

**PEMBANGKITAN *CAPTION* BERKONTEKS  
DARI GAMBAR**

**Laporan Tugas Akhir**

**Disusun sebagai syarat kelulusan tingkat sarjana**

**Oleh**

**JESSIN DONNYSON**

**NIM : 13516112**



**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO & INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
JUNI 2020**

**PEMBANGKITAN *CAPTION* BERKONTEKS  
DARI GAMBAR**

**Laporan Tugas Akhir**

**Oleh**

**JESSIN DONNYSON**

**NIM : 13516112**

**Program Studi Teknik Informatika**

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Telah disetujui dan disahkan sebagai Laporan Tugas Akhir  
di Bandung, pada tanggal 5 Juni 2020

Pembimbing,



Dr. Masayu Leylia Khodra, ST., MT.

NIP. 197604292008122001

## **LEMBAR PERNYATAAN**

Dengan ini saya menyatakan bahwa:

1. Pengerjaan dan penulisan Laporan Tugas Akhir ini dilakukan tanpa menggunakan bantuan yang tidak dibenarkan.
2. Segala bentuk kutipan dan acuan terhadap tulisan orang lain yang digunakan di dalam penyusunan Laporan Tugas Akhir ini telah dituliskan dengan baik dan benar.
3. Laporan Tugas Akhir ini belum pernah diajukan pada program pendidikan di perguruan tinggi mana pun. Jika terbukti melanggar hal-hal di atas, saya bersedia dikenakan sanksi sesuai dengan Peraturan Akademik dan Kemahasiswaan Institut Teknologi Bandung bagian Penegakan Norma Akademik dan Kemahasiswaan khususnya Pasal 2.1 dan Pasal 2.2.

Bandung, 20 Juni 2020

Jessin Donnyson

NIM 13516112

# ABSTRAK

## PEMBANGKITAN CAPTION BERKONTEKS DARI GAMBAR

Oleh

JESSIN DONNYSON

NIM: 13516112

*Image captioning* merupakan *task* untuk membangkitkan teks (*caption*) dari gambar. Beberapa tahun terakhir *image captioning* mendapat banyak perhatian baik dari industri dan akademisi, namun saat ini masih belum banyak digunakan karena masih adanya keterbatasan dan ketidaksesuaian antara *caption* yang dibangkitkan dengan keperluan yang ada. Salah satunya karena kurangnya konteks dari *caption* yang dibangkitkan. Pada Tugas akhir ini, dikembangkan dua pendekatan agar model *image captioning* dapat menambahkan konteks pada *caption* yang dibangkitkan tanpa perlu melatih kembali model atau membuat *dataset* berkonteks yang baru.

Pendekatan pertama memanfaatkan model atribut untuk mengubah distribusi kata yang seharusnya dibangkitkan. Pergeseran dilakukan dengan menambahkan vektor gradien yang didapatkan dari perhitungan *loss* model atribut. *Loss* dapat berupa perbedaan distribusi kata (model atribut *Bag of Words*) maupun perbedaan kelas (model atribut diskriminator). Pendekatan kedua dilakukan dengan mengganti komponen *word embedding* menjadi *contextual word embedding* yang didapatkan dari model bahasa BERT dan GPT-2. Model *image captioning* yang digunakan meliputi model berbasis rekuren (LSTM) dan *Transformer*. BLEU digunakan sebagai metrik penilaian utama.

Hasil eksperimen menunjukkan peningkatan nilai BLEU pada model yang menggunakan *contextual word embedding* GPT-2. Model *Transformer-GPT2* memperoleh kinerja terbaik dengan nilai BLEU 0.613 pada *dataset* MSCOCO. Penambahan konteks dengan model atribut menunjukkan hasil yang positif walaupun penggunaannya masih terbatas pada jenis konteks yang ditambahkan.

Kata kunci: *Image captioning*, konteks, model atribut, *contextual word embedding*.

## KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena dengan kasih karunia dan rahmat-Nya penulis dapat menyelesaikan Laporan Tugas Akhir dengan judul “Pembangkitan *Caption* Berkonteks dari Gambar” dengan baik. Kelancaran penulisan Laporan Tugas Akhir ini tidak lepas dari bantuan berbagai pihak. Oleh sebab itu, pada kesempatan ini penulis dengan segala kerendahan hati dan penuh rasa hormat mengucapkan terima kasih kepada:

1. Ibu Dr. Masayu Leylia Khodra, S.T., M.T. selaku dosen pembimbing yang memberikan arahan, bimbingan, dan nasihat selama pengerjaan Tugas Akhir.
2. Seluruh dosen pengajar di Teknik Informatika ITB yang telah memberikan ilmu dan wawasan yang dipakai untuk pengerjaan Tugas Akhir.
3. Rekan-rekan jurusan Informatika ITB yang telah memberikan dukungan kepada penulis untuk menyelesaikan Tugas Akhir.
4. Orang tua, keluarga, serta saudara penulis yang telah memberikan dukungan baik dari segi materi maupun dorongan moral dari awal pengerjaan Tugas Akhir hingga akhir penulisan Laporan Tugas Akhir.
5. Seluruh pihak lain yang membantu penulis baik secara langsung maupun tidak langsung yang tidak dapat disebutkan satu per satu.

Akhir kata, semoga Laporan Tugas Akhir yang telah penulis susun bermanfaat untuk penelitian-penelitian selanjutnya.

Bandung, 20 Juni 2020

Penulis

## DAFTAR ISI

LEMBAR PERNYATAAN .....	iii
ABSTRAK .....	iv
KATA PENGANTAR.....	v
DAFTAR ISI .....	vi
DAFTAR GAMBAR .....	x
DAFTAR TABEL .....	xii
DAFTAR SINGKATAN.....	xiii
BAB I    PENDAHULUAN.....	1
I.1    Latar Belakang .....	1
I.2    Rumusan Masalah .....	3
I.3    Tujuan.....	4
I.4    Batasan Masalah.....	4
I.5    Metodologi .....	4
I.6    Sistematika Pembahasan .....	5
BAB II   STUDI LITERATUR.....	7
II.1 <i>Image Captioning</i> .....	7
II.1.1   Topologi Model Seq2Seq .....	8
II.1.2   Topologi Model <i>Transformer</i> .....	11
II.1.3   Kerangka Kerja <i>Image Captioning</i> .....	14
II.2 <i>Contextual Word Embedding</i> .....	16
II.2.1   OpenAI GPT.....	17
II.2.2   BERT.....	18

II.3	Evaluasi Kinerja .....	20
II.3.1	BLEU.....	21
II.3.2	ROUGE .....	23
II.3.3	METEOR.....	24
II.4	Penelitian Terkait .....	25
II.4.1	Penelitian Dathathri dkk. (2019) .....	25
II.4.2	Penelitian Keskar dkk. (2019) .....	27
II.4.3	Penelitian Sharma dkk. (2018) .....	28
BAB III	DESKRIPSI SOLUSI .....	30
III.1	Analisis Permasalahan.....	30
III.1.1	Analisis <i>Image Captioning</i> .....	30
III.1.2	Analisis Konteks.....	32
III.2	Alternatif Solusi .....	34
III.2.1	Alternatif Solusi <i>Vector Injection</i> .....	34
III.2.2	Alternatif Solusi Menggunakan <i>Contextualized Word Embedding</i> .....	40
III.3	Pemilihan Alternatif .....	41
III.3.1	Pemilihan Solusi.....	41
III.3.2	Pemilihan Alternatif <i>Contextualized Word Embedding</i> .....	42
III.3.3	Pemilihan Alternatif <i>Inject / Merge</i> (model sekuensial) .....	44
III.3.4	Pemilihan Alternatif Komponen <i>Image Encoder</i> .....	44
III.4	Rancangan Solusi .....	45
III.4.1	Skema Praproses <i>Dataset</i> .....	45
III.4.2	Skema Pelatihan Model.....	46
III.4.3	Skema Pembangkitan <i>Caption</i> .....	48
III.4.4	Skema Persiapan Model Atribut.....	50

III.4.5	Skema Pembangkitan <i>Caption</i> dengan Konteks .....	51
III.4.6	Detail Arsitektur Model.....	53
III.5	Rancangan Penelitian .....	54
III.5.1	<i>Dataset</i> .....	54
III.5.2	Metrik Penilaian .....	55
III.5.3	Implementasi Ulang Model <i>Image Captioning</i> .....	55
III.5.4	Aplikasi Penambahan Konteks pada <i>Caption</i> Gambar .....	56
BAB IV	IMPLEMENTASI.....	58
IV.1	Lingkungan Implementasi .....	58
IV.2	Implementasi .....	58
IV.2.1	Praproses Gambar (tensorflow.keras) .....	60
IV.2.2	Praproses Teks.....	60
IV.2.3	Pembagian Data (sklearn) .....	65
IV.2.4	Model (tensorflow).....	65
IV.2.5	Strategi Pemilihan Kata.....	70
IV.2.6	Pembangkitan <i>Caption</i> .....	73
IV.2.7	Penambahan Konteks .....	74
IV.2.8	Praproses Model Atribut ( <i>Bag of Words</i> ).....	77
IV.2.9	Evaluasi (nlgeval).....	79
BAB V	EKSPERIMEN DAN PENGUJIAN .....	80
V.1	Eksperimen.....	80
V.1.1	Data Eksperimen .....	80
V.1.2	Eksperimen <i>Contextual Word Embedding</i> .....	81
V.1.3	Eksperimen Penambahan Konteks .....	85
V.2	Pengujian .....	92



V.2.1	Data Pengujian .....	92
V.2.2	Pengujian Kinerja <i>Contextual Word Embedding</i> .....	92
V.2.3	Pengujian Kualitas <i>Caption</i> Berkonteks.....	94
BAB VI	KESIMPULAN DAN SARAN.....	99
VI.1	Kesimpulan.....	99
VI.2	Saran.....	99
DAFTAR PUSTAKA.....		xiv
LAMPIRAN .....		xviii
Lampiran A. Contoh Penilaian Pengujian 2 .....		xviii

## DAFTAR GAMBAR

Gambar I.1 Maldives (Centara Grand).....	2
Gambar II.1 Contoh <i>Image Captioning</i> (Karpathy dkk., 2015) .....	7
Gambar II.2 Topologi <i>Sequence-to-Sequence</i> Model (Abeywardana, 2017) .....	8
Gambar II.3 CNN <i>Encoder</i> (Inception) - RNN <i>Decoder</i> (LSTM) (Vinyals dkk., 2014).....	10
Gambar II.4 <i>Transformer</i> (Vaswani dkk., 2017) yang dilengkapi nomor formula yang digunakan.....	12
Gambar II.5 Ilustrasi Umum <i>Transformer</i> (Alammar dkk., 2018) .....	13
Gambar II.6 Kerangka Kerja <i>Image Captioning</i> (Xu dkk., 2016) .....	15
Gambar II.7 Proses <i>Image Captioning</i> (Tensorflow.org, 2020).....	15
Gambar II.8 Arsitektur Model <i>Transformer Encoder</i> (Vaswani dkk., 2017) .....	18
Gambar II.9 Representasi Masukan BERT (Devlin dkk., 2019).....	19
Gambar II.10 Ilustrasi PPLM (Dathathri dkk., 2019) .....	26
Gambar II.11 Perbandingan Hasil <i>Caption</i> (Sharma dkk., 2018) .....	29
Gambar III.1 Sampel Gambar (Mikyoui00, Centara Grand) .....	31
Gambar III.2 Contoh <i>Dataset</i> MSCOCO .....	32
Gambar III.3 Ilustrasi Penggunaan Vektor Informasi .....	36
Gambar III.4 Ilustrasi Penggunaan Vektor Gradien.....	38
Gambar III.5 Ilustrasi Permasalahan BERT saat Pembangkitan <i>Caption</i> .....	43
Gambar III.6 Tabel <i>Benchmark</i> Model (Keras) .....	45
Gambar III.7 Gambar Skema Pemrosesan <i>Dataset</i> .....	46
Gambar III.8 Gambar Skema Pelatihan Model .....	47
Gambar III.9 Gambar Skema Pembangkitan <i>Caption</i> .....	49

Gambar III.10 Gambar Skema Persiapan Model Atribut.....	51
Gambar III.11 Gambar Skema Proses Pengubahan Distribusi Kata .....	52
Gambar III.12 Gambar Skema Pembangkitan <i>Caption</i> Berkonteks .....	52
Gambar III.13 Diagram Arsitektur Berbasis Model Rekuren .....	53
Gambar III.14 Diagram Arsitektur Model Berbasis <i>Transformer</i> .....	54
Gambar IV.1 Diagram Implementasi .....	59

## DAFTAR TABEL

<b>Tabel III.1 Contoh Nilai Vektor pada Gambar III.3 .....</b>	<b>36</b>
<b>Tabel III.2 Contoh Nilai Vektor Distribusi pada Gambar III.3 .....</b>	<b>37</b>
<b>Tabel III.3 Contoh Nilai Vektor pada Gambar III.4 .....</b>	<b>38</b>
<b>Tabel III.4 Contoh Nilai Distribusi pada Gambar III.4 .....</b>	<b>38</b>
<b>Tabel III.5 Contoh Nilai Keluaran Model Atribut pada Gambar III.4 .....</b>	<b>38</b>
<b>Tabel III.6 Detail Penilaian .....</b>	<b>57</b>
<b>Tabel IV.1 Lingkungan Implementasi.....</b>	<b>58</b>
<b>Tabel IV.2 Statistik Kosakata .....</b>	<b>64</b>
<b>Tabel IV.3 Statistik Panjang <i>Caption</i> .....</b>	<b>64</b>
<b>Tabel V.1 Statistik Pembagian Data.....</b>	<b>80</b>
<b>Tabel V.2 Detail Kombinasi Parameter .....</b>	<b>81</b>
<b>Tabel V.3 Detail Kombinasi Parameter .....</b>	<b>86</b>
<b>Tabel V.4 Hasil Penambahan Konteks.....</b>	<b>88</b>
<b>Tabel V.5 Dampak Perubahan Parameter.....</b>	<b>91</b>
<b>Tabel V.6 Hasil Pengujian Model Terbaik pada Eksperimen I.....</b>	<b>93</b>
<b>Tabel V.7 Hasil pengujian II (1).....</b>	<b>95</b>
<b>Tabel V.8 Hasil pengujian II (2).....</b>	<b>96</b>

## DAFTAR SINGKATAN

BERT	: <i>Bidirectional Encoder Representations from Transformers</i>
BLEU	: <i>Bilingual Evaluation Understudy</i>
CNN	: <i>Convolutional Neural Network</i>
GPT	: <i>Generative Pre-training Transformer</i>
LSTM	: <i>Long Short-Term Memory</i>
METEOR	: <i>Metric for Evaluation for Translation with Explicit Ordering</i>
NLP	: <i>Natural Language Processing</i>
RNN	: <i>Recurrent Neural Network</i>
ROUGE	: <i>Recall Oriented Understudy for Gisting Evaluation</i>
Seq2Seq	: <i>Sequence to Sequence</i>
Word2Vec	: <i>Word to Vector</i>

# BAB I

## PENDAHULUAN

### I.1 Latar Belakang

*Image captioning* merupakan *task* untuk membangkitkan teks (*caption*) dari gambar. *Image captioning* memanfaatkan dua bidang dalam intelegensi buatan, yaitu *computer vision* dan *natural language generation*. Beberapa tahun terakhir, *image captioning* mendapat banyak perhatian baik dari industri maupun akademisi. Berbagai teknik dan arsitektur model bermunculan setiap tahunnya. Model *image captioning* yang dikembangkan Xu dkk. (2015) berhasil menjadi *state-of-the-art* pada tiga *dataset* publik yang umum digunakan sebagai *benchmark* yaitu Flickr8k (BLEU-1: 0.67), Flickr30k (BLEU-1: 0.669) dan MSCOCO (BLEU-1: 0.718).

*Image captioning* sebenarnya memiliki potensi besar dalam bidang industri. Salah satu contoh kasus penggunaannya adalah untuk membangkitkan cerita dari kumpulan gambar dan ilustrasi. Model *image captioning* juga dapat digunakan untuk membangkitkan deskripsi produk secara otomatis untuk *e-commerce* dan ulasan untuk lokasi wisata, restoran, hotel, dan masih banyak lagi. Namun kenyataannya *image captioning* saat ini masih belum banyak digunakan dalam industri. Hal ini disebabkan karena masih adanya keterbatasan dan ketidaksesuaian antara *caption* yang dibangkitkan dengan keperluan industri. Sebagai contoh, industri makanan lebih memerlukan *caption* yang menekankan kelezatan makanan (misalnya kata ‘enak’, ‘renyah’, ‘segar’); sedangkan untuk industri perhotelan lebih menekankan kenyamanan kamar (misalnya ‘bersih’, ‘harum’, ‘murah’).

Sayangnya *caption* yang dibangkitkan seringkali hanya menyebutkan objek-objek yang terlihat pada gambar. Permasalahan kurangnya konteks dari *caption* yang dibangkitkan dapat terjadi karena model yang digunakan belum dapat menangkap konteks (makna implisit) dari gambar. Misalkan pada gambar I.1, *caption* yang dihasilkan adalah “*bedroom with single bed*”, padahal sebenarnya gambar tersebut menceritakan salah satu kamar *resort* ternama di Maldives yang sangat terkenal keindahan lautnya.



