilih sejumlah frame dari awal hingga akhir video menggunakan perhitungan interval yang sama. Pendekatan ini memastikan bahwa representasi visual dari keseluruhan durasi video tetap terjaga meskipun terjadi reduksi jumlah frame. Proses ini tidak hanya menjaga informasi temporal penting dalam video, tetapi juga membantu mengontrol kompleksitas komputasi selama pelatihan dan inferensi, sehingga model tetap efisien dalam memproses data berukuran besar.

3.3.5 Rancangan Arsitektur Model

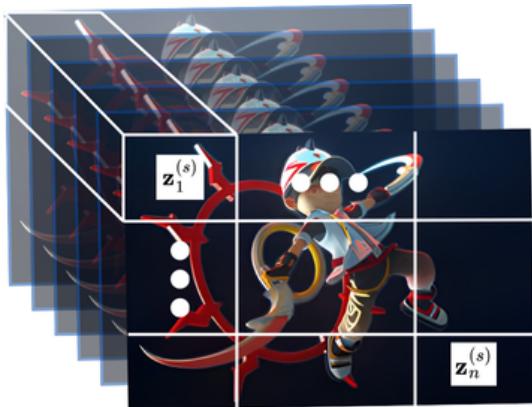
Penelitian ini bertujuan untuk membandingkan kinerja *Cross-Attention Video Vision Transformer* dengan empat pendekatan *Video Vision Transformer* lainnya, sekaligus melakukan studi ablasi guna menginvestigasi peran dan pengaruh komponen *Cross-Attention* dalam arsitektur yang diusulkan.



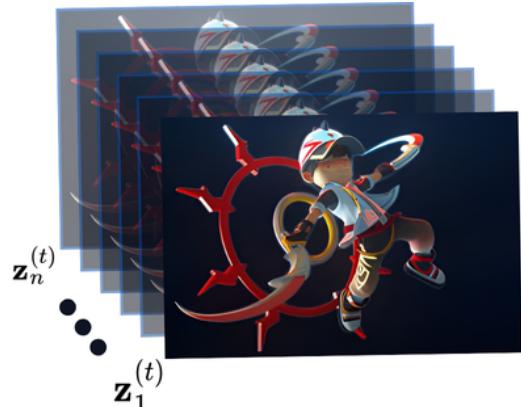
Gambar 3.2: Beberapa variasi model ViViT [15] dan *Cross-Attention* yang memproses token spasial dan temporal.

3.3.5.1 Input Embedding Video

Input video $V \in \mathbb{R}^{D \times W \times H \times C}$ dibagi menjadi 2 *embedding* yaitu *Spatial Embedding* $\mathbf{Z}_s = [\mathbf{z}_1^{(s)}, \mathbf{z}_2^{(s)}, \dots, \mathbf{z}_n^{(s)}]$ dan *Temporal Embedding* $\mathbf{Z}_t = [\mathbf{z}_1^{(t)}, \mathbf{z}_2^{(t)}, \dots, \mathbf{z}_n^{(t)}]$ di mana $\mathbf{z}_i^{(s)} \in \mathbb{R}^{(D/r) \times P_w \times P_h \times C}$ dan $\mathbf{z}_i^{(t)} \in \mathbb{R}^{P_d \times (W/r) \times (H/r) \times C}$ dengan r adalah rasio *resize*. *Resize* dapat dilakukan dengan *Convolutional 3D* secara *trainable* untuk dimensi temporal maupun spasial. *Spatial Embedding* dan *Temporal Embedding* dapat divisualisasikan pada Gambar 3.3 dan Gambar 3.4.

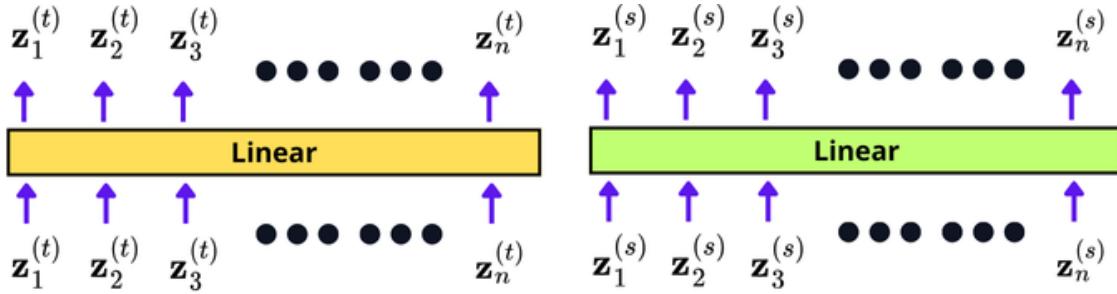


Gambar 3.3: *Spatial Embedding*.



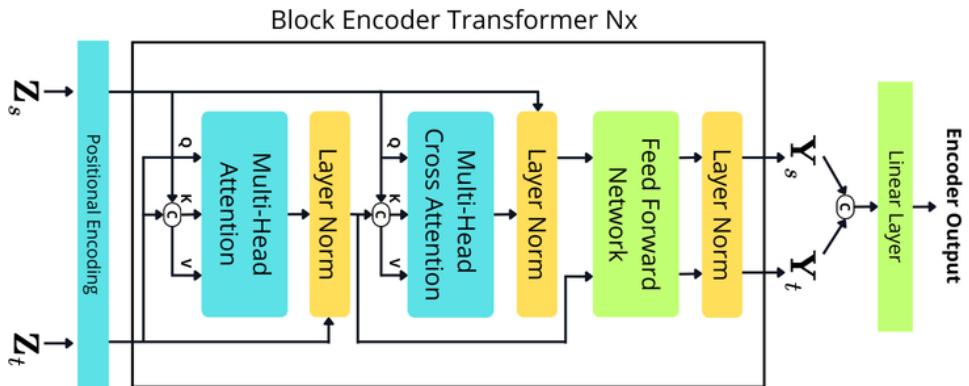
Gambar 3.4: *Temporal Embedding*.

Sebelum masuk *Encoder* $\mathbf{z}_i^{(s)}$ dan $\mathbf{z}_i^{(t)}$ ditransformasi ke dimensi yang sama secara *linear* menggunakan lapisan *linear* secara *trainable* untuk memudahkan proses dalam *Encoder* karena akan berpengaruh kepada banyak parameter yang digunakan dan penggunaan *residual connection* yang memerlukan dimensi yang sama pada proses *concatenate*.



Gambar 3.5: Linear layer untuk mentransformasi $\mathbf{z}_i^{(t)} \in \mathbb{R}^{P_d \times (W/r) \times (H/r) \times C}$ menjadi $\mathbf{z}_i^{(t)} \in \mathbb{R}^d$ dan $\mathbf{z}_i^{(s)} \in \mathbb{R}^{(D/r) \times P_w \times P_h \times C}$ menjadi $\mathbf{z}_i^{(s)} \in \mathbb{R}^d$.

3.3.5.2 Encoder Cross-Attention

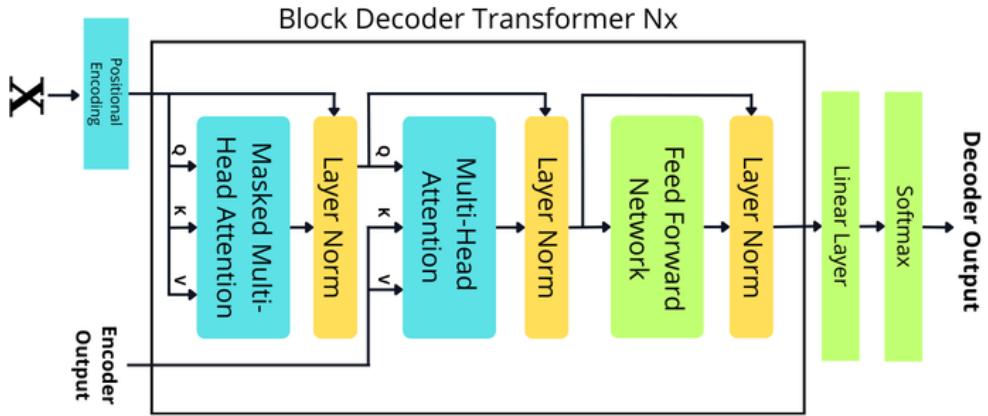


Gambar 3.6: Encoder arsitektur *Cross-Attention*.

Setelah *embedding* \mathbf{Z}_t dan \mathbf{Z}_s melewati *Positional Encoding*, *embedding* \mathbf{Z}_t (input *Query*) akan melakukan *Cross-Attention* dengan gabungan \mathbf{Z}_t dan \mathbf{Z}_s (*Key Value*) yang outputnya akan digabung dengan *embedding* \mathbf{Z}_s sebagai input *Key* dan *Value* untuk melakukan *Cross-Attention*, dengan \mathbf{Z}_s sebagai input *Query*. \mathbf{Z}_s diasumsikan lebih kaya akan informasi daripada \mathbf{Z}_t karena \mathbf{Z}_s berbentuk 3D yang menyimpan informasi spasial dan temporal, tetapi karena \mathbf{Z}_s tidak bisa melakukan *attention* secara temporal, \mathbf{Z}_t yang sudah memiliki konteks digunakan sebagai tambahan pada *Key* dan *Value* dengan idenya adalah bahwa \mathbf{Z}_s akan memperhatikan informasi spasial-temporal berdasarkan \mathbf{Z}_s dan \mathbf{Z}_t begitupun sebaliknya. Hasil kedua *attention* akan memasuki lapisan FFN untuk pemetaan *non-linear* dengan parameter yang sama, hasil akhir adalah *concatenate* dari kedua output yang diteruskan ke layer *linear* sebelum memasuki *Decoder*. Untuk urutan *Query* \mathbf{Z}_s dan \mathbf{Z}_t pada *Cross-Attention* akan dilakukan perbandingan untuk melihat mana yang lebih optimal.

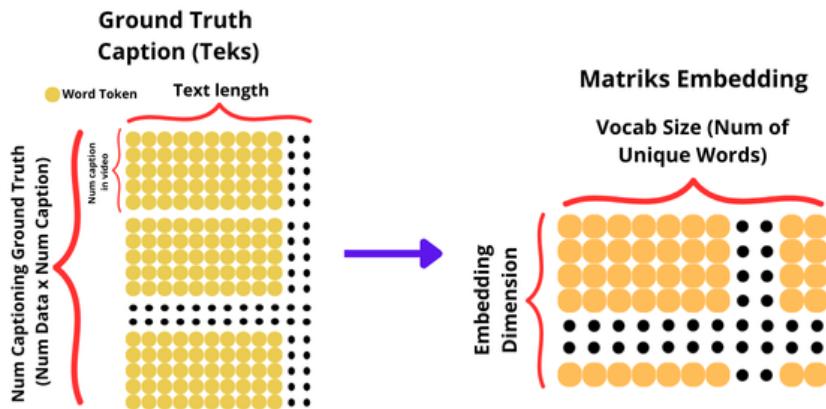
3.3.5.3 Decoder

Decoder akan menerima input *sequence* teks yang setiap katanya sudah menjadi bentuk vektor (*Word Embedding*) berdasarkan indeks *Vocabulary* pada *Matriks Embedding* (lihat Gambar 3.8). Jika \mathbf{X} adalah input *sequence* untuk *Decoder*, maka \mathbf{X} akan melewati lapisan *masked multi-head attention* yang menghasilkan proses *Autoregressive* seperti pada RNN untuk memprediksi kata berikutnya berdasarkan kata sebelumnya. Setiap kata dihasilkan berdasarkan representasi dari output *Encoder* sampai panjang maksimal



Gambar 3.7: *Decoder Transformer*.

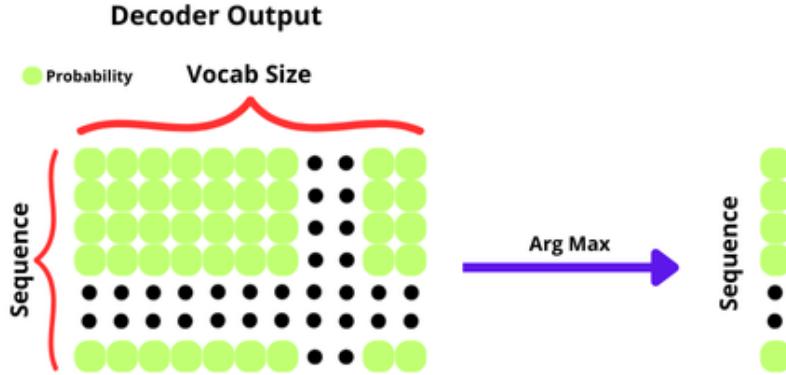
sequence. Untuk input *Decoder* ditambahkan token *<start>* sebagai input awal saat proses inferensi di mana input *sequence* teks tidak dimiliki, sedangkan untuk output ditambahkan token *<end>* sebagai tanda bahwa teks yang dihasilkan sudah berakhir sehingga menghasilkan panjang yang dinamis. Karena setiap video memiliki beberapa *caption*, maka \mathbf{X}_i mewakili satu *caption* untuk setiap video dalam batch dan proses *feed forward decoder* dilakukan sebanyak N kali atau sebanyak jumlah *caption* setiap video.



Gambar 3.8: *Matriks Embedding*.

3.3.5.4 *Output Sequence*

Dalam model *sequence-to-sequence* atau arsitektur berbasis urutan lainnya, output model sering kali direpresentasikan dalam bentuk tensor berdimensi $seq \times vocab$, di mana *seq* adalah panjang output *sequence* dan *vocab* adalah jumlah token atau kata dalam *vocabulary*. Untuk setiap langkah waktu i dalam output *sequence*, model menghasilkan vektor \mathbf{y}_i dari fungsi aktivasi softmax, di mana $\mathbf{y}_i \in \mathbb{R}^{vocab}$ adalah distribusi probabilitas atas kosakata pada waktu i dan $\mathbf{y}_i = p(\mathbf{x}_i | \mathbf{x}_{<i})$ di mana \mathbf{x} adalah input kata. Pada setiap langkah, token atau kata dengan probabilitas tertinggi dapat dipilih sebagai prediksi $y_i = \text{Arg Max}(\mathbf{y}_i)$.



Gambar 3.9: Output *sequence* menghasilkan teks deskripsi.

Dimensi $seq \times vocab$ sering digunakan dalam model *sequence-to-sequence* modern seperti penerjemah bahasa, teks generasi, konversi suara ke teks, deskripsi gambar atau video, dan lain sebagainya.

3.3.5.5 Loss Function

Fungsi kerugian akan menggunakan *Sparse Categorical Cross-Entropy* [45] untuk setiap kata/token dengan $p(x)$ adalah output probabilitas model:

$$\mathcal{L}_t = -\log(p(x)) \quad (3.1)$$

Untuk *loss* per *caption* dengan panjang m , *loss* per kata/token akan dirata-ratakan:

$$\mathcal{L}_c = \frac{\sum_{i=1}^m \mathbb{1}(y_i \neq 0) \cdot (-\log(p(x_i|x_{<i})))}{\sum_{i=1}^m \mathbb{1}(y_i \neq 0)} \quad (3.2)$$

Karena setiap video bukan hanya memiliki *caption* tunggal tetapi memiliki beberapa *caption*, maka *loss function* didefinisikan sebagai:

$$\mathcal{L} = \frac{1}{CB} \sum_{c=1}^C \sum_{b=1}^B \frac{\sum_{i=1}^m \mathbb{1}(y_{b,i}^{(c)} \neq 0) \cdot (-\log(p(x_{b,i}^{(c)}|x_{b,<i}^{(c)})))}{\sum_{i=1}^m \mathbb{1}(y_{b,i}^{(c)} \neq 0)} \quad (3.3)$$

Di mana C adalah *caption*, B adalah batch, dan $\mathbb{1}(y_{i,b}^{(c)} \neq 0)$ menunjukkan bahwa token bukan sama dengan 0 atau bukan token *padding*.

Untuk total gradien yang digunakan untuk pelatihan dari *loss function* \mathcal{L} terhadap probabilitas output $p(x)$ untuk setiap *sequence* token, setiap batch, dan setiap *caption* dihitung sebagai:

$$\frac{\partial \mathcal{L}}{\partial p(x)} = \begin{cases} -\frac{1}{CB\hat{m}} \sum_{c=1}^C \sum_{b=1}^B \sum_{i=1}^{\hat{m}} \frac{1}{p(x_{b,i}^{(c)}|x_{b,<i}^{(c)})} & \text{jika } y_{b,i}^{(c)} \neq 0 \\ 0 & \text{jika } y_{b,i}^{(c)} = 0 \end{cases} \quad (3.4)$$

Di mana $\hat{m} = \sum_{i=1}^m \mathbb{1}(y_{b,i}^{(c)} \neq 0)$. Dengan begitu token yang tidak valid atau token *padding* tidak akan mendapat gradien dan tidak akan dilatih. Persamaan 3.4 berlaku juga untuk gradien terhadap parameter model dengan menambahkan aturan rantai disetiap *loss* per *caption* pada setiap batch dan setiap *caption*.