Gambar I.1 Maldives (Centara Grand)

Terdapat penelitian serupa yang dilakukan Sharma dkk. (2018) untuk menghasilkan model *image captioning* berkonteks dengan cara membuat sebuah *dataset* yang telah dilengkapi konteks. Cara ini intuitif namun memerlukan sumber daya yang besar. Berbeda dari Sharma dkk., tugas akhir ini difokuskan untuk mengembangkan model *image captioning* yang dapat menambahkan konteks pada *caption* yang dibangkitkan tanpa perlu membuat *dataset* berkonteks yang baru. Tujuannya agar *caption* menjadi lebih bermakna dan sesuai dengan kebutuhan industri. Permasalahan kurangnya konteks pada *caption* yang dibangkitkan dapat diselesaikan dengan dua pendekatan.

Pendekatan pertama dilakukan dengan cara menambahkan sumber informasi lain sebagai tambahan konteks saat melakukan pembangkitan *caption*. Park dkk. (2017) dalam penelitiannya — untuk menghasilkan *personalized caption* — telah membuktikan bahwa dengan menggunakan tambahan informasi berupa *caption* yang pernah dibuat seseorang sebelumnya (dalam penelitian ini tambahan konteks berupa gaya bahasa), model dapat menghasilkan *caption* yang lebih *personalized* untuk setiap orang.

Dalam domain terkait pembangkitan teks, ide penambahan informasi untuk mengubah hasil dari pembangkitan juga pernah diteliti oleh Dathathri dkk. (2019)

dan Keskar dkk. (2019). Dalam penelitiannya, diusulkan metode untuk mengubah hasil pembangkitan model bahasa tanpa perlu melakukan pelatihan ulang.

Selain menggunakan sumber informasi tambahan, untuk mengatasi permasalahan kurangnya konteks, dapat dilakukan dengan cara memanfaatkan *contextual word embedding* saat melakukan pembangkitan *caption*. Salah satu model *contextual word embedding* yang dapat digunakan adalah BERT. Devlin dkk. (2019) mengembangkan suatu model bahasa (*language model*) yang diberi nama BERT (*Bidirectional Encoder Representations from Transformers*). Dalam penelitiannya, Devlin dkk. mengklaim BERT mengalahkan hampir semua *state-of-the-art* sebelumnya di berbagai *task natural language processing* dengan mudah. Menurut hipotesis penulis, penggunaan *contextual word embedding* dapat meningkatkan kualitas dan memberikan tambahan konteks pada *caption* yang dibangkitkan. Hal ini dapat terjadi karena *contextual word embedding* memiliki kemampuan untuk membedakan vektor kata berdasarkan konteks penggunaan kata dalam suatu kalimat. Kata yang sama dapat memiliki vektor berbeda tergantung konteks kalimat.

## **I.2 Rumusan Masalah**

Seperti disebutkan pada sub bab sebelumnya, *image captioning* memiliki potensi besar dalam industri, namun penggunaannya masih terbatas dikarenakan adanya ketidaksesuaian hasil *caption* dengan kebutuhan industri. Permasalahan utama yang diselesaikan adalah bagaimana menghasilkan *caption* yang lebih bermakna dengan memanfaatkan konteks dari sumber informasi tambahan dan *contextual word embedding* (misalnya BERT).

Pertanyaan yang dijawab melalui pelaksanaan tugas akhir ini adalah sebagai berikut:

1. Bagaimana membangkitkan *caption* yang memiliki konteks dengan menggunakan tambahan informasi?
2. Bagaimana pengaruh penerapan *contextual word embedding* menggantikan *non-contextual word embedding* dalam bidang *image captioning*?



### **I.3 Tujuan**

Tujuan dari pelaksanaan tugas akhir ini adalah:

1. Menghasilkan model yang dapat menerima tambahan informasi untuk membangkitkan *caption* yang memiliki konteks.
2. Mengetahui dampak penerapan *contextual word embedding* (misalnya: BERT, GPT-2) dibandingkan penggunaan *non-contextual word embedding* (misalnya: Word2Vec) dalam bidang *image captioning*.

### **I.4 Batasan Masalah**

Batasan masalah yang digunakan dalam pelaksanaan tugas akhir ini adalah:

1. *Dataset* yang digunakan adalah *dataset* publik untuk *image captioning*, misalnya Flickr 8K dan MSCOCO. Sifat dari *dataset* publik umumnya tidak memiliki hubungan satu sama lain antar gambar (independen) dan tersedia dalam bahasa Inggris. Gambar dalam *dataset* publik bersifat tidak abstrak, tidak buram dan memiliki intensitas cahaya yang memadai (objek terlihat jelas).
2. Komponen *encoder* untuk mengubah gambar menjadi vektor menggunakan *pre-trained model* (misalnya Xception, Inception, dan VGG19).

### **I.5 Metodologi**

Metodologi utama yang digunakan dalam pelaksanaan tugas akhir ini adalah penelitian kualitatif disertai analisis kuantitatif pendukung. Tahapan yang dilalui dalam pelaksanaan tugas akhir ini adalah sebagai berikut:

1. Analisis Persoalan

Pengerjaan tugas akhir diawali dengan eksplorasi dan analisis terhadap riset terkait dalam bidang *image captioning* dan *natural language generation*. Pada tahap ini, dilakukan pembelajaran literatur untuk mempelajari gagasan utama, teknik implementasi, arsitektur model, hasil penelitian dan konklusi atau kesimpulan dari masing-masing penelitian yang sekiranya relevan dan

diperlukan. Langkah ini juga dilakukan untuk mencari alternatif solusi yang diimplementasikan.

## 2. Analisis dan Perancangan Solusi

Selanjutnya dilakukan perancangan dan pemilihan teknik dan arsitektur model berdasarkan analisis terhadap aspek efektivitas, kinerja pada penelitian terkait, dan kesesuaian model dengan permasalahan.

## 3. Implementasi

Pada tahap ini, masing-masing komponen yang terdapat di dalam arsitektur model diimplementasikan (baik dari nol maupun menggunakan kakas yang sudah tersedia) dan diuji untuk memastikan agar implementasi sudah sesuai seperti rancangan yang diusulkan.

## 4. Eksperimen dan Evaluasi

Eksperimen dilakukan untuk mendapat model terbaik. Evaluasi model dilakukan untuk menguji tingkat generalisasi model.

### **I.6 Sistematika Pembahasan**

Penyusunan Laporan Tugas Akhir ini dibagi menjadi 6 bagian, yaitu:

1. Bab Pendahuluan, bab ini berisi penjelasan dasar pemikiran, kebutuhan atau permasalahan-permasalahan yang ingin diselesaikan pada tugas akhir. Bab ini terdiri dari enam sub bab, yaitu: latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan sistematika pembahasan.
2. Bab Studi Literatur, berisi penjelasan mengenai perkembangan metode, arsitektur, komponen dari domain yang relevan dengan permasalahan dan solusi yang dibangun. Selain itu, bab ini juga berisi penjelasan singkat penelitian terkait yang dijadikan acuan utama dalam membangun solusi.
3. Bab Deskripsi Solusi, berisi analisis persoalan dan rancangan solusi yang diajukan pada Tugas Akhir. Analisis persoalan terdiri atas analisis *caption* dan konteks. Rancangan solusi berisi rancangan modul-modul yang diperlukan untuk membangun model *image captioning* yang dapat membangkitkan *caption* berkonteks.

4. Bab IV Implementasi, berisi penjelasan lingkungan implementasi yang digunakan dan penjelasan komponen-komponen yang telah diimplementasikan beserta kakas yang digunakan dalam menyelesaikan permasalahan.
5. Bab Eksperimen dan Pengujian, berisi penjelasan mengenai hasil eksperimen dan pengujian yang dilakukan serta analisisnya. Bab ini terdiri atas penjelasan mengenai tujuan eksperimen, skenario eksperimen, hasil eksperimen, skenario pengujian, dan hasil pengujian
6. Bab Kesimpulan dan Saran, berisi kesimpulan dan saran berdasarkan hasil pelaksanaan tugas akhir.

## BAB II

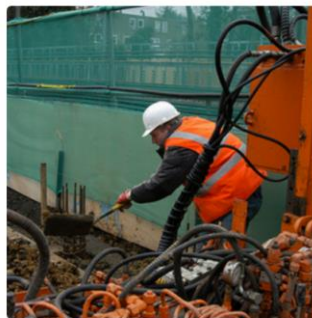
### STUDI LITERATUR

#### II.1 *Image Captioning*

*Image captioning* adalah irisan bidang *computer vision* dan *natural language generation* yang bertujuan untuk menghasilkan deskripsi teks (*caption*) dari sebuah gambar (*image*), beberapa contoh *image captioning* dapat dilihat pada gambar II.1. Untuk menghasilkan *output caption* yang baik, diperlukan teknik dari *computer vision* untuk menginterpretasikan konten dari gambar. Konten gambar dapat berupa objek (menggunakan pendekatan *object detection*), *context vector* dan *latent feature*. *Context vector* (vektor konteks) adalah istilah untuk vektor yang merepresentasikan makna dari sebuah gambar. Sederhananya, gambar yang serupa direpresentasikan dengan vektor konteks yang berdekatan. *Latent feature* adalah istilah yang dipakai untuk menyebut fitur yang ‘tersembunyi’ dan tidak langsung dapat diinterpretasikan. Dari bidang *natural language processing* (lebih spesifiknya *natural language generation*), diperlukan teknik pembangkitan *caption* berdasarkan vektor konteks suatu gambar.



"man in black shirt is playing guitar."



"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."

Gambar II.1 Contoh *Image Captioning* (Karpathy dkk., 2015)

### II.1.1 Topologi Model Seq2Seq

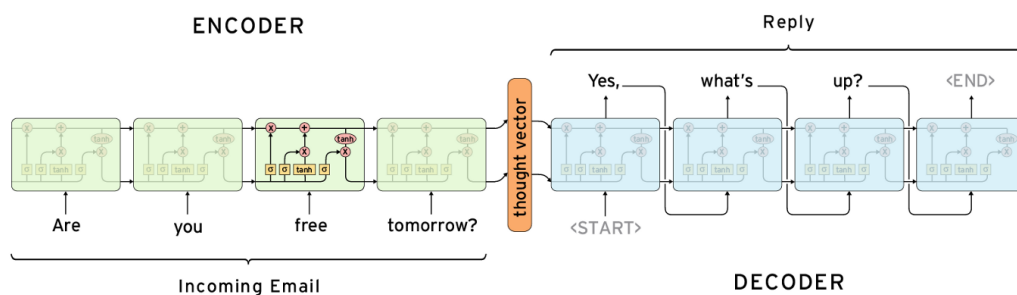
Dalam bidang *image captioning*, salah satu pendekatan yang umum digunakan adalah pendekatan *sequence-to-sequence* (Seq2Seq). Model *sequence-to-sequence* pertama kali diperkenalkan oleh peneliti Google, yaitu Sutskever dkk. (2014). Seq2Seq adalah suatu model yang bertujuan memetakan suatu *input* sekuensial dengan panjang tertentu, ke suatu *output* sekuensial dengan panjang tertentu (panjang *input* dan *output* dapat berbeda). Beberapa komponen penting dari model *sequence-to-sequence* yang perlu diketahui diantaranya *encoder* dan *decoder*. Contoh arsitektur *encoder-decoder* dapat dilihat di gambar II.2.

#### *Encoder*

*Encoder* adalah sebuah model (*Fully Connected*, CNN, RNN, dll) yang menerima *input* (baik gambar, teks, maupun bentuk lainnya), dan mengeluarkan sebuah representasi ‘fitur’ dalam bentuk vektor.

#### *Decoder*

*Decoder* adalah sebuah model (*Fully Connected*, CNN, RNN, dll), — umumnya memiliki struktur yang sama dengan *encoder* namun terbalik — yang bertujuan menghasilkan sekuens *output* sesuai domain permasalahan (umumnya berupa teks) berdasarkan ‘fitur’ yang dihasilkan *encoder*.



Gambar II.2 Topologi *Sequence-to-Sequence* Model (Abeywardana, 2017)

Model *sequence-to-sequence* bekerja dengan menerima *input* menggunakan *encoder* (misalnya RNN) untuk mendapatkan representasi vektor *input*, kemudian representasi vektor tersebut digunakan sebagai *input* untuk *decoder* (misalnya RNN) yang menghasilkan sekuens *output* yang dicari. Secara formal, tujuan dari model Seq2Seq adalah untuk memaksimalkan probabilitas bersyarat  $p(y_1, \dots, y_{T'} | x_1, \dots, x_T)$  dimana  $(x_1, \dots, x_T)$  adalah sekuens masukan dan  $(y_1, \dots, y_{T'})$  adalah sekuens keluaran dengan panjang  $T'$  yang mungkin berbeda dari  $T$ . Sutskever dkk. merumuskan persamaan umum model (II.1) sebagai berikut:

$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, \dots, y_{t-1}) \quad (\text{II.1})$$

dimana  $v$  adalah vektor yang merepresentasikan *input*  $(x_1, \dots, x_T)$  yang diambil dari *hidden state* terakhir *encoder*.

#### II.1.1.1 Encoder-Decoder dalam Image Captioning

Dalam melakukan *image captioning*, umumnya jenis *encoder* yang digunakan adalah CNN. Vektor fitur dari gambar umumnya didapatkan dari keluaran *layer* terakhir CNN sebelum *predictor layer* (keluaran *layer* kedua terakhir dari CNN). Vektor yang dihasilkan *encoder* dapat digunakan untuk berbagai hal; misalnya deteksi objek, klasifikasi gambar, segmentasi gambar, dsb. Vinyals dkk. (2014) menggunakan model CNN Inception sebagai komponen *encoder sequence-to-sequence* yang umumnya berupa LSTM, untuk menghasilkan vektor representasi dari gambar *input* dan kemudian vektor tersebut digunakan sebagai *input* untuk *decoder LSTM*. Kerangka kerja Vinyals dkk. (2014) ditunjukkan pada gambar II.3. Sharma dkk. (2018) mendefinisikan tiga komponen utama dalam arsitektur *image captioning* berbasis RNN sebagai berikut:

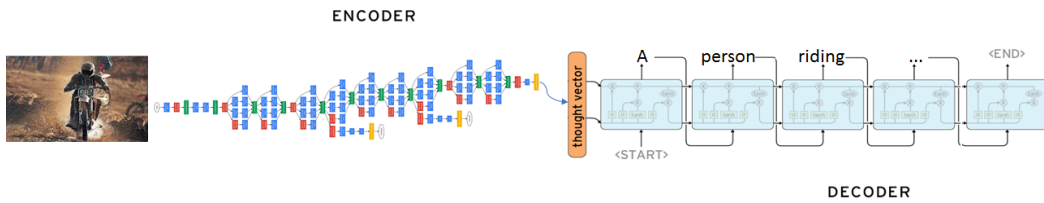
1. Sebuah model CNN yang menerima gambar dan menghasilkan sebuah vektor representasi gambar  $\mathbf{X} = (x_1, x_2, \dots, x_L)$ .
2. Sebuah *encoder* yang menerima vektor representasi gambar dan mengeluarkan sebuah vektor  $\mathbf{H} = f_{enc}(\mathbf{X})$ .

3. Sebuah *decoder* yang membangkitkan keluaran  $z_t = f_{dec}(Y_{1:t}, H)$  pada setiap tahap  $t$ , berdasarkan  $H$  dan juga *input decoder*  $Y_{1:t}$ .

yang kemudian disusun menjadi sebuah model sehingga memenuhi persamaan II.2 dan II.3:

$$h_l \triangleq RNN_{enc}(x_l, h_{l-1}), \text{ dan } H = h_L, \quad (\text{II.2})$$

$$z_t \triangleq RNN_{dec}(y_t, z_{t-1}), \text{ dimana } z_0 = H \quad (\text{II.3})$$



Gambar II.3 CNN *Encoder* (Inception) - RNN *Decoder* (LSTM) (Vinyals dkk., 2014)

#### II.1.1.2 Arsitektur *Encoder-Decoder*

Tanti dkk. (2017 & 2018) dalam dua penelitiannya, mengusulkan dua jenis arsitektur *encoder-decoder*, yaitu *inject* model dan *merge* model. Dalam pendekatan model *inject*, vektor dari *encoder* dan vektor fitur lainnya digabungkan dengan vektor masing-masing kata (*caption*) yang telah dibangkitkan. Dalam pendekatan ini, *decoder* memanfaatkan vektor dari gambar dan vektor masing-masing kata sebagai *input* untuk menghasilkan kata berikutnya, dengan kata lain, *decoder* mempelajari fitur linguistik dan visual secara bersamaan.

Pada pendekatan model *merge*, vektor dari *encoder* digabungkan dengan keseluruhan vektor *caption* yang telah dibangkitkan, selanjutnya vektor hasil penggabungan diberikan sebagai *input* untuk *decoder*. Berdasarkan penelitian Tanti dkk. (2017), umumnya *merge* model menghasilkan *caption* yang lebih baik dibandingkan model *inject*. Menurut Tanti dkk., hal tersebut disebabkan karena *encoder* yang digunakan dalam model *inject* hanya berfokus untuk meng-*encode*

teks yang sudah dihasilkan. Terjadi pembagian tugas antara RNN *decoder* dan CNN *encoder*, dimana RNN melakukan pembelajaran linguistik dan CNN melakukan pembelajaran visual.

Terdapat beberapa cara untuk menggabungkan vektor gambar yang dihasilkan *encoder* dan vektor teks yang dihasilkan *decoder*; diantaranya konkatenasi, perkalian, dan penjumlahan. Tanti dkk. dalam penelitiannya menunjukkan bahwa metode penjumlahan menghasilkan nilai evaluasi terbaik.

Keberhasilan *merge* model mengalahkan *inject* model sebenarnya di luar ekspektasi Tanti dkk. Umumnya RNN memiliki peran besar dalam *sequence generation model*. Namun penelitian Tanti dkk. menyangkal pemahaman tersebut dan mengusulkan agar peran RNN lebih baik dibatasi hanya menjadi *encoder* untuk *caption* yang sedang diprediksi, dan bukan sebagai model pembangkit sekuens.

### **II.1.2 Topologi Model Transformer**

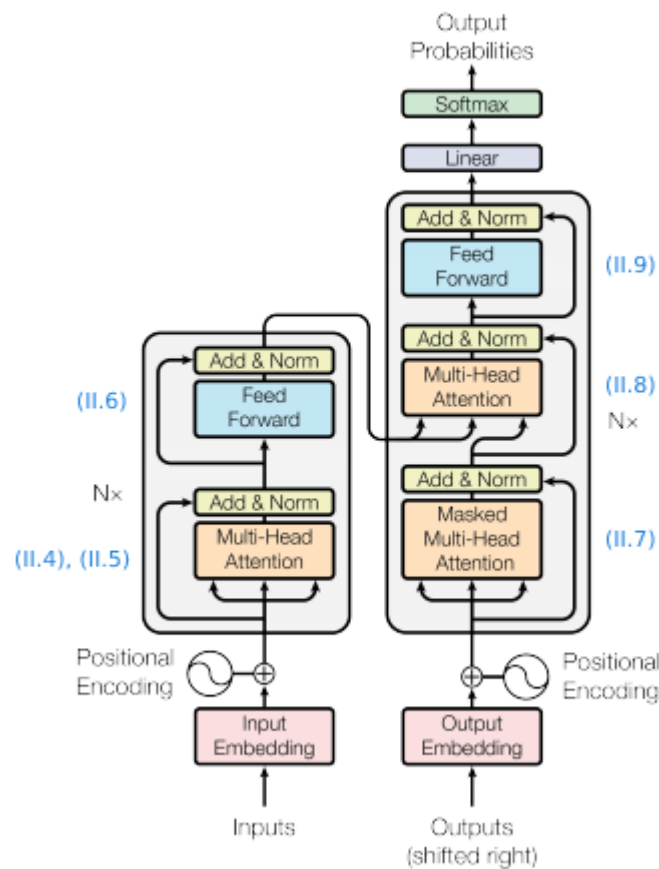
Selain model Seq2Seq yang memanfaatkan *encoder-decoder*, Vaswani dkk. (2017) memperkenalkan suatu metode baru yang dinamakan *Transformer*. *Transformer* mengadopsi teknik *Attention*. Selain menerima vektor (representasi *input*) yang dihasilkan oleh *encoder*, *decoder* juga menerima *input* berupa *hidden state* dari *encoder*. Analoginya sama seperti manusia saat membaca suatu kalimat membaca kata per kata, namun tetap memiliki fokus terhadap kata-kata penting dalam kalimat tersebut.

*Transformer* sebenarnya juga memanfaatkan *encoder-decoder* tetapi berbeda dari model *encoder-decoder* pada umumnya karena tidak menggunakan model RNN (*Recurrent network*). Salah satu alasan mengganti RNN adalah karena keterbatasan RNN yang harus dieksekusi secara sekuensial, tidak seperti CNN yang dapat memproses *input* secara paralel. Sayangnya, CNN tidak terlalu membantu dalam hal mempelajari keterkaitan antar kata dalam sekuens, oleh karena itu dikembangkanlah teknik *self-attention* (Vaswani dkk., 2017).

Arsitektur lengkap *transformer* ditunjukkan oleh Vaswani, dkk. (2018) pada gambar II.4. Komponen *encoder* (kiri) dan *decoder* (kanan) masing-masing terdiri dari beberapa bagian, yang dapat disusun beberapa lapis (notasi  $N \times$  pada gambar).



*Encoder* dan *decoder* yang digunakan tidak memanfaatkan RNN, melainkan tersusun dari *Multi-head Attention* dan *Feed forward layers*. *Multi-head Attention* digunakan *encoder* dan *decoder* untuk membuat representasi kalimat untuk setiap tahap  $t$ , dari kata pertama sampai kata ke- $t$ . Pada *decoder*, representasi ini selanjutnya digabungkan dengan *encoder-attention* yang merupakan *output* dari *encoder* untuk kembali dilewatkan pada *multi-head attention* kedua. *Feed forward layers* digunakan setelah *multi-head attention* untuk mendapatkan representasi dengan dimensi yang diinginkan (Vaswani dkk., 2017).

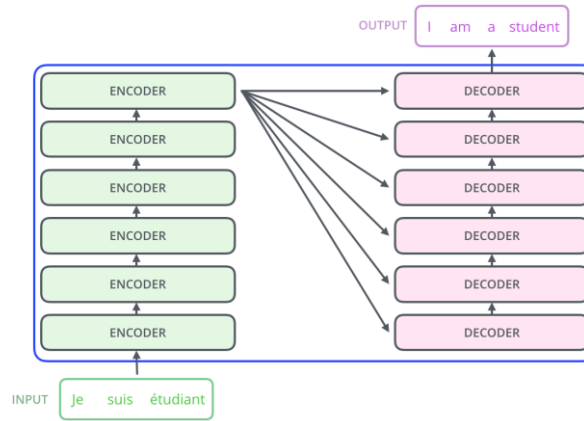


Gambar II.4 *Transformer* (Vaswani dkk., 2017) yang dilengkapi nomor formula yang digunakan

Karena model yang digunakan bukan model rekuren yang dapat menangkap urutan *input*, maka urutan sekuens harus secara eksplisit dicantumkan sebagai *input* (dalam

bentuk vektor n-dimensi). Vektor tersebut ditambahkan oleh komponen *positional encoding*. Seperti model NLP lainnya, teks *input* harus diubah menjadi vektor terlebih dahulu sebelum dimasukkan ke dalam model, tahapan ini dilakukan oleh komponen *input embedding* dan *output embedding* (Vaswani dkk., 2017).

Umumnya sebuah model *transformer* terdiri dari 6 lapis *encoder* dan *decoder* seperti yang disarankan Vaswani dkk. (2017). Walaupun terdiri dari beberapa lapis, hanya *output* terakhir dari susunan *encoder* yang digunakan pada setiap lapisan *decoder* (gambar II.5).



Gambar II.5 Ilustrasi Umum *Transformer* (Alammar dkk., 2018)

Sharma dkk. (2018) menyatakan komponen *encoder* ke- $n$  dari tumpukan  $N$  lapis *encoder* sebagai  $X_n = \{x_{n,1}, \dots, x_{n,L}\}$ , dan  $X_0 = X, H = X_N$ . Masing-masing lapis terdiri dari dua komponen: sebuah *multi-head self-attention layer* *ATTN* dan sebuah *position-wise feedforward network* *FFN* yang memenuhi persamaan:

$$x'_{n,j} = \text{ATTN}(x_{n,j}, X_n; W_q^e, W_k^e, W_v^e) \quad (\text{II.4})$$

$$x'_{n,j} \triangleq \text{softmax}(\langle x_{n,j} W_q^e, X_n W_k^e \rangle) X_n W_v^e \quad (\text{II.5})$$

$$X_{(n+1),j} = \text{FFN}(x'_{n,j}; W_f^e) \quad (\text{II.6})$$

(simbol  $\triangleq$  (*delta equal*): “sama berdasarkan definisi”) dimana  $W_q^e, W_k^e, W_v^e$  adalah matriks bobot untuk *query, key, value* pada komponen *self-attention*; dan  $W_f^e$  adalah matriks bobot *feedforward network*.

Komponen *decoder* ke- $n$  dinyatakan dalam notasi  $Z_n = \{z_{n,1}, \dots, z_{n,T}\}$  dan  $Z_0 = Y$ . Berbeda dari *encoder*, komponen *decoder* memiliki *mask* yang mencegah *decoder* pada posisi  $t$  menerima informasi pada posisi  $t+1 \dots T$ . *Decoder* juga memiliki *cross-attention layer* tambahan yang menghubungkan  $z_{n,j}$  dengan *output encoder*  $H = X_N$ .

$$z'_{n,j} = \text{ATTN}(z_{n,j}, Z_{n,1:j}; W_q^d, W_k^d, W_v^d) \quad (\text{II.7})$$

$$z''_{n,j} = \text{ATTN}(z'_{n,j}, H; W_q^c, W_k^c, W_v^c) \quad (\text{II.8})$$

$$Z_{(n+1),j} = \text{FFN}(z''_{n,j}; W_f^d) \quad (\text{II.9})$$

dimana  $W_q^d, W_k^d, W_v^d$  adalah matriks bobot untuk *query, key, value* pada komponen *self-attention*;  $W_q^c, W_k^c, W_v^c$  adalah matriks bobot pada komponen *cross-attention*; dan  $W_f^d$  adalah matriks bobot *feedforward network*. Korelasi antara persamaan dan arsitektur *transformer* ditunjukkan pada gambar II.4.

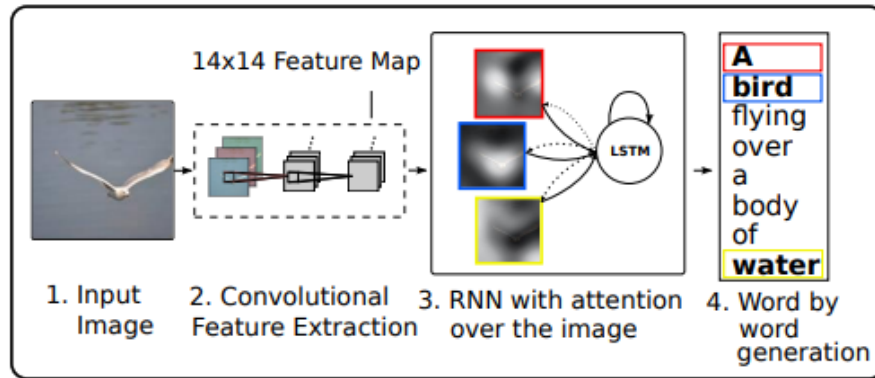
### II.1.3 Kerangka Kerja *Image Captioning*

Kiros dkk. (2014) adalah peneliti yang pertama kali memanfaatkan *neural network* untuk menyelesaikan masalah *image captioning*. Dalam penelitiannya, Kiros dkk. mengajukan sebuah kerangka kerja untuk melakukan *image captioning*.

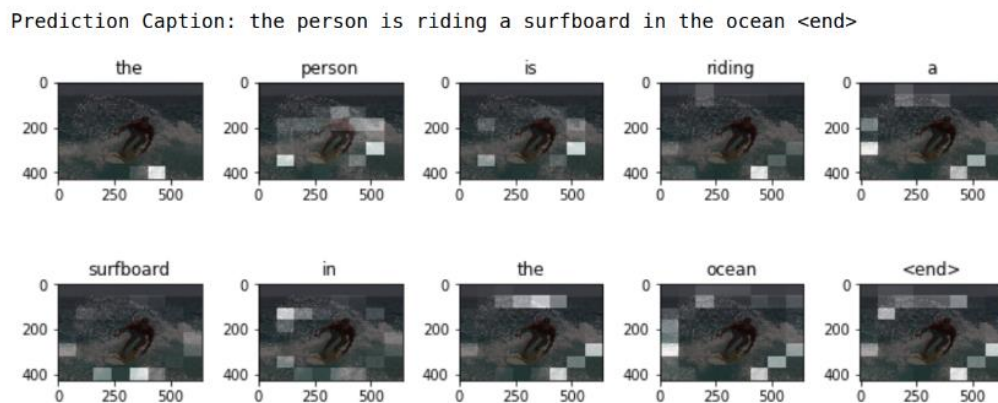
Pertama, gambar dimasukkan ke dalam *encoder CNN* untuk mendapatkan vektor fitur. Selanjutnya vektor fitur dimasukkan ke dalam *decoder RNN* untuk menghasilkan kalimat *output*. Proses pembelajaran (*backpropagation*) didasarkan pada tingkat kesalahan — didefinisikan sebagai perbedaan antara *caption output* dibandingkan dengan *caption asli* — yang dihitung dengan *loss function* seperti *cross entropy / maximum likelihood*. Aspek kemiripan kalimat digunakan sebagai metrik evaluasi.

Xu dkk. (2016) juga menggambarkan kerangka kerja *image captioning* dengan cara yang serupa (gambar II.6). Gambar yang menjadi *input* dimasukkan ke dalam *CNN*

*encoder* untuk mendapatkan fitur vektor, kemudian vektor tersebut dimasukkan ke dalam LSTM *decoder* berulang kali untuk menghasilkan kata satu per satu sampai keseluruhan *caption* terbentuk. Contoh pembangkitan ditunjukkan gambar II.7.



Gambar II.6 Kerangka Kerja *Image Captioning* (Xu dkk., 2016)



Gambar II.7 Proses *Image Captioning* (Tensorflow.org, 2020)

Pada tahap pembangkitan *caption*, karena RNN / LSTM memerlukan *hidden state* saat memprediksi kata pertama, maka digunakan *dummy* token <start> sebagai token pertama untuk semua *caption*. Token <end> digunakan sebagai penanda *caption* selesai. Misalkan pada gambar II.7, untuk memprediksi kata pertama 'the',

diperlukan *input* berupa vektor gambar dan vektor dari token ‘<start>’. Proses pembangkitan *caption* dilakukan sampai *decoder* membangkitkan token <end>.

## II.2 Contextual Word Embedding

*Word embedding* sederhananya adalah representasi vektor dari suatu kata tertentu. *Word embedding* digunakan karena kalkulus yang mendasari pembelajaran mesin tidak mengoperasikan teks, sehingga representasi teks harus diubah menjadi vektor terlebih dahulu.

Cara paling sederhana adalah menggunakan *one-hot vector*, dimana vektor suatu kata bernilai 1 untuk indeks kata yang dituju, dan 0 untuk indeks lainnya. Misalkan untuk kosakata {‘Hai’, ‘selamat’, ‘pagi’}, didapatkan vektor ‘Hai’ = [1,0,0]; ‘selamat’= [0,1,0]; ‘pagi’= [0,0,1]. Namun mengingat jumlah kata unik yang ada sangat banyak (lebih dari ribuan), tidak efisien menggunakan pendekatan seperti ini. Maka dari itu, diciptakanlah representasi vektor general yang dapat menangkap kedekatan sintaksis kata dengan dimensi vektor yang relatif lebih kecil, salah satunya Word2Vec. Dengan pendekatan seperti ini, secara intuitif, kata-kata yang mirip cara penggunaannya (misalnya sinonim) memiliki vektor yang berdekatan.

*Word embedding* yang digunakan sampai akhir 2017 pada umumnya adalah tipe *non-contextual word embedding*, misalnya: GloVe (Pennington dkk., 2014) dan Word2Vec (Mikolov dkk., 2013). Pada *non-contextual word embedding*, suatu kata hanya memiliki satu vektor representasi, walaupun kata tersebut dapat memiliki banyak makna berbeda. Sebagai contoh, misalkan kata “bunga” pada kalimat “Pemerintah menaikkan suku bunga bank” dan “Sammy menanam bunga di kebun” memiliki vektor yang sama walaupun kedua kata tersebut memiliki perbedaan makna karena perbedaan konteks kalimat.

Menindaklanjuti permasalahan tersebut, banyak penelitian dilakukan dengan tujuan untuk mengembangkan *contextualized word embedding*. Para peneliti berusaha memperbaiki *word embedding* yang sudah ada agar dapat menangkap makna dan konteks kalimat (semantik) serta susunan penggunaan kata (sintaktik) dengan lebih baik. Beberapa contoh *contextual word embedding* yang berhasil dikembangkan diantaranya CoVe (McCann dkk., 2017), ELMo (Peters dkk., 2018), *Cross-View*

Training (Clark dkk., 2018), ULMFiT (Howard & Ruder, 2018), OpenAI GPT (Radford dkk., 2018), dan BERT (Devlin dkk., 2019) yang menjadi *state-of-the-art* saat ini.

*Contextual word embedding* yang digunakan umumnya berupa keluaran model bahasa yang dilatih untuk memperhatikan kalimat dan susunan kalimat. Model *non-contextual* umumnya tidak memerlukan komponen model setelah dilatih dan hanya menyimpan representasi vektor yang telah dipelajari. Pada model *contextual*, model bahasa yang telah dilatih diperlukan pada proses pembentukan *embedding* (kalimat dimasukkan dalam model pada saat inferensi). Oleh karena itu, model *contextual* dapat menghasilkan representasi yang berbeda-beda untuk satu kata (*token*) berdasarkan posisi kata tersebut dan kata-kata lain dalam suatu kalimat (Rajasekharan, 2019).