3.3.6 Pelatihan Model

Model akan dilatih menggunakan *Kaggle Notebook* dengan 2 GPU NVIDIA Tesla T4 (15GB). Seluruh eksperimen dalam penelitian ini menggunakan konfigurasi arsitektur yang sama, yaitu terdiri dari 2 *encoder blocks* dan 2 *decoder blocks*, dengan 8 *heads* pada setiap *multi-head attention* dan dimensi *embedding* sebesar 512. Proses pelatihan dilakukan menggunakan *optimizer* Adam [13, 46] dengan nilai *hyperparameter* seperti *learning rate*, β_1 , dan β_2 mengikuti pengaturan yang digunakan oleh Vaswani et al. dalam *paper* "Attention is All You Need" [13].

Algoritma 1 Pelatihan model

```

1: Inisialisasi:  $m_0 \leftarrow 0, v_0 \leftarrow 0, t \leftarrow 0, \beta_1 \leftarrow 0.9, \beta_2 \leftarrow 0.98, \epsilon \leftarrow 10^{-9}$ 
2: Inisialisasi: epochs, batch_size
3: while epoch → epochs do
4:   while step → N // batch_size do
5:      $lr \leftarrow (d_{\text{embed}}^{-0.5} \cdot \min(\text{step}^{-0.5}, \text{step} \times 4000^{-1.5}))$ 
6:      $t \leftarrow t + 1$ 
7:      $g_t \leftarrow \nabla_{\theta} \mathcal{L}(\theta_{t-1})$  (Gradien loss function)
8:      $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  (Update momentum pertama)
9:      $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  (Update momentum kedua)
10:     $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Koreksi bias momentum pertama)
11:     $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Koreksi bias momentum kedua)
12:     $\theta_t \leftarrow \theta_{t-1} - lr \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Update parameter)
13:  end while
14: end while
15: Return: Parameter  $\theta_t$ 

```

3.3.7 Evaluasi dan Analisis Hasil

Evaluasi perbandingan model menggunakan nilai metrik yaitu akurasi yang dihasilkan, tetapi mungkin akurasi saja kurang maksimal. Oleh karena itu, pada penelitian ini akan menggunakan metrik tambahan yang dirancang khusus untuk tugas *captioning* yaitu CIDEr. Selain evaluasi performa berdasarkan metrik, penelitian ini juga melakukan analisis dan perbandingan kompleksitas masing-masing model, meliputi jumlah parameter, jumlah FLOPs, dan waktu eksekusi (*runtime*). Secara khusus, untuk model rancangan penelitian ini yang menggunakan mekanisme *Cross-Attention*, akan dilakukan Studi Ablasi untuk menganalisis kontribusi komponen tersebut terhadap performa model secara keseluruhan.

BAB IV

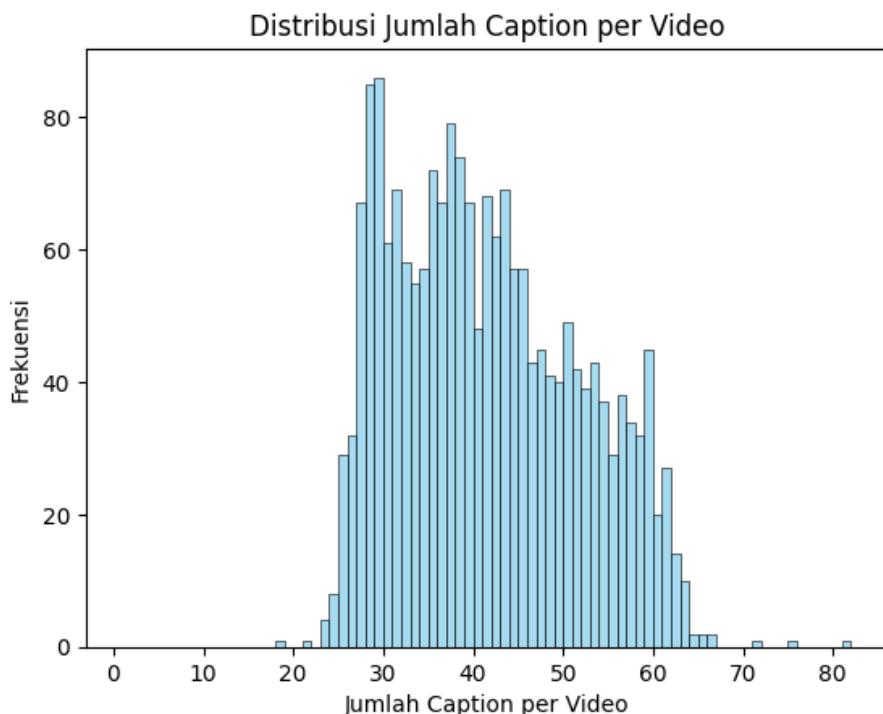
HASIL DAN PEMBAHASAN

4.1 Perbandingan Performa Variasi Model *Video Vison Transformer*

4.1.1 Eksplorasi Dataset

Dataset *Microsoft Video Description* (MSVD) memiliki total sebanyak 9.452 kata unik yang terdapat dalam seluruh *caption*. Berdasarkan jumlah tersebut, penelitian ini menetapkan ukuran *vocabulary* sebesar 10.000 untuk memastikan cakupan seluruh kosakata dalam dataset secara optimal selama proses pelatihan model.

Dalam dataset, setiap video memiliki jumlah *caption* yang bervariasi, sebagaimana ditunjukkan pada Gambar 4.1. Distribusi frekuensi menunjukkan bahwa sebagian besar video memiliki jumlah *caption* berkisar antara 25 hingga 65 dengan jumlah *caption* minimum adalah 18 dan jumlah *caption* maximum adalah 81. Pada penelitian ini, jumlah *caption* per video ditetapkan sebanyak 20 agar tidak terlalu membebani komputasi pelatihan dan tetap menjaga variasi *caption*. Untuk video dengan jumlah *caption* kurang dari 20, proses *broadcast* akan dilakukan agar jumlahnya sesuai. Sementara itu, untuk video yang memiliki lebih dari 20 *caption*, kelebihannya akan dipotong (*truncated*) sehingga hanya 20 *caption* yang digunakan.



Gambar 4.1: Distribusi jumlah *caption* setiap video.

Panjang *sequence* yang digunakan dalam penelitian ini adalah 20, sesuai dengan panjang maksimum *caption* yang terdapat pada dataset. Setiap *caption* yang memiliki jumlah kata kurang dari 20 akan diberikan *padding* dengan cara menduplikat *caption* yang ada hingga mencapai panjang tersebut.

4.1.2 Network Architecture

Eksperimen pada penelitian ini menggunakan *Tubelet Embedding* untuk keempat variasi model ViViT dengan ukuran patch sebesar $2 \times 16 \times 16 \times 3$. Input yang diberikan ke *Encoder* berukuran $16 \times 224 \times 224 \times 3$, yang terdiri dari 16 frame video dengan resolusi spasial 224×224 piksel dan 3 kanal warna (RGB).

Pada model *Cross-Attention* yang dirancang dalam penelitian ini, dilakukan proses *resizing* terhadap dimensi spasial dan temporal dengan rasio $r = 4$. Proses *resizing* ini dilakukan menggunakan dua lapisan *linear 3D Convolutional* [47] yang berfungsi sebagai *projection layer*, bukan sebagai ekstraksi fitur secara menyeluruh. Penerapan 3D *Convolutional Projection* ini bertujuan untuk mengurangi dimensi spasial dan temporal dan juga mengurangi jumlah parameter dan kompleksitas komputasi serta menyesuaikan jumlah patch temporal dan spasial pada model sebelum masuk ke dalam *Encoder Cross-Attention*. Konfigurasi terperinci ditunjukkan pada Tabel 4.1.