### II.2.1 OpenAI GPT

Pada tahun yang sama (2018), Radford dan tim OpenAI juga mengembangkan model bahasa yang memiliki kemiripan dengan ELMo, namun dengan skala model yang jauh lebih besar (dibanding ELMo); model tersebut diberi nama GPT (*Generative Pre-training Transformer*). Radford dkk. melatih model GPT menggunakan *dataset* ‘BooksCorpus’ (mengandung lebih dari 7000 judul buku dengan bermacam genre) dan *dataset* ‘Word Benchmark’. GPT berhasil menjadi *state-of-the-art* untuk hampir semua *natural language task* pada saat itu (Juni 2018).

Walaupun terlihat mirip, GPT memiliki dua perbedaan utama dengan ELMo:

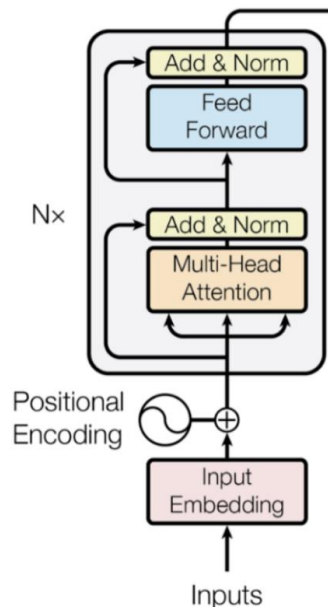
1. Arsitektur model: ELMo menggunakan penggabungan (kata asli: *concatenation*) dua model *multi-layer* LSTM yang dilatih terpisah (kiri-ke-kanan dan kanan-ke-kiri), sedangkan GPT memanfaatkan model *multi-layer transformer decoder*.
2. Cara penggunaan *contextualized embedding* hasil training: pada ELMo, *embedding* yang dihasilkan digunakan sebagai fitur tambahan untuk model *task-specific*, sedangkan pada GPT tidak ada pemisahan antara model bahasa dan *task-specific* model. Model bahasa yang sama disempurnakan (*tune*) untuk semua *task specific*.

Salah satu perkembangan yang penting dari penelitian GPT adalah pendekatan yang tidak memerlukan *task-specific* model, cukup dengan model bahasa saja. Walaupun demikian model GPT masih memiliki satu keterbatasan yaitu model hanya mempelajari kalimat dari kiri ke kanan (satu arah / unidirectional).

## II.2.2 BERT

BERT (*Bidirectional Encoder Representations from Transformers*) yang dikembangkan oleh Devlin, dkk. (2019) adalah model turunan langsung dari GPT yang berhasil menjadi *state-of-the-art* dan dengan mudah mengalahkan model terbaik yang telah dikembangkan di hampir semua *task* NLP yang ada. BERT dilatih menggunakan data Wikipedia dan BookCorpus (~3,3 miliar kata). Satu hal yang menjadi terobosan utama dari BERT (dibandingkan GPT) adalah sifat *bidirectional* yang tidak dimiliki oleh GPT.

BERT menggunakan arsitektur *multi-layer bidirectional transformer encoder* (gambar II.8). BERT dilatih untuk mengerjakan dua tugas: *mask language modelling* dan *next sentence prediction*; dengan tujuan agar BERT dapat menghasilkan prediksi yang bersifat *bidirectional* (memperhatikan konteks dari dua arah) dan memperhatikan konteks kalimat.

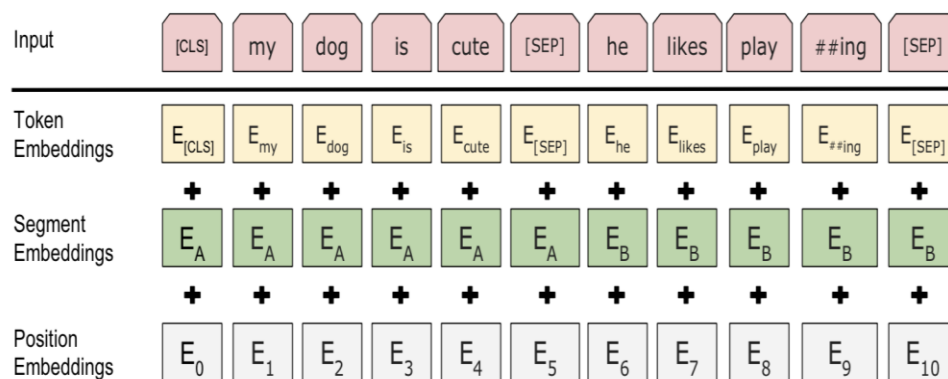


Gambar II.8 Arsitektur Model *Transformer Encoder* (Vaswani dkk., 2017)

Proses pembelajaran BERT menggunakan pendekatan “*mask language modelling*”. Pendekatan ini dilakukan dengan 'menyembunyikan' 15% token (kata) di setiap sekuens secara acak. Token yang disembunyikan dapat diganti dengan token [MASK]. Namun permasalahannya, token [MASK] tidak pernah ditemukan / muncul pada tahapan *fine-tuning*. Oleh karena itu, tidak dapat hanya sekedar mengganti token dengan [MASK], perlu ada beberapa trik heuristik lainnya, misalnya:

1. Mengganti token yang dipilih dengan token [MASK] (probabilitas 80%).
2. Mengganti token yang dipilih dengan token sembarang (probabilitas 10%).
3. Token yang dipilih tidak diubah (probabilitas 10%).

BERT dilatih untuk memprediksikan kata-kata yang dihilangkan, namun tidak mengetahui kata mana yang telah diganti, maupun kata mana yang seharusnya diganti. Masukan yang diterima BERT adalah penjumlahan dari tiga vektor (gambar II.9), diantaranya:



Gambar II.9 Representasi Masukan BERT (Devlin dkk., 2019)

1. *WordPiece tokenization embeddings*: Intinya memecah suatu kalimat atau frasa menjadi bagian yang lebih kecil yang disebut “WordPiece”.
2. *Segment embedding*: Jika kalimat *input* terdiri dari dua kalimat, maka kalimat pertama diberi *embedding* A, sedangkan kalimat kedua diberi



*embedding* B dan dipisahkan dengan sebuah token pemisah [SEP] (separator). Apabila *input* hanya berupa satu kalimat, hanya *embedding* A yang dipakai.

3. *Position embedding: embedding* urutan dari ‘*WordPiece*’.

Contoh vektor *output* BERT vs Word2Vec untuk kata ‘bank’ pada kalimat: “*After stealing money from the bank vault, the bank robber was seen fishing on the Mississippi riverbank.*”

BERT:

bank (*vault*): [ 1.1868, -1.5298, -1.3770, 1.0648, 3.1446, 1.4003, -4.2407, ...]

bank (*robber*): [ 2.1319, -2.1413, -1.6260, 0.8638, 3.3173, 0.1796, -4.4853, ...]

bank (*river*): [ 1.1295, -1.4725, -0.7296, -0.0901, 2.4970, 0.5330, 0.9742, ...]

Word2Vec:

bank (*vault*): [ 0.6744, 2.1924, -4.5306, 0.4758, 1.1703, -4.4859, 0.7119,...]

bank (*robber*): [ 0.6744, 2.1924, -4.5306, 0.4758, 1.1703, -4.4859, 0.7119,...]

bank (*river*): [ 0.6744, 2.1924, -4.5306, 0.4758, 1.1703, -4.4859, 0.7119,...]

### II.3 Evaluasi Kinerja

Beberapa *task* dalam NLP yang relatif sederhana, misalkan analisis sentimen, dapat dengan mudah dievaluasi dengan metrik seperti akurasi / f1-score. Namun untuk permasalahan di luar klasifikasi, misalnya peringkasan teks, *question answering*, *chatbots*, *machine translation*, dll. dibutuhkan sistem penilaian yang sedikit berbeda dari metrik penilaian standar. Beberapa metode penilaian yang umum digunakan dalam penelitian pembangkitan sekuens (terutama di bidang *machine translation*) adalah BLEU, ROUGE, dan METEOR.

Intuisi dibalik penilaian teks yang dihasilkan sebenarnya sama saja dengan cara penilaian kesesuaian label. Jika teks A lebih mirip dengan teks referensi daripada teks B, maka seharusnya nilai A lebih tinggi dari B. Seperti pada metrik lainnya, penilaian menggunakan *precision* (atau *specificity*) dan *recall* (atau *sensitivity*).

*Precision* adalah rasio jumlah kata yang diprediksi dengan benar, dari keseluruhan kata yang diprediksi. *Recall* adalah rasio jumlah kata yang diprediksi dengan benar, dari keseluruhan kata yang benar (yang ada di referensi).

Misalkan:

Teks referensi: *A cat napping on the mat.*

Teks A: *Cat napping*

Teks B: *Cat sleeping on the mat.*

Pada contoh diatas, *precision* dari A adalah 100% (semua kata yang ditebak terdapat dalam referensi), dibandingkan B yang hanya 80% (hanya 4 dari 5 kata yang ditebak terdapat dalam referensi). Namun dari segi *recall*, A memiliki *recall* 33% (hanya 2 dari 6 kata dalam referensi yang dimunculkan), sedangkan B 66% (ada 4 dari 6 kata dalam referensi yang dimunculkan). Dalam contoh sederhana ini, hanya digunakan pencocokan satu kesatuan kata (unigram) untuk memudahkan ilustrasi.

### II.3.1 BLEU

BLEU adalah singkatan dari *Bilingual evaluation understudy* yang pertama kali dikembangkan oleh Papineni dkk. (2002). Sejauh ini, BLEU adalah metrik yang paling populer dalam mengevaluasi model *machine translation*. Dalam metrik BLEU, *precision* dan *recall* di aproksimasi dengan menggunakan teknik *modified n-gram* dan *best match length*.

Mengacu pada contoh pada sub bab II.3, misalkan ditambahkan teks contoh lainnya:

Teks referensi: *A cat napping on the mat.*

Teks B: *Cat sleeping on the mat.*

Teks C: *Cat sleeping on the mat on the mat.*

*Precision* unigram dari teks B adalah 80%, sedangkan teks C adalah 87.5% (7 dari 8), padahal secara nalar kalimat C tidak lebih mirip dengan referensi dibandingkan kalimat B. Maka dari itu, diperlukan sedikit pengubahan dimana perhitungan kata yang muncul di teks (B dan C) hanya dilakukan maksimal sebanyak kemunculan

kata tersebut di referensi. Jadi untuk contoh di atas, unigram ‘on’, ‘the’ dan ‘mat’ hanya dihitung satu kali, sehingga *precision* untuk teks C menjadi 50% (4 dari 8).

Setelah menghitung semua *precision* dengan n-gram tertentu (misal unigram, bigram, trigram, dll), untuk menggabungkan semua nilai itu digunakan rata-rata geometris. Papineni dkk. menjelaskan bahwa *precision* berkurang secara eksponensial terhadap n, sehingga perlu digunakan rata-rata logaritmik agar tetap adil untuk semua n-gram.

Rumus perhitungan untuk presisi dinyatakan oleh Papineni dkk. (2002) sebagai berikut:

$$Precision = \exp(\sum_{n=1}^N w_n \log p_n), \quad (II.10)$$

$w_n$  = bobot positif untuk n-gram (total seluruh  $w_n = 1$ )

$p_n$  = presisi untuk n-gram

Permasalahan yang dihadapi oleh *recall* adalah kalimat referensi yang digunakan dapat berjumlah lebih dari satu, sehingga sulit menghitung sensitivitas teks kandidat terhadap referensi. Namun secara intuitif, kalimat kandidat yang lebih panjang memiliki peluang lebih besar untuk memunculkan kata-kata yang terdapat dalam kalimat referensi, jika dibandingkan dengan kalimat yang pendek. Disisi lain, jika kalimat terlalu panjang, nilai *precision* secara alami menjadi rendah (banyak kata yang salah). Oleh karena itu, untuk menghitung *recall* Papineni dkk. (2002) menggunakan *brevity penalize* (BP) yang dirumuskan sebagai berikut:

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \quad (II.11)$$

$c$  = panjang teks kandidat

$r$  = rata-rata dari panjang teks referensi (jika lebih dari 1)

Setelah mendefinisikan *precision* dan BP, secara keseluruhan, Papineni dkk. (2002) merumuskan BLEU menjadi:

$$\text{BLEU} = \text{BP} \cdot \exp \left( \sum_{n=1}^N w_n \log p_n \right) \quad (\text{II.12})$$

Dalam penelitiannya, Papineni dkk. menggunakan  $N = 4$  dan bobot  $w_n = 1 / N$ .

### II.3.2 ROUGE

*Recall Oriented Understudy for Gisting Evaluation (ROUGE)* di kembangkan Lin (2004). Sesuai namanya, ROUGE hanya memperhatikan *recall*. ROUGE sering digunakan sebagai metrik untuk menilai *text summarization*. ROUGE memiliki banyak varian yaitu ROUGE-N (N-gram), ROUGE-L (*longest common subsequence (LCS)*), ROUGE-W (*weighted LCS*), dan ROUGE-S (*skip-bigram co-occurrence statistic*).

Cara perhitungan *recall* pada ROUGE-N sama seperti perhitungan *recall* pada umumnya (rasio jumlah kata dalam referensi yang muncul pada teks hasil prediksi). Pada perhitungan *recall* ROUGE-L/W/S, Lin tidak hanya menggunakan *recall*, tetapi menggunakan pendekatan *F-score*. Misalkan untuk ROUGE-L, untuk teks A dan B dan teks referensi yang memiliki panjang m dan n, perhitungan dilakukan dengan (Lin, 2004):

$$P = \frac{LCS(A,B)}{m} \text{ dan } R = \frac{LCS(A,B)}{n} \quad (\text{II.13})$$

P = *recall* untuk kalimat referensi 1

m = panjang referensi 1

R = *recall* untuk kalimat referensi 2

n = panjang referensi 2

Kemudian *F-score* dapat dihitung dengan menggunakan rata-rata harmonik dari P dan R (Lin, 2004).

$$F = \frac{(1 + b^2) RP}{R + b^2 P} \quad (\text{II.14})$$

b = bobot atau weight

P = *recall* untuk kalimat referensi 1

R = *recall* untuk kalimat referensi 2

### II.3.3 METEOR

Banerjee dan Lavie (2005) mengembangkan METEOR (*Metric for Evaluation for Translation with Explicit Ordering*) yang diklaim mempunyai korelasi lebih baik dengan penilaian manusia. METEOR memodifikasi cara perhitungan *precision* dan *recall* dengan menggunakan *weighted F-score* dan *penalty function*.

Pertama, dilakukan pencarian terhadap n-gram yang memiliki korespondensi dari teks prediksi terhadap teks referensi. Pencarian dilakukan mulai dari yang kesamaan eksak (*exact match*), lalu kesamaan setelah melalui proses *stemming*, dan selanjutnya kemiripan menurut sinonim WordNet. Setelah mendapatkan daftar kata yang memiliki korespondensi, perhitungan *precision* dan *recall* dirumuskan Banerjee dan Lavie (2005) sebagai berikut:

$$P = \frac{m}{c}, R = \frac{m}{r} \quad (\text{II.15})$$

P = *precision*

R = *recall*

m = jumlah n-gram yang memiliki korespondensi

c = panjang kandidat

r = panjang referensi

kemudian dicari *F-score* dengan rumus (Banerjee dan Lavie, 2005):

$$F = \frac{PR}{\alpha P + (1-\alpha) R} \quad (\text{II.16})$$

P = *precision*

R = *recall*

$\alpha$  = bobot atau *weight*

Untuk memastikan keteraturan urutan kata dalam teks kandidat, Banerjee dan Lavie (2005) menggunakan sebuah fungsi penalti yang didefinisikan sebagai berikut:

$$Penalty = \gamma \left( \frac{c}{m} \right)^\beta \text{ dimana } 0 \leq \gamma \leq 1 \quad (\text{II.17})$$