Tabel 4.1: Konfigurasi 3D Convolutional Projection

Layer	Filters (F)	Kernel Size (K)	Stride (S)
<i>Spatial Projection</i> Conv 1	4	(1, 3, 3)	(1, 2, 2)
<i>Spatial Projection</i> Conv 2	3	(2, 3, 3)	(2, 2, 2)
<i>Temporal Projection</i> Conv 1	4	(3, 1, 1)	(2, 1, 1)
<i>Temporal Projection</i> Conv 2	3	(3, 1, 1)	(2, 1, 1)

Penggunaan *stride* $(1, 2, 2)$ dan $(2, 2, 2)$ pada *Spatial Projection* akan mengurangi dimensi spasial sebesar faktor $r = 4$ atau sebesar 4, sehingga ukuran resolusi spasial berubah dari 224×224 menjadi 56×56 ($224/4 = 56$). Hal serupa juga diterapkan pada *Temporal Projection* dengan *stride* $(2, 1, 1)$ di setiap lapisan, yang mengurangi panjang dimensi temporal dari 16 frame menjadi 4 frame ($16/4 = 4$). Selain itu karena *stride* $(2, 2, 2)$ pada lapisan kedua *Spatial Projection* akan menghasilkan jumlah patch temporal (n_d) yang setara dengan patch berukuran $2 \times P_w \times P_h \times C$ pada *Tubelet Embedding*. Sehingga diperoleh jumlah patch temporal $n_d = 8$ dan jumlah patch spasial $(n_h \cdot n_w) = 196$ untuk semua model. *Source code* lengkap dari eksperimen ini dapat diakses di GitHub.

Penelitian ini tidak menggunakan *sinusoidal embedding* pada *Positional Encoding* seperti pada Persamaan 2.1 dan Persamaan 2.2, melainkan menerapkan *trainable embedding* agar representasi posisi dapat belajar secara lebih fleksibel dari data. Fungsi aktivasi GELUs [48] digunakan pada setiap lapisan *non-linear* dalam FFN [49]. Fungsi GELUs didefinisikan sebagai berikut [48]:

$$\text{GELU}(x) = x \cdot \frac{1}{2} \left(1 + \frac{2}{\sqrt{\pi}} \int_0^{\frac{x}{\sqrt{2}}} e^{-t^2} dt \right) \approx x \left(\frac{1}{1 + e^{-1.702x}} \right) \quad (4.1)$$

4.1.3 Variasi Model *Video Vison Transformer*

Pada penelitian ini, dilakukan eksperimen terhadap empat variasi model ViViT dan model rancangan yang diusulkan, yaitu *Spatio-Temporal Attention*, *Factorised Encoder*, *Factorised Self-Attention*, *Factorised Dot Product Attention* dan *Cross-Attention*. Untuk detailnya dapat dilihat pada Tabel 4.2.

Tabel 4.2: Variasi model

	Embedding	Encoder Backbone	Decoder
Model 1	<i>Tubelet Embedding</i> [15]	<i>Spatio-Temporal Attention</i> [15]	Transformer Decoder [13]
Model 2	<i>Tubelet Embedding</i> [15]	<i>Factorised Encoder</i> [15]	Transformer Decoder [13]
Model 3	<i>Tubelet Embedding</i> [15]	<i>Factorised Self-Attention</i> [15]	Transformer Decoder [13]
Model 4	<i>Tubelet Embedding</i> [15]	<i>Factorised Dot Product Attention</i> [15]	Transformer Decoder [13]
Model 5	<i>Spatial Temporal Embedding + Linear 3D Conv Projection</i>	<i>Cross-Attention</i>	Transformer Decoder [13]

Kelima variasi model yang telah dirangkum pada Tabel 4.2 dievaluasi dan dibandingkan kinerjanya pada tugas *video captioning* menggunakan dataset MSVD. Model 5 merupakan model rancangan dalam penelitian ini yang mengintegrasikan *Cross-Attention* dan 3D *Convolutional Projection* untuk mengurangi kompleksitas komputasi sekaligus mempertahankan performa model.

4.1.4 Evaluasi Model

Akurasi dan CIDEr digunakan sebagai metrik evaluasi utama untuk menilai kinerja model Video Vision Transformer yang disajikan pada Tabel 4.2. Dalam konteks ini, akurasi didefinisikan sebagai rata-rata proporsi token yang diprediksi dengan benar dalam setiap *sequence* (tidak termasuk token *padding*), yang selanjutnya dirata-ratakan di seluruh *caption* per video dan seluruh sampel dalam dataset.

Tabel 4.5 menyajikan hasil evaluasi dari lima arsitektur model yang berbeda, masing-masing diuji menggunakan dua metrik utama, yaitu *Sequence Accuracy* dan *CIDER Score*. Model dengan performa terbaik dalam hal *Sequence Accuracy* adalah *Factorised Dot Product Attention* yang menunjukkan bahwa model tersebut lebih akurat dalam menyusun urutan kata secara keseluruhan, sehingga menghasilkan kalimat yang lebih mendekati struktur referensi. Namun, skor CIDEr dari model ini lebih rendah dibandingkan dengan model *Spatio-Temporal Attention*, yang memiliki skor CIDEr tertinggi yang mencerminkan kemampuan model dalam menangkap makna dari setiap *n*-gram dalam kalimat dan unggul dalam hal pemahaman konteks umum hal ini wajar karena *Spatio-Temporal Attention* melakukan semua *attention* pada *embedding* $Z \in \mathbb{R}^{N \times d}$. Dalam tugas *captioning*, metrik CIDEr umumnya dianggap lebih penting karena menilai kesesuaian informasi semantik secara lebih menyeluruh. Seperti

dijelaskan dalam makalah “ViViT: A Video Vision Transformer” [15], model *Factorised Dot Product Attention* dinilai memiliki performa yang mendekati *Spatio-Temporal Attention*.

Sementara itu, model *Factorised Self-Attention* menunjukkan performa yang jauh lebih rendah pada kedua metrik. Hal ini kemungkinan disebabkan oleh pemisahan proses *attention* pada dimensi hasil *reshaping* yang kurang optimal untuk tugas *video captioning* karena daripada *reshaping* lebih baik terpisah seperti *Factorised Encoder*. Di sisi lain, model *Cross-Attention* menunjukkan performa yang cukup baik yang mendekati *Factorised Dot Product Attention* dan *Spatio-Temporal Attention* bahkan mampu mengungguli dengan konfigurasi paling optimal seperti hasil pada Tabel 4.7. Hasil ini menunjukkan bahwa pendekatan *Cross-Attention*, seperti yang ditampilkan pada Gambar 3.6, cukup efektif dalam mengekstraksi fitur semantik dari video.

Tabel 4.3: Perbandingan akurasi dan skor CIDEr model *Cross-Attention* dengan empat variasi model ViViT

Model	Sequence Accuracy	CIDEr Score
<i>Spatio-Temporal Attention</i>	0.7695	1.7183
<i>Factorised Encoder</i>	0.7516	1.4249
<i>Factorised Self-Attention</i>	0.3706	0.0958
<i>Factorised Dot Product Attention</i>	0.7809	1.6679
<i>Cross-Attention</i>	0.7740	1.6013

4.1.5 Analisis Kompleksitas

4.1.5.1 Perbandingan Kompleksitas Waktu

Kompleksitas waktu untuk *self-attention* adalah $O(n^2 \cdot d)$ [13] yang bertambah secara kuadratik untuk panjang *sequence* dan *linear* untuk dimensi *embedding*. Jika $n_d = (D/P_d)$, $n_h = (H/P_h)$, dan $n_w = (W/P_w)$ *Spatio-Temporal Attention* memiliki kompleksitas waktu $O(N^2 \cdot d)$ di mana $N = n_d \cdot n_h \cdot n_w$. *Factorised Encoder* memiliki kompleksitas waktu $O((n_h \cdot n_w)^2 \cdot d)$ sebanyak n_d atau $O((n_h \cdot n_w)^2 \cdot d \cdot n_d)$ ditambah dengan $O(n_d^2 \cdot d)$. *Factorised Self-Attention* memiliki kompleksitas waktu $O((n_h \cdot n_w)^2 \cdot (n_d \cdot d))$ ditambah dengan $O(n_d^2 \cdot (n_h \cdot n_w \cdot d))$. *Factorised Dot Product Attention* menggunakan *Key* dan *Value* terpisah sehingga kompleksitas waktunya adalah $O((N \cdot n_h \cdot n_w) \cdot d)$ ditambah $O((N \cdot n_d) \cdot d)$. Pada model yang penelitian ini usulkan (*Cross-Attention*) memiliki kompleksitas waktu $O(n_d \cdot (n_d + n_h \cdot n_w) \cdot d)$ ditambah $O((n_h \cdot n_w \cdot (n_h \cdot n_w + n_d)) \cdot d)$ yang dapat disederhanakan menjadi $O((n_d^2 + N) \cdot d + ((n_h \cdot n_w)^2 + N) \cdot d)$. Perbandingan kompleksitas waktu dapat dilihat pada tabel 4.4.

Tabel 4.4: Perbandingan kompleksitas waktu

Model	Kompleksitas Waktu
Spatio-Temporal Attention	$O(N^2 \cdot d)$
Factorised Encoder	$O((n_h \cdot n_w)^2 \cdot d \cdot n_d + n_d^2 \cdot d)$
Factorised Self-Attention	$O((n_h \cdot n_w)^2 \cdot n_d \cdot d + n_d^2 \cdot n_h \cdot n_w \cdot d)$
Factorised Dot Product Attention	$O(N \cdot n_h \cdot n_w \cdot d + N \cdot n_d \cdot d)$
Cross-Attention	$O((n_d^2 + N) \cdot d + ((n_h \cdot n_w)^2 + N) \cdot d)$

Berdasarkan Tabel 4.4, dapat diamati bahwa model dengan kompleksitas waktu paling rendah adalah *Cross-Attention*. Kompleksitas waktu *Cross-Attention* adalah $O((n_d^2 + N) \cdot d + ((n_h \cdot n_w)^2 + N) \cdot d)$, yang secara umum lebih ringan dibandingkan model lain. Berbeda dengan *Factorised Dot Product Attention* yang masih menggunakan seluruh dimensi pada proses *attention*, penelitian ini menekan pertumbuhan kompleksitas kuadrat *Cross-Attention* dan mengoptimalkan beban komputasi dengan benar-benar memisahkan kompleksitas antar dimensi spasial dan dimensi temporal.

4.1.5.2 Evaluasi Kompleksitas

Kompleksitas dari berbagai varian model Video Vision Transformer (ViViT) yang ditampilkan pada Tabel 4.2 dibandingkan berdasarkan jumlah parameter yang digunakan oleh model, nilai GFLOPs (*Giga Floating Point Operations Per Second*) sebagai indikator beban komputasi, serta kecepatan waktu inferensi (*runtime*) yang menggambarkan seberapa cepat model dapat melakukan prediksi terhadap data video masukan yang diperlihatkan pada Tabel 4.5.

GFLOPs atau setara dengan 1 miliar FLOPs (*Floating Point Operations Per Second*) yang digunakan oleh model *Cross-Attention* jauh lebih rendah dan memiliki waktu inferensi yang lebih cepat dibandingkan pada model lainnya, meskipun model *Cross-Attention* memiliki jumlah parameter yang lebih banyak dibandingkan model lainnya. Hal ini menunjukkan bahwa *Cross-Attention* memerlukan lebih sedikit operasi komputasi, yang berdampak langsung pada penggunaan sumber daya dan waktu inferensi yang lebih singkat. Efisiensi ini menjadikan *Cross-Attention* memiliki keuntungan terhadap penggunaan perangkat keras, baik dari segi kapasitas memori maupun kecepatan pemrosesan dan mungkin lebih efisien untuk aplikasi dengan keterbatasan sumber daya. Namun model lainnya juga tidak menjadi masalah untuk sumber daya komputasi yang kuat seperti penggunaan GPU atau TPU [50].

Sama seperti yang ditunjukkan pada hasil eksperimen ViViT [15], model *Spatio-Temporal Attention* memiliki kompleksitas paling tinggi, baik dari segi jumlah FLOPs maupun waktu komputasi (*runtime*). Kondisi ini memunculkan kebutuhan akan varian yang lebih ringan dan sekaligus menjadi motivasi dalam pengembangan arsitektur yang lebih efisien [15, 51].

Tabel 4.5: Jumlah parameter, GFLOPs, dan Runtime

Model	Params	GFLOPs	Runtime
<i>Spatio-Temporal Attention</i>	19.486 M	6.576	19.103
<i>Factorised Encoder</i>	22.642 M	6.342	8.578
<i>Factorised Self-Attention</i>	21.589 M	6.349	12.733
<i>Factorised Dot Product Attention</i>	19.224 M	6.319	8.090
<i>Cross-Attention</i>	26.760 M	1.843	5.294

Penelitian ini juga telah menemukan model *Cross-Attention* yang lebih optimal dengan jumlah parameter yang lebih sedikit seperti yang ditunjukkan pada Tabel 4.7, berdasarkan tabel tersebut *Cross-Attention* memiliki jumlah parameter yang tidak berbeda jauh dengan model lainnya bahkan lebih sedikit daripada *Factorised Encoder*.

4.2 Evaluasi Performa *Cross-Attention*

4.2.1 Studi Ablasi *Cross-Attention*

4.2.1.1 *Input Embedding*

Dalam studi ini, dilakukan eksperimen untuk mengevaluasi pengaruh ukuran *Temporal Embedding* terhadap performa model yang dapat dilihat pada Tabel 4.6. Eksperimen dilakukan menggunakan dua ukuran *patch* yang berukuran $2 \times (W/r) \times (H/r) \times 3$ (default) dan $1 \times (W/r) \times (H/r) \times 3$ yang akan menghasilkan jumlah *patch* (panjang *sequence*) yang berbeda. Semakin besar ukuran *patch*, maka panjang *sequence* yang akan dihasilkan semakin pendek karena *seq length* = *Input Size*/*Patch Size*. Sebaliknya, semakin kecil ukuran *patch*, panjang *sequence* menjadi lebih panjang dan detail visual yang ditangkap bisa lebih kaya.

Konfigurasi $\mathbf{z}_i^{(t)} = 1 \times (W/r) \times (H/r) \times 3$ yang berarti $n_d = D$ memberikan hasil yang lebih baik dibandingkan konfigurasi default $\mathbf{z}_i^{(t)} = 2 \times (W/r) \times (H/r) \times 3$ yang berarti $n_d = (D/2)$ pada kedua metrik evaluasi. Dengan menggunakan *patch* berukuran lebih kecil, model menerima input *sequence* yang lebih panjang dan lebih kaya akan informasi spasial membuat *Cross-Attention* memperhatikan hubungan temporal yang lebih detail .

Tabel 4.6: Efek dan perbandingan akurasi serta skor CIDEr model *Cross-Attention Video Vision Transformer* dengan Variasi Ukuran *Patch* pada *Temporal Embedding* (\mathbf{Z}_t) yang menghasilkan $n_d = (D/2)$ dan $n_d = D$

Patch Size ($\mathbf{z}_i^{(t)}$)	Sequence Accuracy	CIDEr Score	Params
$2 \times (W/r) \times (H/r) \times 3$	0.7740	1.6013	26.760 M
$1 \times (W/r) \times (H/r) \times 3$	0.7825	1.7464	26.762 M

Hasil ini menunjukkan bahwa meskipun menggunakan *patch* yang lebih kecil dapat meningkatkan panjang *sequence* dan memperbesar beban memori, dalam model *Cross-Attention*, strategi ini justru meningkatkan kualitas prediksi model. Ini menjadi

pertimbangan penting dalam perancangan arsitektur, terutama untuk tugas yang menekankan pemahaman detail visual atau hubungan semantik yang kompleks antara modalitas input.

4.2.1.2 Rasio Resize Linear 3D Convolutional

Selain ukuran *patch*, dilakukan eksperimen untuk mengevaluasi pengaruh rasio *resize r* terhadap performa model. Eksperimen ini bertujuan untuk mengamati bagaimana *trainable 3D Convolutional* melakukan proses *resize* terhadap dimensi spasial dan temporal. Semakin besar nilai rasio *resize r*, maka semakin besar ukuran *kernel* yang digunakan dalam konvolusi yaitu $k = (r//2) + 1$, yang mungkin memengaruhi representasi fitur yang dihasilkan.