$c$  = jumlah *chunk* yang berkorespondensi

$m$  = jumlah  $n$ -gram yang berkorespondensi

Dengan menggunakan rumus tersebut, apabila kebanyakan  $n$ -gram yang berkorespondensi terletak berdekatan / berurutan, maka jumlah *chunk* semakin kecil, dan *penalty* berkurang. Setelah mendefinisikan *F-Score* dan *Penalty*, maka Banerjee dan Lavie (2005) mendefinisikan METEOR sebagai:

$$METEOR = (1 - Penalty) F \quad (\text{II.18})$$

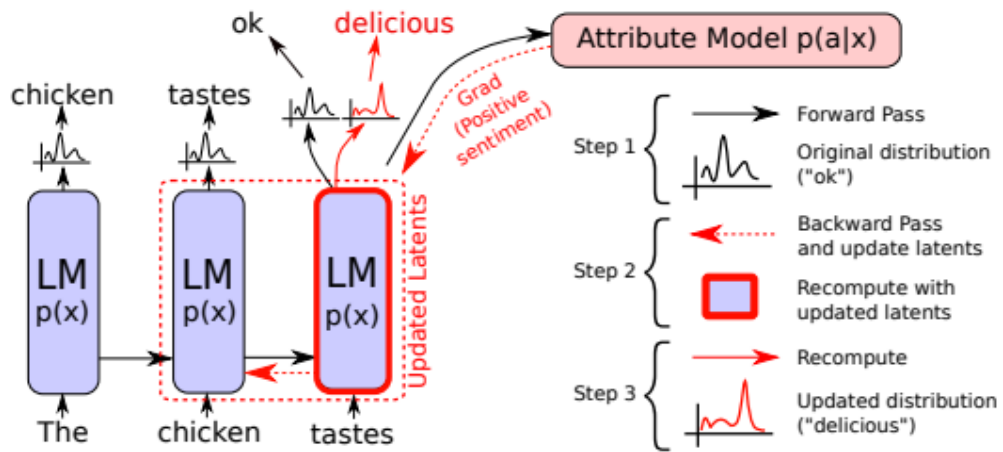
## II.4 Penelitian Terkait

### II.4.1 Penelitian Dathathri dkk. (2019)

Dilatarbelakangi banyaknya terobosan model bahasa berbasis *transformer* yang berkembang belakangan ini (BERT, GPT, dll.), Dathathri dkk. melakukan penelitian untuk menjawab pertanyaan apakah model bahasa berbasis *transformer* tersebut dapat dimanfaatkan sebagai *decoder* serbaguna, dimana salah satu syaratnya adalah model dapat dikendalikan sesuai kebutuhan tanpa perlu dilatih kembali. Pada umumnya, mengendalikan keluaran dari suatu model bahasa sulit dilakukan tanpa mengubah arsitektur model atau melakukan *fine-tuning* sesuai kebutuhan tertentu. Dathathri dkk. mengajukan suatu model yang disebut PPLM (*Plug and Play Language Model*) yang menggabungkan model bahasa yang sudah dilatih dengan satu atau lebih *attribute classifier* yang dapat menuntun pembangkitan teks dari model bahasa tersebut

Untuk mengendalikan keluaran dari model bahasa, pada setiap tahap pembangkitan  $t$ , PPLM menggeser vektor  $H_t$  (*hidden state* pada tahap  $t$ ), menuju arah penjumlahan dua jenis gradien. Gradien pertama bertujuan memaksimalkan *log-likelihood* atribut

$a$  dari model  $p(a|x)$ , dan yang kedua memaksimalkan  $\log\text{-likelihood } p(x)$  dari model bahasa asli. Dengan menggabungkan kedua gradien tersebut PPLM memiliki parameter yang dapat mengatur tingkat kekuatan pergeseran vektor  $H_t$  yang diinginkan. Perubahan  $H_t$  hanya dilakukan terhadap  $H_t$  (pembangkitan pada tahap  $t + 1$  cukup bergantung pada  $H_t$  saja) dan perubahan dilakukan sebanyak  $i$  iterasi. Secara keseluruhan proses PPLM dan dapat diinterpretasikan sebagai perubahan bertahap  $p(x | H_t)$  menuju distribusi keluaran  $p(x | H'_t, a)$  yang diinginkan berdasarkan atribut  $a$  (gambar II.10).



Gambar II.10 Ilustrasi PPLM (Dathathri dkk., 2019)

Atribut  $a$  yang digunakan dibagi menjadi dua jenis, yaitu atribut yang didapat dari *Bag of Words* (BoW) dan diskriminator. Pada model atribut BoW, digunakan vektor dari kumpulan kata yang tergabung dalam satu topik, lalu dicari jarak antara  $x$  dengan masing-masing vektor  $v$  dalam BoW, kemudian dijumlahkan sebagai fungsi  $loss$ . Untuk mendapatkan vektor atribut dari diskriminator, vektor  $x$  dimasukkan ke dalam model diskriminator  $D$  untuk menghitung  $loss$  antara  $output D(x)$  dan target yang diharapkan (misal: jenis kalimat positif / netral / negatif).

PPLM berhasil mengendalikan proses pembangkitan teks dari model bahasa yang besar, baik dengan BoW maupun model diskriminator yang sederhana. Tingkat

pengendalian PPLM juga dapat diatur sesuai kebutuhan sehingga menjadikan model PPLM fleksibel untuk semua kemungkinan atribut  $a$  yang digunakan.

#### II.4.2 Penelitian Keskar dkk. (2019)

Sebelum Dathathri dkk. (2019), Keskar dkk. (2019) melakukan penelitian serupa dengan tujuan menghasilkan model bahasa berbasis *transformer* yang dapat dikendalikan keluarannya sesuai keperluan. Model yang diajukan Keskar dkk. diberi nama CTRL, adalah suatu model bahasa yang dilatih menggunakan *control code* yang dapat mengendalikan gaya bahasa, konten dan perilaku spesifik *task* yang didapatkan dari struktur bahasa dan domain teks. Model bahasa tersebut mempelajari representasi teks  $x$  dan distribusi  $p(x|c)$  dimana  $c$  adalah *control code* untuk teks tersebut

Model bahasa yang dikembangkan berbasis *transformer* (Vaswani dkk., 2017) yang terdiri dari tumpukan *multi-head attention layer* dan *dense layer*. Saat pelatihan, *control code* ditambahkan pada awal kalimat menjadi bagian dari kalimat yang dilewatkan *embedding layer* dan seterusnya. *Dataset* yang digunakan mencapai 140 GB dan berasal dari berbagai sumber dan domain; diantaranya Wikipedia (En, De, Es, Fr), berbagai subReddit, OpenWebText, Amazon Reviews, dll. untuk mendapatkan teks dengan domain dan karakteristik yang berbeda-beda. Karena model dilatih berdasarkan *control code*, maka model bahasa ini dapat juga digunakan untuk mengklasifikasikan ‘sumber’ atau domain dari suatu teks. Hal ini dapat dicapai dengan teorema Bayes:

$$p(c|x) \propto p(x|c) p(c) \quad (\text{II.19})$$

Model bahasa yang dihasilkan memiliki kualitas tinggi karena model dilatih untuk memaksimalkan  $p(x|c)$  secara spesifik, namun untuk mencapai hal ini, tidak mudah dan relatif mahal untuk dilakukan. *Control code* yang digunakan juga harus ditetapkan dari awal, dan model yang dilatih tergolong sangat besar (1.6B parameter).



#### II.4.3 Penelitian Sharma dkk. (2018)





Sharma dkk. (2019) melakukan penelitian untuk menghasilkan model *image captioning* yang dapat menghasilkan *caption* berkonteks. Penelitian ini dibagi menjadi dua bagian besar, yaitu mempersiapkan *dataset caption* berkonteks, dan membuat model *image captioning*. Berbeda dari dua penelitian sebelumnya, Sharma dkk. membangun *dataset* baru yang diberi nama *Conceptual Captions dataset*; memiliki jumlah gambar hampir 10x lebih banyak dari MSCOCO dan dilengkapi dengan konteks.

*Dataset* yang dibangun berasal dari 5 miliar gambar dari gambar yang dapat ditemukan di internet, dan *captionnya* berbasis *alt-text* gambar tersebut. Gambar yang dapat dimasukkan dalam *dataset* telah melalui empat tahapan praproses yang menyisakan hanya sekitar 0.3% gambar:

1. Seleksi berdasarkan gambar: gambar diseleksi berdasarkan jenis *encoding*, ukuran, bentuk, resolusi, membuang gambar dengan konten negatif.
2. Seleksi berdasarkan teks: *Alt-text* cenderung tidak merepresentasikan deskripsi gambar dengan baik. Maka dari itu *alt-text* diseleksi berdasarkan *POS-tag*, jumlah kata benda, memiliki kata kerja, memiliki preposisi, dsb.), *alt-text* memiliki huruf kapital, tidak berulang, kata yang digunakan cukup umum, bukan merupakan kalimat *template*, dsb.
3. Seleksi berdasarkan gambar-teks: memanfaatkan Google Cloud Vision API untuk mengecek apakah kelas gambar relevan dengan *alt-text*.
4. Perubahan teks menggunakan hipernim: memanfaatkan Google Cloud Natural Language APIs dan Google Knowledge Graph untuk mengubah nama entitas dan kata ganti menjadi jenis asosiasi hipernimnya. Proses ini juga menghilangkan tanggal, durasi, jenis ukuran benda, dan menggabungkan benda yang sama menjadi kata majemuknya.

Model *image captioning* yang dibangun terbagi menjadi dua jenis: model berbasis rekuren (Vinyals dkk., 2015) dan berbasis *transformer* (Vaswani dkk., 2017). Model *image captioning* yang dibangun dilatih menggunakan *Conceptual Captions*

*dataset* dan dibandingkan dengan model pembanding yang dilatih dengan *dataset* MSCOCO. Model yang dilatih dengan *dataset* *Conceptual Captions dataset* memiliki hasil *caption* yang lebih baik, dan menangani kelemahan model yang dilatih dengan *dataset* MSCOCO (gambar II.11).

				
<b>COCO-trained</b>				
RNN8x8	a group of men standing in front of a building	a couple of people walking down a walkway	a child sitting at a table with a cake on it	a close up of a stuffed animal on a table
T2T8x8	a group of men in uniform and ties are talking	a narrow hallway with a clock and two doors	a woman cutting a birthday cake at a party	a picture of a fish on the side of a car
<b>Conceptual-trained</b>				
RNN8x8	graduates line up for the commencement ceremony	a view of the nave	a child's drawing at a birthday party	a cartoon businessman thinking about something
T2T8x8	graduates line up to receive their diplomas	the cloister of the cathedral	learning about the arts and crafts	a cartoon businessman asking for help

Gambar II.11 Perbandingan Hasil *Caption* (Sharma dkk., 2018)

## **BAB III**

### **DESKRIPSI SOLUSI**

Bab III Analisis dan Perancangan Solusi ini membahas hal terkait secara lebih detail; mencakup analisis permasalahan, alternatif solusi, pemilihan solusi berdasarkan kelebihan dan kekurangannya, serta perancangan umum arsitektur model sesuai solusi yang telah dipilih.

#### **III.1 Analisis Permasalahan**

Seperti sebelumnya disebutkan pada latar belakang (bab I.1), terdapat permasalahan dalam bidang *image captioning*, yaitu ketidaksesuaian *caption* yang dihasilkan dengan keperluan industri. Misalnya dari segi jenis *caption* dan konteks dari *caption* yang seringkali tidak ditunjukkan oleh model. Permasalahan lainnya muncul saat ingin menggunakan gambar pendukung untuk memperkaya konteks *caption* karena model yang ada saat ini tidak mengakomodasi *input* lebih dari satu gambar.

##### **III.1.1 Analisis Image Captioning**

Kebanyakan penelitian di bidang ini lebih berfokus pada bagaimana menghasilkan *caption* yang 'tepat' (minimal menunjukkan sesuatu yang dapat dilihat pada gambar) dan berkorelasi dengan gambar, namun seringkali tidak memperhatikan aspek kegunaan *caption* yang dihasilkan. Misalkan untuk gambar III.1a, apabila model menghasilkan *caption* "*Sashimi on black plate*" atau misalkan untuk gambar III.1b, model memberi *caption* "*Bedroom with wooden floor*" kemungkinan besar model dianggap sudah sangat baik. Namun apabila kasusnya adalah pemilik hotel dan restoran ingin membuat deskripsi untuk mempromosikan bisnisnya, tentu deskripsi tersebut kurang dapat diterima. Deskripsi yang diharapkan mungkin lebih menekankan "*sushi art*" untuk gambar III.1a dan keindahan laut yang terlihat dari jendela kamar untuk gambar III.1b.



Gambar III.1 Sampel Gambar (Mikyoui00, Centara Grand)

Kualitas *caption* yang dibangkitkan memang tidak dapat dipungkiri sangat bergantung pada kualitas data yang digunakan untuk melatih model. Ketika hasil *caption* yang dibangkitkan kurang bermakna, dugaan pertama adalah kualitas *dataset* kurang bagus. Setelah melihat contoh data dari *dataset* Flickr 8k dan MSCOCO, ditemukan bahwa jenis dan kualitas *caption* yang ada dalam *dataset* cukup beragam. Hal ini dikarenakan banyaknya variasi gambar dan *annotator* independen yang membuat *caption*. Beberapa diantaranya dapat dikatakan baik karena *caption* dapat menjelaskan konteks gambar dengan baik. Namun terdapat juga gambar yang kurang jelas konteksnya (membingungkan) atau gambar yang cukup berkonteks namun *caption* tidak menggambarkan konteksnya. Misalkan pada gambar III.2, walaupun secara intuitif dapat diketahui bahwa gambar menunjukkan perayaan ulang tahun, tidak ada *caption* yang secara eksplisit menjelaskan hal tersebut.

Penulis mengajukan hipotesis bahwa apabila model diberikan informasi tambahan (pendukung), *caption* yang dihasilkan memiliki konteks yang lebih dalam dan berarti. Hipotesis ini sejalan dengan penelitian Park dkk. (2017) untuk menghasilkan *personalized caption* untuk keperluan *post* media sosial. Saat membuat *caption*, Park dkk menambahkan informasi tambahan berupa contoh-contoh *caption* yang sering dibuat oleh pengguna. Penelitian Park menunjukkan adanya peningkatan kualitas hasil *caption* (subjektif dan objektif).



Gambar III.2 Contoh *Dataset* MSCOCO

*Captions:*

['<start> A woman who is cutting into a cake. <end>',  
'<start> two people standing near a table with a cake <end>',  
'<start> A woman cutting into a cake with a man standing behind her <end>',  
'<start> A man and woman standing in front of a cake. <end>',  
'<start> a man and woman cut into a big cake <end>']

### III.1.2 Analisis Konteks

Dari perspektif konteks gambar dan kalimat, *caption* yang dihasilkan model sangat bergantung pada dua vektor konteks. Vektor pertama diperoleh dari *output* CNN *encoder* yang dapat diinterpretasikan sebagai 'konteks' dari gambar (mengacu pada

sub bab II.1.2.1). Model *encoder* yang umum digunakan adalah *pretrained image classification model* yang sudah dilatih dengan *dataset* sangat besar (misalkan VGG-19) dan untuk membuat model tersebut sendiri, diperlukan sumber daya memori & waktu yang besar sehingga tidak fisibel untuk dilakukan. Implikasinya, vektor pertama ini tidak dapat diubah. Walaupun semua *pretrained model* menghasilkan vektor konteks yang berbeda untuk gambar yang sama, hal ini tidak menjadi masalah karena pada akhirnya model *image captioning* dilatih untuk menyesuaikan dengan vektor konteks dari *encoder*.

Vektor konteks kedua adalah vektor dari *caption* yang dihasilkan dari komponen *word embedding* (misalnya Word2Vec). Umumnya, model *word embedding* yang digunakan adalah model *non-contextual* (mengacu sub bab II.2) karena sudah umum dan banyak digunakan. Walaupun demikian, model tersebut memiliki kelemahan karena kata-kata bermakna ganda hanya direpresentasikan dengan satu vektor yang sama. Misalkan, dalam data *caption* untuk training, terdapat *caption* "Japan has beautiful riverbanks" dan "National Bank of Singapore" memiliki vektor "bank" yang sama. Implikasinya, *caption* yang digunakan dalam data latih dan *caption* yang dihasilkan dapat saja memiliki vektor sama walaupun memiliki makna yang jauh berbeda.

Para peneliti juga menyadari adanya kekurangan tersebut dan telah banyak penelitian dilakukan dengan tujuan untuk mengembangkan pendekatan *contextualized word embedding*, diantaranya: CoVe (McCann dkk., 2017), ELMo (Peters dkk., 2018), *Cross-View Training* (Clark dkk., 2018), ULMFiT (Howard & Ruder, 2018), OpenAI GPT (Radford dkk., 2018), dan BERT (Devlin dkk., 2019) yang menjadi *state-of-the-art* saat ini.

Apabila *contextualized word embedding* dapat memahami konteks kalimat, maka penggunaan *contextualized word embedding* menggantikan *embedding* yang umum digunakan dalam *image captioning* berpotensi untuk meningkatkan kualitas *caption* yang dihasilkan, memperbaiki dan menambahkan konteks dari sisi teks (*caption* latih dan *caption* yang dibangkitkan).

## III.2 Alternatif Solusi

Untuk membuat model yang dapat menghasilkan *caption* yang lebih bermakna, maka diusulkan dua alternatif solusi dalam mengubah model, yaitu pendekatan injeksi vektor (*vector injection*) dan pendekatan mengganti *word embedding* biasa (*non-contextual*) menjadi *contextual word embedding*.

### III.2.1 Alternatif Solusi *Vector Injection*

Ide dari pendekatan ini adalah vektor konteks dari CNN *encoder* digabungkan dengan vektor lain (vektor pendukung) sebelum dimasukkan ke dalam *decoder* untuk menghasilkan *caption*. Tujuan dari penggunaan vektor pendukung ini untuk menambahkan informasi dan mengendalikan hasil pembangkitan *caption*.

Secara formal, jika  $X = \{x_0, x_1, \dots, x_n\}$  dimana  $X$  adalah kumpulan *caption* dibangkitkan,  $x$  adalah kata yang dibangkitkan pada tahapan ke -  $t$ , dan  $H_t$  adalah *hidden state decoder* pada tahap pembangkitan ke -  $t$ , maka penambahan vektor pendukung  $a$  bertujuan mengubah  $p(x_t | x_0, \dots, x_{t-1}, H_t)$  menjadi  $p(x_t | x_0, \dots, x_{t-1}, H_t, a)$ . Model *image captioning* yang semula hanya bergantung pada *caption* yang telah dibangkitkan sebelumnya ( $x_0, \dots, x_{t-1}$ ) dan *hidden state* ( $H_t$ ) dalam menentukan distribusi peluang kata yang dibangkitkan, menjadi mempertimbangkan vektor  $a$  dalam proses pembangkitan vektor.