Tabel 4.7 menunjukkan bahwa rasio *resize r* = 8 untuk dimensi spasial dan *r* = 4 untuk dimensi temporal memberikan hasil terbaik dalam hal akurasi, skor CIDEr, serta efisiensi jumlah parameter. Hal ini mengindikasikan bahwa pemilihan rasio *resize* perlu disesuaikan secara proporsional dengan ukuran input, mengingat resolusi spasial pada video umumnya jauh lebih besar dibandingkan resolusi temporal. Oleh karena itu, melakukan *resize* yang lebih kecil pada dimensi temporal dibandingkan dimensi spasial lebih efektif. Selain itu, konfigurasi dengan *r* = 4 untuk kedua dimensi menghasilkan performa yang lebih rendah, yang menunjukkan bahwa ukuran *kernel* konvolusi untuk *resize* memengaruhi kualitas hasil *resize*. Meskipun *kernel* yang lebih besar dapat menangkap konteks lebih luas, tetapi dapat menyebabkan hilangnya detail penting. Karena itu diperlukan keseimbangan antara skala *resize* spasial dan temporal untuk mencapai performa optimal.

Tabel 4.7: Efek dan perbandingan akurasi serta skor CIDEr model *Cross-Attention Video Vision Transformer* dengan rasio *resize r* yang berbeda.

Rasio Resize	Sequence Accuracy	CIDEr Score	Params
$r = 4$	0.7740	1.6013	26.760 M
$r = 8$	0.7670	1.5424	22.361 M
$r = 8(S), r = 4(T)$	0.7861	1.7845	22.245 M

4.2.1.3 3D Convolutional Projection Feature

Karena rasio *resize* terbukti memengaruhi performa model, hal ini memotivasi dilakukannya eksperimen lanjutan untuk mengevaluasi pengaruh jumlah filter pada proses *resize*. Eksperimen ini dilakukan dengan menggunakan jumlah filter yang berbeda, sama dengan konfigurasi pada Tabel 4.1, namun dengan perbedaan bahwa konvolusi pertama pada masing-masing dimensi menggunakan jumlah filter sebanyak 8 yang hasilnya dapat dilihat pada Tabel 4.8.

Jumlah filter yang lebih besar justru menghasilkan performa yang lebih rendah. Kemungkinan, fitur yang dihasilkan oleh proses *resize* menjadi lebih rinci sehingga tidak lagi mencerminkan ciri-ciri utama dari video secara keseluruhan. Hal ini dapat menyebabkan proses *attention* pada *Encoder Cross-Attention* tidak bekerja secara efektif

Tabel 4.8: Efek dan perbandingan akurasi serta skor CIDEr model *Cross-Attention Video Vision Transformer* dengan jumlah fitur 3D *Convolutional Projection* yang berbeda.

Filters Conv 1	Sequence Accuracy	CIDEr Score	Params
4	0.7740	1.6013	26.760 M
8	0.7388	0.9934	26.761 M

karena fitur yang digunakan kurang sesuai. Dan kemungkinan dari hasil ini dapat diketahui bahwa *Cross-Attention* memerlukan fitur yang lebih original dari data input.

4.2.1.4 Urutan *Query Cross-Attention*

Pada Gambar 3.6 didefinisikan bahwa Z_t dijadikan *query* terlebih dahulu dalam mekanisme *Cross Attention*. Namun, untuk memahami representasi mana yang lebih membutuhkan informasi asli dari representasi lainnya (dalam tingkat kedalaman jaringan yang setara), dilakukan eksperimen untuk membandingkan urutan pemrosesan antara embedding spasial (Z_s) dan embedding temporal (Z_t). Tabel 4.9 memperlihatkan hasil perbandingan performa dari kedua konfigurasi tersebut. Pada konfigurasi pertama, Z_t digunakan sebagai *query* terhadap Z_s pada tahap awal, kemudian diikuti dengan tahap sebaliknya. Sedangkan pada konfigurasi kedua, urutan tersebut dibalik, yaitu Z_s dijadikan *query* terhadap Z_t terlebih dahulu.

Hasil eksperimen menunjukkan bahwa konfigurasi dengan urutan $Z_s \rightarrow Z_t$, di mana *embedding* spasial dijadikan *query* terlebih dahulu terhadap *embedding* temporal, menghasilkan performa model yang lebih unggul dibandingkan konfigurasi sebaliknya. Perbedaan performa ini mengindikasikan bahwa menggunakan *embedding* spasial sebagai *query* terlebih dahulu membantu model dalam mengekstraksi informasi yang lebih relevan. Hal ini mungkin disebabkan oleh *embedding* spasial (Z_s) yang memiliki informasi lebih kuat dibanding *embedding* temporal (Z_t) karena Z_s mengandung fitur 3D dan menunjukkan bahwa *embedding* spasial lebih membutuhkan informasi dari *embedding* temporal yang belum termodifikasi (dalam tingkat kedalaman jaringan yang setara), sehingga lebih optimal bila *embedding* spasial dilakukan terlebih dahulu terhadap fitur asli melalui mekanisme *Cross-Attention* dibanding *embedding* temporal.

Tabel 4.9: Perbandingan performa model berdasarkan urutan *query* pada *Cross-Attention*.

Cross Attention	Sequence Accuracy	CIDEr Score	Params
$Z_t \rightarrow Z_s$	0.7740	1.6013	26.760 M
$Z_s \rightarrow Z_t$	0.7830	1.6605	26.760 M

4.2.2 Generalisasi Model

Untuk mengevaluasi kemampuan generalisasi model dalam menghasilkan *caption* yang sesuai dengan konten video, dilakukan analisis kualitatif terhadap hasil *caption* yang dihasilkan oleh model. Dalam proses ini, sampel data diambil secara acak dari set validasi dan digenerasi dengan konfigurasi model paling optimal yang terdapat pada

Hasil dan Pembahasan

Tabel 4.7 yaitu rasio $resize\ r = 8(S)$ dan $r = 4(T)$. Setiap *caption* hasil generasi kemudian dibandingkan dengan *ground truth* untuk menilai sejauh mana kesesuaian semantik dan sintaksis antara keduanya. Visualisasi perbandingan ini bertujuan untuk memberikan gambaran yang lebih konkret dan mendalam mengenai kemampuan model dalam memahami konteks serta mendeskripsikan isi video secara akurat dan bermakna.



Generate Caption: seekor anjing sedang mandi di bak mandi

Ground Truth 1: anjing bermain di nampakan air

Ground Truth 2: seekor anjing sedang mandi di bak mandi dengan air



Generate Caption: sloth ada di tempat sampah plastik

Ground Truth 1: sloth mencoba keluar dari kotak

Ground Truth 2: seekor binatang bermain di wadahnya

Gambar 4.2: Perbandingan *caption* hasil generate dengan beberapa *Ground Truth*



Generate Caption: seorang pria sekarat di tandu

Ground Truth 1: seorang dokter memberikan kompresi dada kepada seorang pria di tandu

Ground Truth 2: seorang pria sekarat di ruang operasi



Generate Caption: seorang pria bermain gitar akustik dan bernyanyi

Ground Truth 1: seorang pria bernyanyi saat bermain gitar

Ground Truth 2: pria itu bernyanyi dan bermain gitar



Generate Caption: wanita itu mengiris bawang dengan pisau

Ground Truth 1: seorang wanita mengiris bawang dengan pisau

Ground Truth 2: seorang wanita mengiris bawang dengan pisau besar

Gambar 4.3: Perbandingan *caption* hasil generate dengan beberapa *Ground Truth*

Hasil dan Pembahasan



Generate Caption: anak-anak melakukan trik membalik pada trampolin
Ground Truth 1: dua orang menunjukkan beberapa trik trampolin
Ground Truth 2: orang melakukan flips dan trik pada trampolin



Generate Caption: dua pria saling tinju dalam pertarungan tinju
Ground Truth 1: dua pria bertinju di atas ring
Ground Truth 2: dua pria melakukan pertandingan tinju