Selain jenis vektor yang langsung berupa informasi (representasi gambar, teks), dapat juga digunakan vektor yang bersifat vektor optimasi / gradien. Tujuannya untuk memaksimalkan *log likelihood caption* semula agar mendekati kondisi yang diinginkan, seperti dalam pendekatan yang dilakukan Dathathri dkk. (2019) untuk mengontrol pembangkitan teks dari model bahasa. Untuk mendapatkan vektor gradien ini, maka perlu pembandingan yang dapat digunakan untuk mengukur seberapa dekat *caption* yang dihasilkan dengan *caption* yang diinginkan. Model atau referensi pembandingan ini disebut model atribut (*attribute model*).

Vektor gradien umumnya didapatkan dari perhitungan turunan fungsi *loss decoder* model *image captioning*, terhadap error dari *output* model atribut. Vektor gradien yang dimaksud sama dengan gradien yang umumnya digunakan pada proses *stochastic gradient descent* saat melatih model.

Secara formal, dalam memaksimalkan  $p(x_t | x_1, \dots, x_{t-1}, H_t, a)$ , vektor  $a$  yang digunakan adalah vektor yang memaksimalkan *log likelihood caption*  $x_t$  sehingga  $D(x_t) \approx c$  dimana *output* / hasil prediksi model atribut  $D$  mendekati kondisi *caption* / target  $c$  yang diinginkan; dengan kata lain, vektor yang memaksimalkan  $p(c | D(x_t))$ .

Pada kasus penggunaan vektor gradien, model memaksimalkan  $p(x_t | a)$  dengan cara menggeser distribusi keluaran agar mendekati distribusi yang diinginkan dengan bantuan umpan balik (*feedback*) dari model atribut. Pergeseran distribusi dapat dilakukan beberapa kali tergantung arsitektur model. Proses pergeseran ini menyerupai proses *stochastic gradient descent* saat melatih model *neural network* pada umumnya.

### III.2.1.1 Jenis Vektor yang digunakan

Dalam memilih jenis vektor yang digunakan, untuk kategori vektor yang bersifat informasi, merujuk pada penelitian serupa yang dilakukan oleh Park dkk. (2017) dan Sow dkk. (2019). Park dkk. menggunakan vektor konteks dari pengguna (vektor yang merepresentasikan *caption* yang sering dibuat oleh pengguna) yang digabungkan dengan vektor konteks gambar. Dalam penelitian Sow dkk. digunakan *hidden state* suatu LSTM sebagai '*guiding vector*' (terobosan yang diusulkan untuk meningkatkan kualitas *caption*) dengan cara menggabungkan *guiding vector* dengan vektor konteks dari gambar. Contoh lainnya dapat menggunakan vektor gambar pendukung, representasi teks yang menceritakan konteks, atau *control code* seperti yang dilakukan Keskar dkk. (2019).

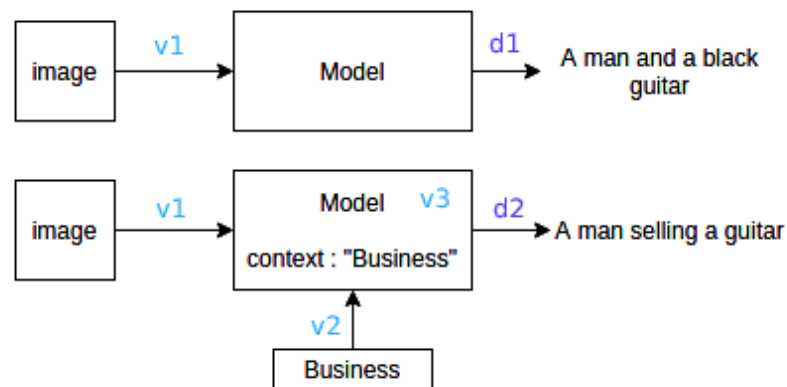
Salah satu kelemahan dari menggunakan vektor yang bersifat informasi adalah model *image captioning* harus dipersiapkan dari awal untuk menerima informasi tambahan. Karena model melakukan optimasi  $p(x_t | x_1, \dots, x_{t-1}, H_t, a)$  dilakukan untuk vektor  $a$  yang spesifik pada tahap pelatihan, maka implikasinya,  $a$  yang digunakan perlu ditetapkan dari awal. Hal ini ditunjukkan oleh Keskar dkk. (2019) dengan mempersiapkan dulu sejumlah *control code* dan *dataset* yang disesuaikan dengan masing-masing jenis *control code*. *Dataset* yang digunakan juga harus



diberi label untuk menandakan konteks dari setiap *caption*, menyesuaikan dengan jenis konteks yang ingin diakomodasi oleh model *image captioning*.

Disisi lain, kelebihan dari penggunaan vektor yang bersifat informasi adalah kualitas *caption* yang dihasilkan seharusnya lebih baik, karena model *captioning* memang dilatih untuk memaksimalkan  $p(x_t | a)$  dengan  $a$  yang spesifik.

Sebagai ilustrasi (gambar III.3), jika ingin membangkitkan *caption* yang memiliki konteks bisnis, maka perlu ditambahkan vektor yang merepresentasikan 'bisnis' pada setiap tahap pembangkitan, dan menambah *caption* dalam data latih yang menunjukkan konteks 'bisnis'.



Gambar III.3 Ilustrasi Penggunaan Vektor Informasi

Ilustrasi pada gambar III.3, tahap pembangkitan kata '*selling*', diperjelas oleh tabel III.1 dan III.2. Proses penggabungan vektor (v3) dijelaskan lebih lanjut pada sub bab III.2.2.

**Tabel III.1 Contoh Nilai Vektor pada Gambar III.3**

v1	gambar	0.09	0.13	...	-0.21	0.04
v2	representasi konteks ' <i>business</i> '	-0.12	0.11	...	0.38	-0.01
v3	penggabungan (penjumlahan)	-0.03	0.24	...	0.17	0.03
	penggabungan (konkatenasi gambar-konteks)	0.09	0.13	...	0.38	-0.01

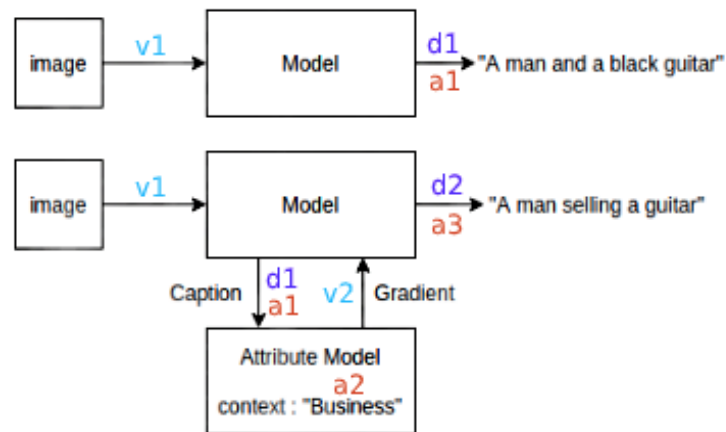
**Tabel III.2 Contoh Nilai Vektor Distribusi pada Gambar III.3**

	Distribusi	<i>'and'</i>	<i>'playing'</i>	...	<i>'selling'</i>	<i>'with'</i>
d1	awal	<b>0.43</b>	0.21	...	0.01	0.22
d2	setelah ditambahkan konteks <i>'business'</i>	0.20	0.11	...	<b>0.34</b>	0.11

Penggunaan jenis vektor yang bersifat optimasi / gradien mengacu pada penelitian Dathathri dkk. (2019). Salah satu metode dilakukan dengan cara membandingkan *caption* (yang dibangkitkan) dengan kumpulan kata BoW (*bag of words*) yang diinginkan. Metode lainnya dapat juga memanfaatkan model diskriminator yang membandingkan kategori / kelas dari *caption* yang dibangkitkan dengan kategori / kelas yang diinginkan, misalnya dari segi jenis kalimat (netral / ulasan / berita), sentimen (netral / positif / negatif) dan sebagainya.

Pada model atribut BoW, maka gradien didapatkan sebagai turunan dari *loss* yang didapatkan dengan membandingkan distribusi kata (keluaran model *image captioning*) dengan BoW yang digunakan. Pada model atribut yang berupa diskriminator, *loss* didapatkan dengan membandingkan hasil prediksi model atribut dengan keluaran yang diharapkan. Fungsi *loss* yang digunakan bergantung pada arsitektur model *image captioning* yang digunakan, umumnya berupa *categorical cross entropy* pada *feedforward layer* terakhir model *image captioning* (bukan model atribut). Skema detail dapat dilihat pada sub bab III.4.4. dan sub bab III.4.5.

Sebagai ilustrasi (gambar III.4), dengan contoh yang sama dengan ilustrasi sebelumnya, jika ingin membuat model *image captioning* yang dapat menghasilkan *caption* berkonteks 'bisnis', maka digunakan model atribut yang mampu mengenali konteks 'bisnis' (diskriminator atau BoW), yang fungsinya membandingkan *output caption* asli dengan *output* yang diinginkan. vektor yang ditambahkan pada *decoder* adalah vektor gradien dari model atribut tersebut. Ilustrasi pada gambar III.4, tahap pembangkitan kata *'selling'*, diperjelas oleh tabel III.3, III.4 dan III.5.



Gambar III.4 Ilustrasi Penggunaan Vektor Gradien

Tabel III.3 Contoh Nilai Vektor pada Gambar III.4

v1	gambar	0.09	0.13	...	-0.21	0.04
v2	gradien	-0.12	0.11	...	0.38	-0.01

Tabel III.4 Contoh Nilai Distribusi pada Gambar III.4

	Distribusi	<i>'and'</i>	<i>'playing'</i>	...	<i>'selling'</i>	<i>'with'</i>
d1	awal	<b>0.43</b>	0.21	...	0.01	0.22
d2	setelah ditambahkan konteks <i>'business'</i>	0.20	0.11	...	<b>0.34</b>	0.11

Tabel III.5 Contoh Nilai Keluaran Model Atribut pada Gambar III.4

	Prediksi Model Atribut	<i>'food'</i>	<i>'business'</i>	...	<i>'hobby'</i>	<i>'education'</i>
a1	awal	0.02	0.11	...	<b>0.52</b>	0.05
a2	yang diharapkan	0	<b>1</b>	...	0	0
a3	setelah 'diperbaiki'	0.01	<b>0.42</b>	...	0.27	0.03

### III.2.1.2 Metode Penggunaan Vektor

Penulis mengusulkan dua alternatif cara penggunaan vektor pendukung, yang pertama menggunakan pendekatan konkatenasi seperti yang dilakukan Park dkk. (2017) dan Sow dkk. (2019); dan yang kedua menggunakan pendekatan *weighted average* seperti yang dilakukan (Vaswani dkk., 2017) dengan mencari ‘rata-rata’ dari vektor utama dan vektor pendukung.

Pendekatan konkatenasi vektor dapat dipandang sebagai cara untuk memberi informasi tambahan kepada *decoder*, dalam bentuk fitur tanpa perlu mengubah vektor utama. Kelebihan pendekatan ini adalah mudah diimplementasikan dan model *decoder* mendapatkan kedua informasi, baik vektor asli dan vektor pendukung. Baik vektor yang bersifat informasi maupun gradien dapat menggunakan pendekatan ini.

Pada pendekatan *weighted average*, vektor utama dirata-ratakan dengan vektor pendukung sesuai bobot masing-masing. Proses ini cukup intuitif untuk vektor yang bersifat gradien karena dapat diinterpretasikan sebagai proses ‘menyesuaikan’ vektor *decoder* agar kata yang diprediksi lebih mendekati kondisi yang diinginkan.

Pada vektor yang bersifat informasi, pendekatan *weighted average* dapat memunculkan perilaku model yang tidak terduga. Hal tersebut dapat terjadi karena adanya perubahan langsung pada vektor konteks gambar. Penggabungan dua jenis informasi yang berbeda dapat menghasilkan hasil yang kurang sesuai karena kedua vektor tidak lagi memiliki makna dan nilai yang seharusnya.

Sebagai contoh, vektor ‘gambar gajah’ jika ditambahkan vektor konteks ‘berenang’ bisa saja mendekati nilai dari vektor ‘gambar kuda’. Hal tersebut menyebabkan model membangkitkan *caption* yang tidak relevan.

### III.2.1.3 Dampak Penambahan Vektor Gradien

Pada kasus penggunaan vektor gradien, karena model *image captioning* tidak secara langsung dilatih untuk memaksimalkan  $p(x_t | a)$ , maka hasil keluaran sudah pasti tidak sebagus jika model dilatih secara spesifik untuk memaksimalkan  $p(x_t | a)$ .

*Decoder* hanya bisa bergantung pada  $a$  dan model atribut yang bertugas mengarahkan agar *decoder* menghasilkan *caption* yang sesuai keinginan. Oleh karena itu, bisa terjadi beberapa kasus / kondisi tergantung seberapa besar perubahan distribusi kata yang diakibatkan oleh vektor  $a$ :

1. Kondisi terbaik: perubahan distribusi menghasilkan *caption* yang lebih baik dan sesuai keinginan.

contoh: "A man and a guitar" -> "A man selling a guitar"

2. Kondisi perubahan terlalu besar: hasil *caption* mencerminkan konteks sesuai keinginan, namun *caption* tidak lagi relevan.  
contoh: "A man and a guitar" -> "A selling selling guitar selling"

3. Kondisi perubahan terlalu kecil: *caption* tidak berhasil mencerminkan konteks, namun masih baik dan relevan.

contoh: "A man and a guitar" -> "A man and his guitar"

4. Kondisi terburuk: *caption* tidak lagi relevan dan tidak menampilkan konteks. "A man and a guitar" -> "A hippo lion black marshal"

### III.2.2 Alternatif Solusi Menggunakan *Contextualized Word Embedding*

Seperti dijelaskan sebelumnya (mengacu sub bab II.2), secara teori penggunaan *contextual word embedding* menghasilkan vektor yang lebih bermakna, karena masing-masing kata diberikan vektor berbeda bukan hanya menurut sintaksis (bentuk dan urutan kata), namun juga berdasarkan semantik penggunaannya. Jika model *word embedding* yang digunakan dapat menangkap konteks kalimat, implikasinya model *captioning* juga dapat menangkap konteks dengan baik. Harapannya saat proses pembangkitan *caption*, tidak lagi ditemukan permasalahan homograf atau kesalahan penggunaan kata yang tidak sesuai konteksnya. Sebagai contoh, jika kata yang sebelumnya dibangkitkan adalah 'bank', maka kata berikutnya yang dibangkitkan dapat menyesuaikan, apakah konteksnya 'bank' sebagai perusahaan, gedung, atau 'bank' sebagai bagian dari 'riverbank' (pinggiran sungai). Jika kata 'bank' yang digunakan merujuk perusahaan, maka seharusnya

kata yang relatif dekat adalah '*money*' (uang), namun jika kata '*bank*' yang digunakan adalah untuk *riverbank*, maka kata '*fish*' kemungkinan lebih relevan.

### III.3 Pemilihan Alternatif

Dalam penelitian ini digunakan pendekatan vektor pendukung dan pendekatan *contextualized word embedding*. Berdasarkan hipotesis, penggunaan *contextual word embedding* meningkatkan kualitas *caption*, baik dalam keadaan berkonteks maupun tidak. Untuk detail implementasi, dipilih beberapa pendekatan berdasarkan hasil analisis dan studi literatur.

#### III.3.1 Pemilihan Solusi

Alternatif penambahan vektor yang dipilih mengadopsi teknik yang digunakan oleh Dathathri dkk. (2019), yaitu menggunakan vektor pendukung berupa gradien dari model atribut, yang kemudian dijumlahkan dengan vektor utama dengan perbandingan tertentu. Pendekatan yang dilakukan Dathathri dkk. unggul karena model PPLM fleksibel terhadap model bahasa yang menjadi basis modelnya dan juga model atributnya, walaupun hasilnya tidak bisa sebagus model CTRL Keskar, dkk. (2019).

Model CTRL memiliki hasil yang lebih baik karena model tersebut dilatih untuk memaksimalkan probabilitas untuk model atribut (*control code*) tertentu yang sudah ditetapkan dari awal. Dalam kasus *image captioning*, jenis konteks sebaiknya dapat disesuaikan / diganti kemudian hari. Apabila menggunakan pendekatan seperti CTRL, maka model harus dilatih kembali untuk berbagai model atribut yang digunakan dan menjadi tidak fleksibel. Keterbatasan *dataset* yang merepresentasikan masing-masing konteks juga menjadi salah satu pertimbangan untuk tidak menggunakan model *image captioning* yang terikat dengan model atributnya.

Sebaliknya dengan pendekatan seperti PPLM Dathathri dkk., (2019), model *image captioning* yang dibangun terpisah dari model atributnya, artinya jika diperlukan berbagai macam konteks yang berbeda, maka yang perlu dilatih ulang hanya bagian model atributnya saja. Model utama *image captioning*nya tidak perlu dilatih ulang.

Hal ini yang menjadikan pendekatan Dathathri dkk. unik dan menarik untuk dicoba penerapannya dalam *task image captioning*.