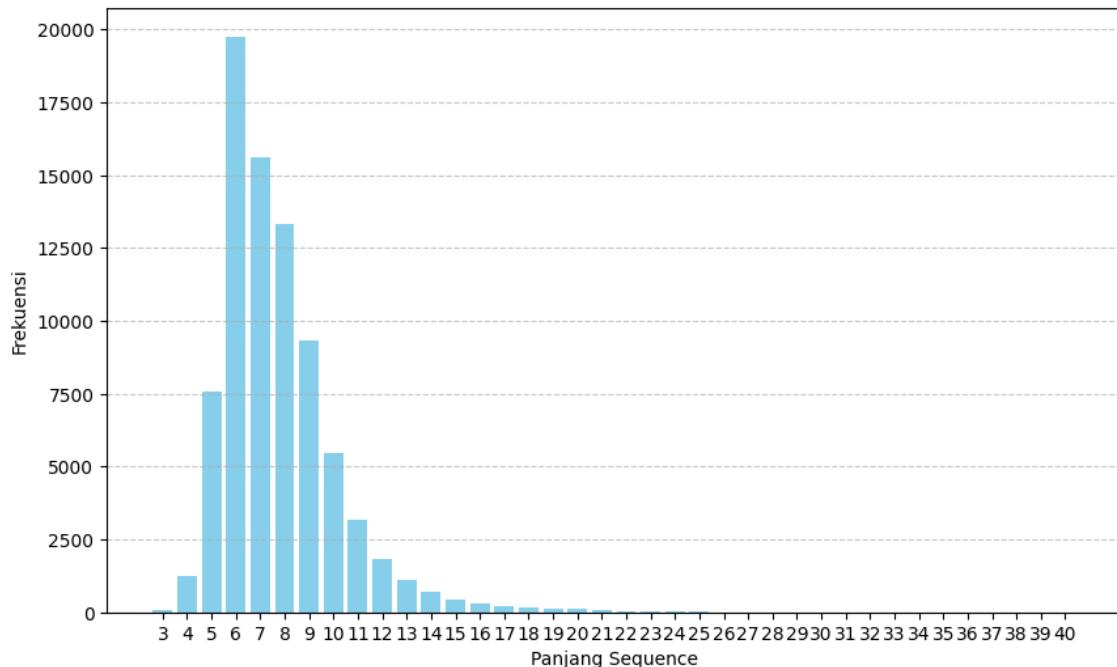


Generate Caption: dua pria sedang berjalan di klub malam
Ground Truth 1: dua pria berjalan melalui klub
Ground Truth 2: dua pria kulit hitam sedang berjalan melalui bar

Gambar 4.4: Perbandingan *caption* hasil generate dengan beberapa *Ground Truth*

Gambar 4.2 memperlihatkan perbandingan antara *caption* hasil generasi model dan *Ground Truth* yang tersedia. Pada bagian pertama dari gambar tersebut, terlihat bahwa *caption* hasil generasi menunjukkan tingkat kesesuaian yang tinggi, baik dari segi makna semantik maupun dalam struktur atau urutan katanya. Hal ini menghasilkan *caption* yang memiliki kemiripan erat dengan *Ground Truth*, mencerminkan bahwa model mampu memahami dan merepresentasikan isi video secara akurat. Sebaliknya, pada bagian kedua gambar, *caption* yang dihasilkan tetap relevan secara semantik, namun terdapat perbedaan dalam susunan kata dibandingkan dengan *Ground Truth* yang ditampilkan. Meskipun demikian, perlu dicatat bahwa *caption* tersebut kemungkinan memiliki kemiripan dengan referensi *Ground Truth* lainnya dalam yang tidak ditampilkan pada visualisasi gambar ini.

Secara umum, Gambar 4.3 dan Gambar 4.4 juga menunjukkan hasil evaluasi yang konsisten dari segi relevansi semantik antara *caption* yang dihasilkan dan konten video yang bersangkutan. Walaupun terdapat beberapa contoh di mana susunan kata pada *caption* berbeda dengan *Ground Truth*, tetap ditemukan kasus-kasus di mana kesesuaian urutan kata sangat tinggi, bahkan sama, seperti yang terlihat pada bagian ketiga dari Gambar 4.3. Berdasarkan tingkat kesesuaian semantik ini, dapat disimpulkan bahwa kualitas *caption* yang dihasilkan oleh model tergolong cukup baik dalam menangkap dan mengekspresikan makna visual dari video. Meskipun tidak selalu konsisten dalam hal urutan kata, aspek tersebut tidak terlalu krusial dalam konteks evaluasi semantik, selama informasi inti tetap tersampaikan secara akurat dan relevan.



Gambar 4.5: Distribusi panjang *sequence* pada dataset



Generate Caption: seorang wanita menambahkan kecap dan gula ke dalam mangkuk yang mengandung daging yang diiris
Ground Truth 1: seorang wanita menambahkan beberapa bahan ke dalam mangkuk
Ground Truth 2: wanita menambahkan gula ke daging

Gambar 4.6: Caption hasil generasi dengan panjang mencapai sekitar 14 kata

Berdasarkan Gambar 4.2, Gambar 4.3, dan Gambar 4.4 , panjang caption hasil generasi umumnya tidak berbeda jauh dengan Ground Truth. Gambar 4.5 menunjukkan bahwa panjang caption dalam dataset paling sering berada dalam rentang 5 hingga 12 kata. Namun, ditemukan caption hasil generasi dengan panjang sekitar 14 kata yang ditampilkan pada Gambar 4.6. Meskipun panjang 14 tidak termasuk yang paling umum dalam dataset, proporsinya tetap cukup signifikan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Studi ini mengevaluasi kinerja dan kompleksitas model *Video Vision Transformer* berbasis *Cross-Attention* untuk tugas deskripsi video, dengan membandingkannya terhadap empat pendekatan *Video Vision Transformer* yang telah ada. Hasil eksperimen menunjukkan bahwa model yang diusulkan mampu mencapai performa yang kompetitif, khususnya dibandingkan dengan pendekatan unggulan seperti *Spatio-Temporal Attention* dan *Factorised Dot Product Attention*. Dalam konfigurasi standar, performanya hampir menyamai kedua pendekatan tersebut dengan efisiensi komputasi yang lebih baik tetapi parameter yang lebih banyak, bahkan dengan konfigurasi paling optimal, model ini mampu menghasilkan parameter yang lebih sedikit.

Karena kualitas caption hasil generasi model yang baik, yang menunjukkan bahwa mekanisme *Cross-Attention* mampu menangkap dan menyelaraskan fitur-fitur penting dalam dimensi temporal dan dimensi spasial dengan representasi bahasa yang sesuai. Hasilnya, model ini dapat menghasilkan deskripsi video dengan baik dan sesuai dengan konteks.

5.2 Saran

Penelitian mengenai *Cross-Attention* dapat dikembangkan lebih lanjut dengan menggunakan dataset tambahan seperti MPII-MD, MVAD, dan YouCook2. Evaluasi lanjutan dapat dilakukan dengan membandingkan model ini terhadap model-model *state-of-the-art* lainnya, atau terhadap varian lain dari *Video Vision Transformer* yang digunakan untuk tugas klasifikasi video seperti dalam makalah aslinya. Selain itu, penelitian mendalam juga dapat difokuskan pada penyempurnaan proses pembentukan *embedding* agar tetap efisien dan tidak terlalu kompleks, terlepas dari ukuran atau durasi video yang digunakan.

DAFTAR PUSTAKA

- [1] C. M. Annur, *Individu pengguna internet global tembus 5,35 miliar pada januari 2024*, <https://databoks.katadata.co.id/datapublish/2024/02/08/individu-pengguna-internet-global-tembus-535-miliar-pada-januari-2024>, Diakses pada: 13-September-2024, 2024.
- [2] S. Kemp, *Internet use in 2024*, <https://datareportal.com/reports/digital-2024-deep-dive-the-state-of-internet-adoption>, Diakses pada: 13-September-2024, 2024.
- [3] Sandvine, *2024 global internet phenomena report*, <https://www.sandvine.com/phenomena>, Diakses pada: 13-September-2024, 2024.
- [4] K. Funk, *Accessibility in digital media for the visually impaired*, <https://www.accessibility.com/blog/accessibility-in-digital-media-for-the-visually-impaired>, Diakses pada: 13-September-2024, 2022.
- [5] M. Abdar *et al.*, *A review of deep learning for video captioning*, 2023. arXiv: 2304 . 11431 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2304.11431>.
- [6] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, *Show and tell: A neural image caption generator*, 2015. arXiv: 1411 . 4555 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1411.4555>.
- [7] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles, *Dense-captioning events in videos*, 2017. arXiv: 1705 . 00754 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1705.00754>.
- [8] A. M. Nugroho and A. F. Hidayatullah, “Keterangan gambar otomatis berbahasa indonesia dengan cnn dan lstm”, *jurnal.uii.ac.id*, [Online]. Available: <https://journal.uii.ac.id/AUTOMATA/article/download/17389/10928/45684>.
- [9] R. D. Z. PUTRA, Azhari, and D. U. K. Putri, “Pembangkitan deskripsi gambar dalam bahasa indonesia menggunakan pendekatan berbasis transformer”, *etd.repository.ugm.ac.id*, 2023. [Online]. Available: <https://etd.repository.ugm.ac.id/penelitian/detail/219285>.
- [10] M. R. S. MAHADI, A. ARIFIANTO, and K. N. RAMADHANI, “Membangkitkan deskripsi gambar dalam bahasa indonesia menggunakan adaptive attention”, *repository.telkomuniversity.ac.id*, 2020. [Online]. Available: <https://repository.telkomuniversity.ac.id/pustaka/159403/membangkitkan-deskripsi-gambar-dalam-bahasa-indonesia-menggunakan-adaptive-attention.html>.
- [11] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, *Sequence to sequence – video to text*, 2015. arXiv: 1505 . 00487 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1505.00487>.