Penulis mengetahui adanya risiko bahwa penggunaan vektor gradien dapat membuat model menghasilkan kata-kata yang tidak sesuai seperti yang sebelumnya disebutkan pada bagian analisis. Namun masalah tersebut dapat ditangani dengan cara penggunaan variabel bobot yang dapat diatur, dimana proses penjumlahan antara vektor konteks utama dan vektor pendukung dikalikan dengan suatu nilai  $\gamma$  yang bernilai antara 0 sampai 1.

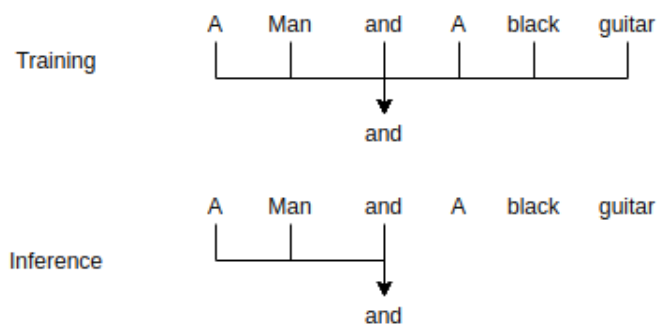
Penggunaan variabel ini juga memungkinkan pengguna untuk bebas mengatur tingkat perubahan yang dilakukan oleh vektor pendukung. Apabila nilai  $\gamma$  diatur mendekati 0, maka tidak ada dampak dari penambahan vektor pendukung, sebaliknya jika  $\gamma$  mendekati 1, vektor pendukung sepenuhnya menggantikan posisi vektor utama. Walaupun demikian, tidak dapat dijamin adanya suatu konstanta  $\gamma$  yang optimal untuk semua kasus, dan nilai  $\gamma$  yang optimal hanya bisa didapatkan melalui eksperimen untuk setiap model atribut yang digunakan.

Menggunakan pendekatan ini, secara hipotesis pengguna dimungkinkan untuk menambahkan lebih dari satu konteks pada satu waktu. Hal ini dilakukan dengan menggunakan beberapa model atribut yang berbeda dan variabel tambahan untuk mengatur bobot dari masing-masing model atribut. Apabila digunakan dua model atribut yang selaras (saling mendukung) atau tidak saling terkait, maka hasilnya akan serupa dengan satu model atribut yang memaksimalkan kedua atribut tersebut. Sebaliknya jika menggunakan dua model atribut yang bertolak belakang, maka keduanya akan saling menghilangkan efek pergeseran distribusi dan model *image captioning* akan kembali menggunakan distribusi awal. Kasus ini berlaku dengan asumsi bahwa untuk atribut yang memiliki sifat bertolak belakang, akan direpresentasikan dengan vektor yang juga bertolak belakang oleh model *image captioning*.

### **III.3.2 Pemilihan Alternatif *Contextualized Word Embedding***

Beberapa *contextualized word embedding* yang dipertimbangkan untuk digunakan adalah BERT (Devlin dkk., 2019) dan OpenAI GPT-2 (Radford dkk., 2019) (serta

turunannya). Walaupun kedua model ini mendapatkan popularitas karena kinerjanya, kedua model ini memiliki keunggulan dan kekurangan masing-masing. Berkat sifat *bidirectional*-nya, BERT mampu menangkap konteks suatu kata dengan lebih baik, karena mempertimbangkan baik kata sebelum maupun kata sesudahnya. Disisi lain, GPT-2 tidak bersifat *bidirectional*, dan hanya menangkap konteks dari kiri ke kanan, oleh karena itu, sering kali dalam penelitian yang banyak yang menyebutkan BERT lebih unggul untuk menangkap representasi suatu kalimat. Sayangnya, sifat *bidirectional* BERT ini juga menjadi kelemahan BERT untuk kasus-kasus tertentu yang bersifat *autoregressive*, salah satunya *task image captioning*. Dalam *task image captioning*, umumnya kata yang telah dibangkitkan kembali dimasukkan untuk membangkitkan kata berikutnya dan proses pembangkitan bersifat satu arah. Hal ini menyebabkan representasi dua arah BERT menjadi 'pengganggu' yang dapat mengacaukan proses pembangkitan karena memberikan *word embedding* yang tidak 'utuh'. Sebagai contoh (gambar III.5), saat training BERT merepresentasikan kata '*and*' berdasarkan lima kata lainnya, namun saat pembangkitan BERT hanya mempertimbangkan dua kata karena *caption* belum selesai. Hal ini menyulitkan *decoder* karena menerima representasi kalimat yang berbeda walaupun kalimat sebenarnya sama.



Gambar III.5 Ilustrasi Permasalahan BERT saat Pembangkitan *Caption*

Walaupun kemampuan GPT-2 merepresentasikan kalimat hanya terbatas satu arah, namun sesuai dengan sifat *task image captioning* yang memang dilakukan satu arah.



Penulis tidak berhasil menemukan literatur yang dengan tegas menyatakan BERT tidak dapat digunakan untuk *task image captioning*. Untuk membuktikan hipotesis yang disebutkan, maka penelitian dilakukan untuk kedua model *contextual word embedding*.

### **III.3.3 Pemilihan Alternatif *Inject / Merge* (model sekuensial)**

Dalam tahap eksplorasi model *image captioning*, kedua pendekatan telah dicoba dan memiliki hasil yang relatif sama saat diujikan dengan metrik BLEU. Dalam penelitian ini digunakan pendekatan *merge* dimana vektor gambar tidak dimasukkan ke dalam model sekuensial. Hal ini dilakukan untuk meningkatkan efisiensi dari segi pemakaian sumber daya. Dengan menggunakan pendekatan *merge*, ukuran vektor yang diterima model sekuensial berkurang dan jumlah parameter yang perlu dilatih menjadi lebih sedikit.

### **III.3.4 Pemilihan Alternatif Komponen *Image Encoder***

Komponen terakhir yang perlu dipertimbangkan adalah *pretrained model* pada komponen *encoder* (mengubah gambar menjadi vektor konteks). Seperti sebelumnya pernah disebutkan (sub bab II.1.2.1), salah satu komponen penting dalam model *image captioning* adalah *encoder* itu sendiri. Saat ini terdapat banyak model *image encoder* yang berasal dari *layer* terakhir model klasifikasi gambar (*image classification model*), misalnya Xception, InceptionV3, ResNet, VGG19, dll. dan hampir semuanya memiliki *open source library* yang dapat langsung digunakan (misalnya terdapat pada Keras).

Keras menyediakan model-model pralatih yang dapat langsung digunakan, diantaranya Xception, VGG16, VGG19, ResNet, ResNetV2, InceptionV3, InceptionResNetV2, MobileNet, MobileNetV2, DenseNet, dan NASNet. Berdasarkan pendapat umum praktisi, didukung penelitian Bianco dkk. (2018) mengenai *benchmarking* berbagai model, serta dokumen resmi dari Keras mengenai spesifikasi model (gambar III.6), dipilihlah model Xception untuk digunakan dalam penelitian karena model tersebut ringan dan kinerjanya bersaing dengan model-model besar lainnya.

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-

Gambar III.6 Tabel *Benchmark Model* (Keras)

Selain statistik pada gambar III.6, Xception pralatih yang digunakan memiliki spesifikasi tambahan sebagai berikut:

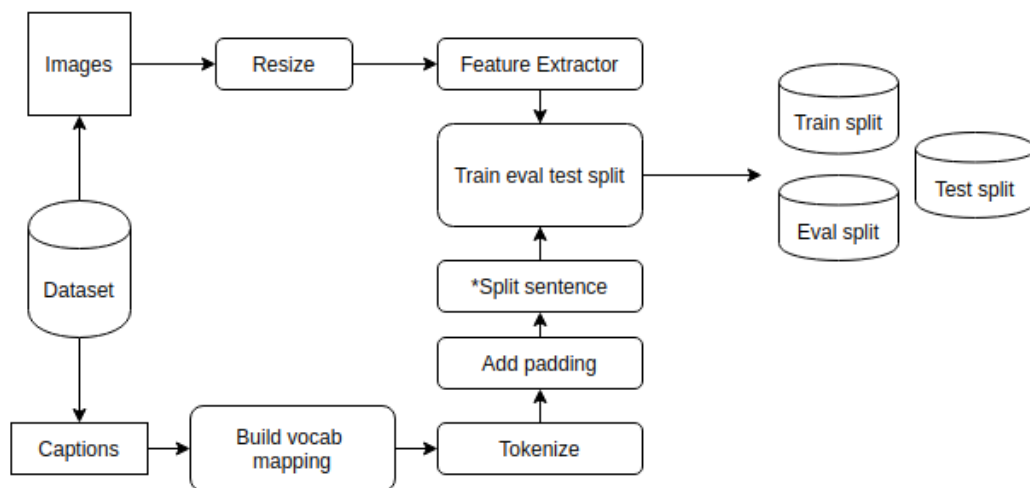
1. Dimensi masukan:  $299 \times 299 \times 3$
2. Dilatih pada dataset *ImageNet*
3. Jumlah kelas: 1000 (*ImageNet*)

### III.4 Rancangan Solusi

Alternatif solusi yang telah diajukan dan dipilih selanjutnya disusun menjadi sebuah sistem yang terpadu.

#### III.4.1 Skema Praproses *Dataset*

Rangkaian praproses *dataset* (gambar III.7) dibagi menjadi dua kategori: gambar dan teks. Semua gambar dalam *dataset* diseragamkan ukurannya, lalu dimasukkan dalam model CNN *encoder* untuk mendapatkan vektor representasi gambar. Model CNN *encoder* yang dipilih adalah Xception karena model Xception ringan dan kinerjanya bersaing dengan model besar lainnya (VGG, Inception, dll).

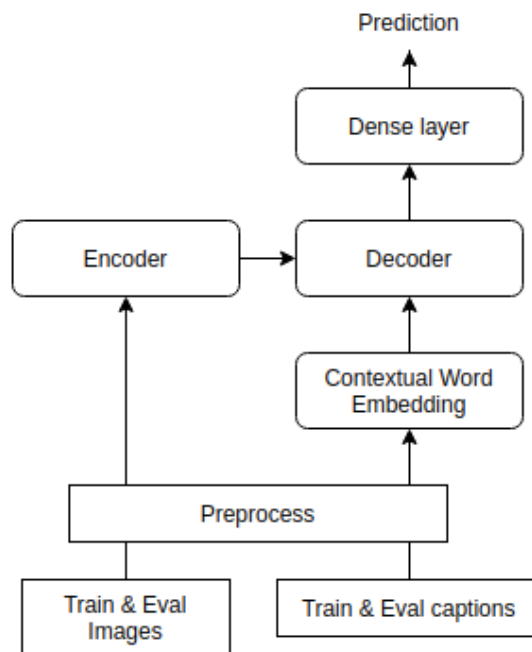


Gambar III.7 Gambar Skema Pemrosesan *Dataset*

Praproses teks *caption* dalam *dataset* terdiri dari beberapa tahap. Pertama-tama dibangun sebuah *mapping* (korespondensi 1-1) antara id *vocab* yang digunakan oleh model *captioning* dan id *vocab* yang digunakan model *contextual word embedding*. Hal ini dilakukan karena kosakata dari BERT / GPT-2 ukurannya tergolong besar (ukuran kosakata BERT: 30k~ dan GPT-2: 52k~) sehingga perlu dikecilkan untuk menghemat memori. Proses ini dilakukan dengan cara mengubah semua *caption* menjadi representasi token yang dikenali model *embedding*, lalu menghitung dan mengambil *top-k* token yang paling sering muncul dalam *dataset*, sisanya diubah menjadi token [OOV]. Selanjutnya ditambahkan *padding* untuk menyeragamkan panjang sekuens. Lalu jika diperlukan, dapat memecah satu kalimat menjadi  $n$  kalimat. misalnya: A B C dipecah menjadi A [pad] [pad], A B [pad], dan A B C. Tujuannya untuk melatih model *captioning* yang menebak satu kata setiap waktunya (bukan berbasis *transformer*). Setelah itu gambar dan *caption* dibagi menjadi 3 bagian, untuk data latih, evaluasi, dan uji.

#### III.4.2 Skema Pelatihan Model

Setelah praproses data dilakukan, proses pelatihan model *captioning* memerlukan beberapa komponen seperti ditunjukkan pada gambar III.8:



Gambar III.8 Gambar Skema Pelatihan Model

1. *Encoder*: digunakan untuk menerima vektor gambar. Karena gambar sudah dalam representasi vektor, sebenarnya *encoder* bersifat opsional. Umumnya hanya terdiri dari satu *fully connected layer* disertai fungsi aktivasi linear (misal ReLU).
2. *(Contextual) Word embedding*: digunakan untuk menerima token *caption*. Komponen *positional encoding* yang merupakan bagian dari *word embedding* juga diperlukan untuk model berbasis *transformer*. *Positional encoding* berfungsi untuk merepresentasikan informasi posisi suatu kata. Karena model *transformer* tidak lagi memiliki sifat rekuren, maka informasi urutan kata perlu direpresentasikan secara eksplisit. Dalam penelitian, model yang digunakan adalah BERT dan GPT-2. Proses pembuatan dan pelatihan model BERT dan GPT-2 dari awal tidak feasible dilakukan sehingga dalam penelitian digunakan model prelatih.
3. *Decoder*: bertugas mengeluarkan *output* yang merepresentasikan *state* dari *caption* berdasarkan *output encoder* dan *word embedding*. *Decoder* dapat

berupa model rekuren (RNN, LSTM, GRU, dll) maupun tumpukan *layer decoder transformer*. *Attention mask* diperlukan *decoder* berbasis *transformer* untuk mencegah model menggunakan informasi kata ke  $n + 1$  dan seterusnya saat mempelajari kata ke- $n$ .

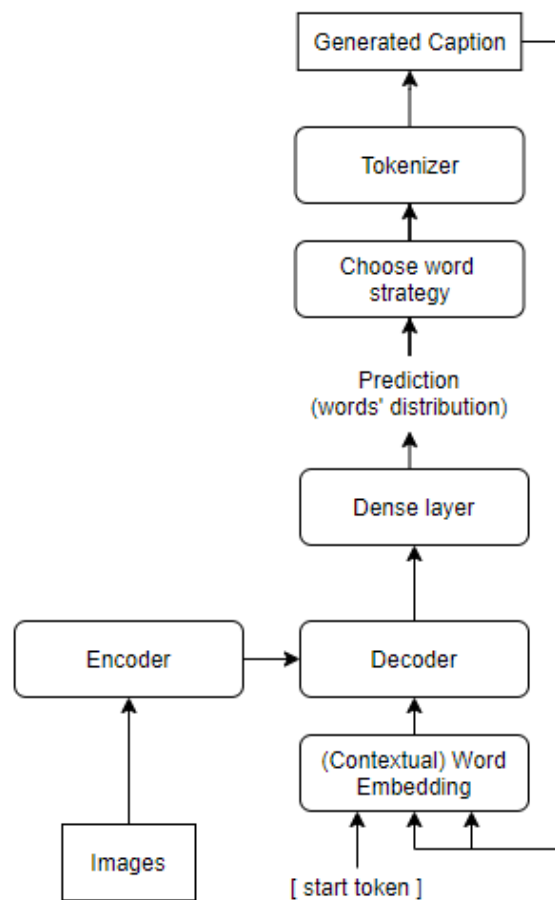
4. *Dense layer*: komponen ini bertugas menghasilkan distribusi probabilitas kata berdasarkan *output decoder*. *Dense layer* memiliki dimensi sesuai ukuran kosakata yang digunakan, dan umumnya *output* dari *dense layer* ini dilewatkan fungsi *softmax* untuk langsung mendapatkan probabilitas setiap kata. Selanjutnya probabilitas setiap kata dibandingkan dengan kata yang seharusnya dibangkitkan oleh *loss function*.
5. *Loss function*: komponen terakhir yang diperlukan untuk menilai hasil prediksi. Umumnya dalam *task image captioning*, digunakan *loss Categorical Cross Entropy* karena pada dasarnya *task image captioning* termasuk *task* klasifikasi dimana terdapat banyak jenis kelas.

### III.4.3 Skema Pembangkitan *Caption*

Skema pembangkitan *caption* (gambar III.9) relatif mirip dengan proses pelatihan model. Namun proses pembangkitan *caption* umumnya dilakukan secara *autoregressive*. Artinya kata yang menjadi *output* pada tahap  $t$  kembali dimasukkan ke dalam model untuk mendapatkan kata ke  $t + 1$ , dan terus diulang sampai kata terakhir berhasil dimunculkan.

Sebagai contoh:

<i>input</i> : [Start] [pad] ...,	<i>output</i> : A
<i>input</i> : [Start] A [pad] ...	<i>output</i> : brown
<i>input</i> : [Start] A brown [pad] ...	<i>output</i> : dog
...	
<i>input</i> : [Start] A brown dog on the field [pad] ...	<i>output</i> : [end]



Gambar III.9 Gambar Skema Pembangkitan *Caption*

Salah satu komponen penting pada proses pembangkitan *caption* adalah modul untuk memilih kata yang dibangkitkan. Komponen *dense layer* dari *decoder* hanya menghasilkan probabilitas dari kata yang dibangkitkan berikutnya, namun bagaimana strategi memilih kata dari distribusi tersebut bukan merupakan tanggung jawab *dense layer*. Terdapat beberapa strategi yang umum digunakan, diantaranya:

#### 1. *Greedy*

Pendekatan *greedy* berarti selalu memilih kata yang memiliki nilai probabilitas tertinggi dari distribusi yang dihasilkan. Pendekatan ini dalam literatur umumnya menjadi *baseline* karena sifatnya cepat dan selalu mengeluarkan keluaran yang konsisten.