- [12] Y. Pan, T. Yao, H. Li, and T. Mei, *Video captioning with transferred semantic attributes*, 2016. arXiv: 1611 . 07675 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1611.07675>.
- [13] A. Vaswani *et al.*, *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL]. [Online]. Available: <https://arxiv.org/abs/1706.03762>.
- [14] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, *Videobert: A joint model for video and language representation learning*, 2019. arXiv: 1904.01766 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1904.01766>.
- [15] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. LuÄiÄ, and C. Schmid, *Vivit: A video vision transformer*, 2021. arXiv: 2103 . 15691 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2103.15691>.
- [16] P. H. Seo, A. Nagrani, A. Arnab, and C. Schmid, *End-to-end generative pretraining for multimodal video captioning*, 2022. arXiv: 2201.08264 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2201.08264>.
- [17] D. Chen and W. Dolan, “Collecting highly parallel data for paraphrase evaluation”, in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, D. Lin, Y. Matsumoto, and R. Mihalcea, Eds., Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 190–200. [Online]. Available: <https://aclanthology.org/P11-1020>.
- [18] A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele, *A dataset for movie description*, 2015. arXiv: 1501 . 02530 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1501.02530>.
- [19] A. Torabi, C. Pal, H. Larochelle, and A. Courville, *Using descriptive video services to create a large data source for video annotation research*, 2015. arXiv: 1503 . 01070 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1503.01070>.
- [20] L. Zhou, C. Xu, and J. J. Corso, *Towards automatic learning of procedures from web instructional videos*, 2017. arXiv: 1703 . 09788 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1703.09788>.
- [21] O. Russakovsky *et al.*, *Imagenet large scale visual recognition challenge*, 2015. arXiv: 1409 . 0575 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1409.0575>.
- [22] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, *Imagenet-21k pretraining for the masses*, 2021. arXiv: 2104 . 10972 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2104.10972>.
- [23] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, *Revisiting unreasonable effectiveness of data in deep learning era*, 2017. arXiv: 1707 . 02968 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1707.02968>.
- [24] W. Kay *et al.*, *The kinetics human action video dataset*, 2017. arXiv: 1705 . 06950 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1705.06950>.

- [25] D. Damen *et al.*, *Scaling egocentric vision: The epic-kitchens dataset*, 2018. arXiv: 1804.02748 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1804.02748>.
- [26] M. Monfort *et al.*, *Moments in time dataset: One million videos for event understanding*, 2019. arXiv: 1801.03150 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1801.03150>.
- [27] R. Goyal *et al.*, *The "something something" video database for learning and evaluating visual common sense*, 2017. arXiv: 1706.04261 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1706.04261>.
- [28] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, *Learning spatiotemporal features with 3d convolutional networks*, 2015. arXiv: 1412.0767 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1412.0767>.
- [29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019. arXiv: 1810.04805 [cs.CL]. [Online]. Available: <https://arxiv.org/abs/1810.04805>.
- [30] D. Zhang, X. Dai, X. Wang, and Y.-F. Wang, *S3d: Single shot multi-span detector via fully 3d convolutional networks*, 2018. arXiv: 1807.08069 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1807.08069>.
- [31] A. Dosovitskiy *et al.*, *An image is worth 16x16 words: Transformers for image recognition at scale*, 2021. arXiv: 2010.11929 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2010.11929>.
- [32] A. Singh, T. D. Singh, and S. Bandyopadhyay, *Nits-vc system for vatex video captioning challenge 2020*, 2020. arXiv: 2006.04058 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2006.04058>.
- [33] J. Yang, *Rethinking tokenization: Crafting better tokenizers for large language models*, 2024. arXiv: 2403.00417 [cs.CL]. [Online]. Available: <https://arxiv.org/abs/2403.00417>.
- [34] F. Almeida and G. XexÃ©o, “Word embeddings: A survey”, *arXiv preprint arXiv:1901.09069*, Jan. 2019. [Online]. Available: <https://arxiv.org/abs/1901.09069>.
- [35] TensorFlow, *Word embeddings*, https://www.tensorflow.org/text/guide/word_embeddings, Accessed: 2025-05-27, n.d.
- [36] A. W. Service, *What is a transformer in artificial intelligence?*, <https://aws.amazon.com/id/what-is/transformers-in-artificial-intelligence/>, Diakses pada: 23-September-2024.
- [37] R. Xiong *et al.*, *On layer normalization in the transformer architecture*, 2020. arXiv: 2002.04745 [cs.LG]. [Online]. Available: <https://arxiv.org/abs/2002.04745>.
- [38] Suyanto, *Machine Learning Tingkat Dasar dan Lanjut*, 1st ed. Bandung: Penerbit INFORMATIKA, 2022.

- [39] K. OâShea and R. Nash, “An introduction to convolutional neural networks”, *arXiv*, vol. 1511.08458, 2015. [Online]. Available: <https://arxiv.org/abs/1511.08458>.
- [40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: <http://www.deeplearningbook.org>.
- [41] Tensorflow, *Video classification with a 3d convolutional neural network*, Diakses pada: 22-September-2024, 2024. [Online]. Available: https://www.tensorflow.org/tutorials/video/video_classification.
- [42] R. Vedantam, C. L. Zitnick, and D. Parikh, *Cider: Consensus-based image description evaluation*, 2015. arXiv: 1411 . 5726 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1411.5726>.
- [43] W. F. Hendria, “MSVD-Indonesian: A benchmark for multimodal video-text tasks in indonesian”, *arXiv preprint arXiv:2306.11341*, Jun. 2023. [Online]. Available: <https://arxiv.org/abs/2306.11341>.
- [44] Mux, *Mux video glossary: Frame*, <https://www.mux.com/video-glossary/frame>, Diakses pada: 30-September-2024.
- [45] TensorFlow, *Tf.keras.losses.sparsecategoricalcrossentropy*, https://www.tensorflow.org/api_docs/python/tf/keras/losses/_SparseCategoricalCrossentropy, Diakses pada: 20-Oktober-2024, 2024.
- [46] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412 . 6980 [cs.LG]. [Online]. Available: <https://arxiv.org/abs/1412.6980>.
- [47] H. Talebi and P. Milanfar, *Learning to resize images for computer vision tasks*, 2021. arXiv: 2103.09950 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2103.09950>.
- [48] D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, 2023. arXiv: 1606 . 08415 [cs.LG]. [Online]. Available: <https://arxiv.org/abs/1606.08415>.
- [49] H. Fang, J.-U. Lee, N. S. Moosavi, and I. Gurevych, *Transformers with learnable activation functions*, 2023. arXiv: 2208.14111 [cs.CL]. [Online]. Available: <https://arxiv.org/abs/2208.14111>.
- [50] Y. E. Wang, G.-Y. Wei, and D. Brooks, *Benchmarking tpu, gpu, and cpu platforms for deep learning*, 2019. arXiv: 1907 . 10701 [cs.LG]. [Online]. Available: <https://arxiv.org/abs/1907.10701>.
- [51] G. Menghani, “Efficient deep learning: A survey on making deep learning models smaller, faster, and better”, *ACM Computing Surveys*, vol. 55, no. 12, 1â37, Mar. 2023, ISSN: 1557-7341. DOI: 10 . 1145 / 3578938. [Online]. Available: <http://dx.doi.org/10.1145/3578938>.