## 2. *Top-k sampling*

Pendekatan ini dilakukan dengan memilih  $k$  kata yang memiliki nilai probabilitas tertinggi, lalu dilakukan *sampling* berdasarkan probabilitas masing-masing kata. Pendekatan ini pada umumnya jarang dipakai karena walaupun tujuannya menambah variasi kata, seringkali kata yang dipilih tidak terlalu sesuai dengan kata sebelumnya dan hanya terpilih karena faktor acak.

Sebagai contoh: misal dari dua kata tertinggi ( $k = 2$ ) didapatkan probabilitas (A, 0.5) dan (B, 0.25). Maka saat memilih kata yang dibangkitkan, kata A memiliki probabilitas 2x lebih besar untuk dipilih dibandingkan B, karena probabilitas A 2x probabilitas B.

## 3. *Beam search*

Pendekatan *beam search* umumnya menghasilkan kualitas *caption* paling baik dengan risiko meningkatkan waktu komputasi yang diperlukan secara signifikan. *Beam search* memanfaatkan ekspansi graf dengan pendekatan *best first search* untuk mencari kandidat sekuens *caption* terbaik. Pada setiap tahap ekspansi, algoritma hanya mempertahankan  $k$  kandidat terbaik untuk diteruskan pencarian.

### III.4.4 Skema Persiapan Model Atribut

Dengan mengadopsi pendekatan yang dilakukan Dathathri dkk. (2019), model *image captioning* yang dibangun pada setiap tahap pembangkitan kata dapat dikendalikan proses pembangkitannya menggunakan sebuah 'model' atribut. Model atribut yang digunakan dalam penelitian berupa kumpulan kata (*bag of words*). Sebelum dapat digunakan, model atribut yang berbasis *bag of words* (BoW) perlu dilewatkan serangkaian praproses yang ditunjukkan pada gambar III.10:

1. Setiap kata dalam BoW diubah menjadi bentuk token menggunakan *tokenizer* yang sama untuk mengubah *caption* menjadi token.
2. Selanjutnya untuk setiap kata, apabila token kata tersebut adalah token di luar kosakata ([UNK]), maka kata tersebut dibuang dari BoW. Apabila hasil tokenisasi berupa beberapa sub token, dan terdapat sub token [UNK], maka kata tersebut juga dibuang dari BoW.

3. Token yang tersisa dibuat representasi *one hot encoding* sesuai posisi indeks token tersebut.

Contoh: token tersisa adalah [3, 5], maka dibuat menjadi  $[[0,0,0,1,0, 0,...], [0,0,0,0,0,1,...]]$

4. Hasil *one hot encoding* digabungkan menjadi satu representasi BoW dengan cara dijumlahkan.

Contoh:  $[[0,0,0,1,0,0,...], [0,0,0,0,0,1,...]]$  dijumlahkan menjadi  $[0,0,0,1,0,1,...]$ .



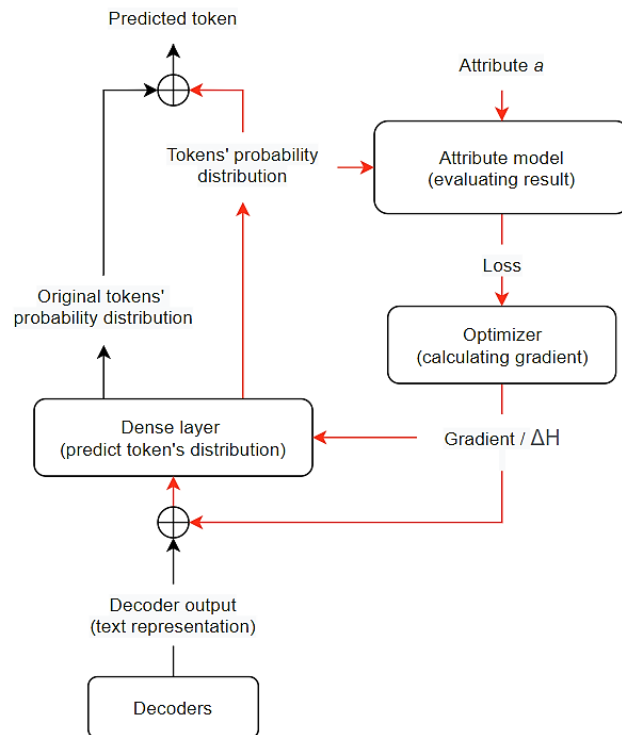
Gambar III.10 Gambar Skema Persiapan Model Atribut

#### III.4.5 Skema Pembangkitan *Caption* dengan Konteks

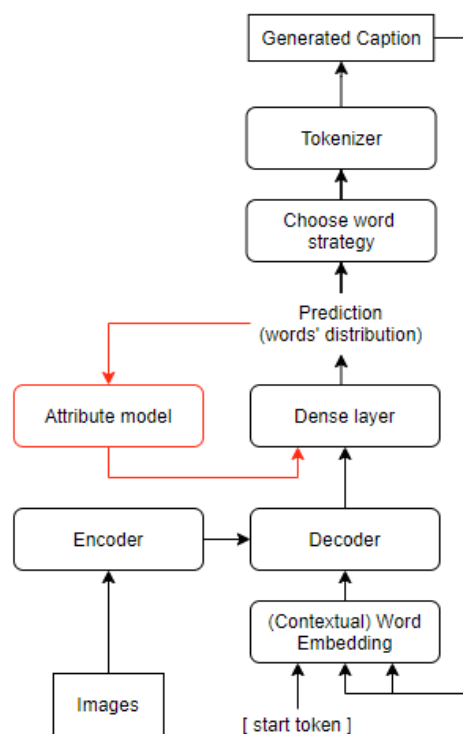
Model atribut yang telah dipersiapkan digunakan untuk membandingkan *output caption* yang dihasilkan dengan *output* yang diharapkan sesuai dengan model atribut. Susunan peletakan model atribut dalam keseluruhan sistem ditunjukkan pada gambar III.11.

Dalam penelitian, digunakan *categorical cross entropy loss* untuk membandingkan distribusi probabilitas kata yang dihasilkan *dense layer* terakhir dengan vektor kumpulan kata (BoW) yang diinginkan. Perbedaan antara distribusi probabilitas kata yang dihasilkan *dense layer* dengan distribusi BoW dihitung sebagai *loss* yang gradiennya ditambahkan sebagai vektor pendukung *a* bagi vektor *decoder*. Alternatif lainnya dapat juga menggunakan vektor gradien terhadap *hidden state (H)* *dense layer* dan ditambahkan bersamaan dengan vektor pendukung *a*. Proses pengubahan distribusi kata ini dapat dilakukan beberapa iterasi sesuai parameter yang ditetapkan oleh pengguna, lebih jelasnya ditunjukkan pada gambar III.12.





Gambar III.11 Gambar Skema Proses Pengubahan Distribusi Kata

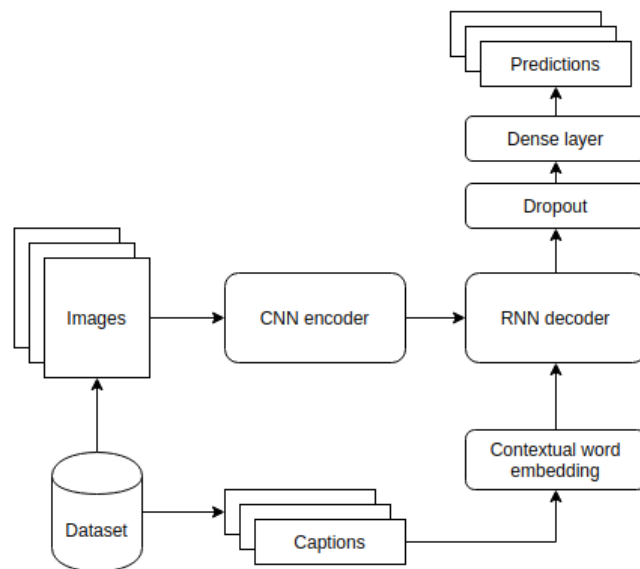


Gambar III.12 Gambar Skema Pembangkitan *Caption* Berkonteks

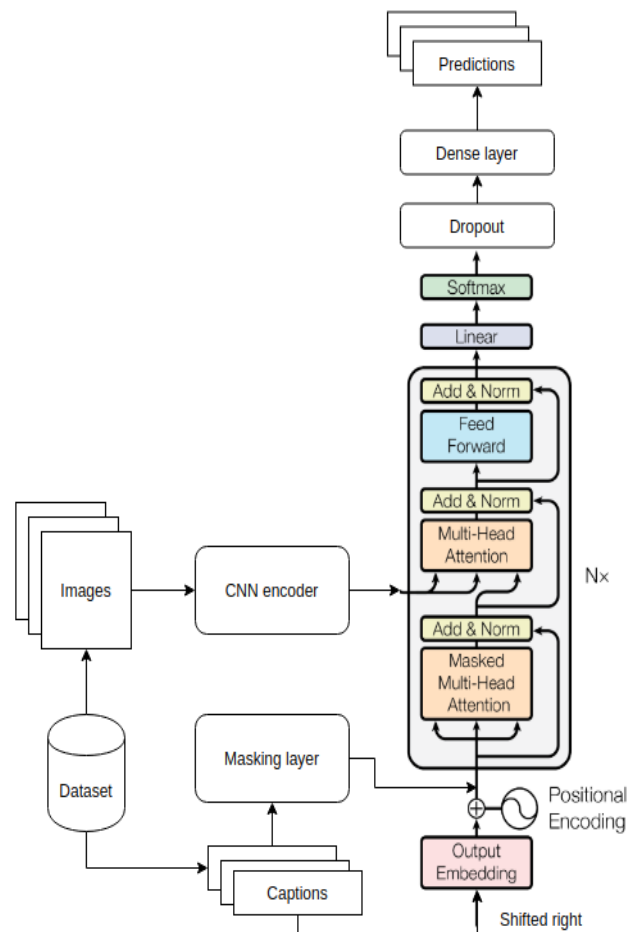
Sebagai contoh, misalkan  $d$  adalah *output decoder*,  $d = [0.31, -2.91, \dots, 5.21, 0.04]$  yang dilewatkan pada *dense layer* dan menghasilkan  $x$ , dimana  $x$  adalah distribusi kata sesungguhnya,  $x = [0.1, 0.3, 0.5, 0.1]$ . Saat  $x$  dibandingkan dengan vektor BoW  $y$  dengan  $y = [0, 1, 0, 1]$ , didapatkan nilai gradien  $a$  yang bernilai  $[-0.21, 0.22, \dots, -0.03, 0.05]$ . Selanjutnya  $a$  ditambahkan pada *output decoder*  $d$  untuk memaksimalkan *log likelihood*  $x$  agar menyerupai  $y$ , sehingga  $d' = d + a = [0.1, -2.69, \dots, 5.18, 0.09]$ . Ketika  $d'$  dilewatkan kembali ke *dense* model untuk mendapatkan  $x'$ , dimana  $x' = \text{dense}(d')$ , maka  $\text{loss}(x', y) < \text{loss}(x, y)$ .

#### III.4.6 Detail Arsitektur Model

Melihat perkembangan model *image captioning* beberapa tahun terakhir, kecenderungan model *image captioning* mengikuti kerangka kerja yang dicetuskan Kiros dkk. (2014) dan Xu dkk. (2016) yang terdiri dari komponen *encoder* dan *decoder*. Berikut rancangan arsitektur model *image captioning* berbasis model rekuren (gambar III.13) dan *Transformer* (Vaswani dkk., 2017) (gambar III.14). Komponen *decoder* berbasis rekuren memanfaatkan model rekuren seperti RNN, LSTM, dan GRU untuk menangkap informasi sekuens dari *caption*, sedangkan *decoder* berbasis *transformer* mengadopsi model *decoder* yang dikembangkan Vaswani dkk. (2017).



Gambar III.13 Diagram Arsitektur Berbasis Model Rekuren



Gambar III.14 Diagram Arsitektur Model Berbasis *Transformer*

### III.5 Rancangan Penelitian

#### III.5.1 Dataset

Dalam penelitian, digunakan dua *dataset* publik yaitu Flickr 8K dan MSCOCO. *Dataset* tersebut terdiri dari gambar (foto) yang digunakan untuk membangkitkan *caption* dan masing-masing gambar memiliki minimal lima *caption* berbeda. *Dataset* Flickr 8k terdiri dari ~8000 gambar unik dan *dataset* MSCOCO terdiri dari ~82500 gambar unik. Kedua *dataset* dibagi menjadi empat bagian (*split*) yang terdiri dari data latihan (*train*), data evaluasi (*eval*) dan data uji (*test*).

### III.5.2 Metrik Penilaian

Dalam penelitian digunakan dua macam metrik penilaian, objektif dan subjektif. Metrik objektif meliputi metrik BLEU, METEOR, dan ROUGE. Metrik subjektif yang digunakan (pada eksperimen dan pengujian penambahan konteks) meliputi *fluency* (kefasihan), ketepatan (berdasarkan gambar), dan kualitas atau ketepatan cara penambahan konteks.

Dalam penelitian digunakan tiga macam metrik objektif dengan tujuan sebagai pembandingan. Umumnya cukup digunakan salah satu saja karena ketiga metrik ini memiliki sifat korelasi berbanding lurus; artinya apabila nilai BLEU rendah, maka nilai METEOR dan ROUGE juga rendah, begitu juga sebaliknya. Dalam penelitian, metrik utama yang dipilih adalah BLEU karena banyak penelitian terkait dan penelitian sebelumnya yang menggunakan metrik BLEU. Nilai BLEU dihitung dengan cara membandingkan *caption* yang dibangkitkan dengan *caption* referensi dari *dataset*. Setiap gambar dalam *dataset* Flickr dan MSCOCO memiliki lima referensi, sehingga digunakan nilai rata-rata dari kelima referensi tersebut.

### III.5.3 Implementasi Ulang Model *Image Captioning*

Tujuan dari bagian pertama adalah untuk mendapatkan dan mengoptimasi *baseline image captioning* model untuk masing-masing *dataset*. *Baseline* model didapatkan dengan cara implementasi ulang model mengikuti arsitektur *encoder-decoder* pada penelitian Xu dkk. (2015) dan mengadopsi model *transformer* Vaswani dkk. (2017). Model *image captioning* dilatih menggunakan data latih dan dievaluasi dengan data evaluasi. Selanjutnya dilakukan pencarian parameter terbaik untuk mendapatkan model *image captioning* dengan kinerja paling mendekati (atau sama) dengan hasil penelitian Xu dkk. Model diuji dengan data uji (*test*) untuk mendapatkan nilai akhir kinerja model. Nilai akhir digunakan sebagai pembandingan dengan hasil yang didapatkan oleh Xu dkk. (2015).

Model tersebut kemudian dioptimasi dengan cara dilatih ulang menggunakan *caption* yang dihasilkan oleh *contextual word embedding* yaitu BERT dan GPT-2. Tujuan dari pemakaian *contextual word embedding* ini adalah untuk mengetahui dampak penerapan *contextual word embedding* BERT dan GPT-2 dibandingkan

penggunaan *non-contextual word embedding* (Keras *embedding layer*). Model selanjutnya melalui proses pelatihan, evaluasi, dan pengujian menggunakan *split* yang sama seperti pada saat pembuatan model *baseline* (paragraf sebelumnya). Hipotesis sementara model yang telah dioptimasi dengan *contextual word embedding* memiliki nilai *loss* yang lebih kecil dan BLEU yang lebih baik dibandingkan model tanpa *contextual word embedding*.

#### **III.5.4 Aplikasi Penambahan Konteks pada *Caption* Gambar**

Pada tahapan ini, model terbaik dimodifikasi agar dapat menerima dan memproses model atribut berupa kumpulan kata (*bag of words*) dengan tujuan agar model dapat menambahkan konteks pada *caption* yang dibangkitkan. Model *image captioning* yang digunakan tidak dilatih ulang, namun hanya dimodifikasi dan diujikan pada data uji. Modifikasi yang dilakukan mengacu pada alternatif solusi yang telah disebutkan pada sub bab III.2.

Selanjutnya penilaian objektif dilakukan dengan cara membandingkan nilai BLEU dan jumlah kata-kata dalam BoW yang berhasil dimunculkan (*recall*) antara hasil *caption* yang baru (telah ditambahkan konteks) dengan *caption* yang dibangkitkan model tanpa penambahan konteks. Penilaian juga dilakukan secara subjektif dengan cara membandingkan hasil *caption* secara manual.

Kriteria penilaian dibagi menjadi tiga, yaitu: *fluency* (kefasihan), ketepatan (berdasarkan gambar), dan kualitas atau ketepatan cara penambahan konteks. Ketiga metrik memiliki rentang nilai 1-4 dimana 1 menandakan kurang baik (*poor*), 2 berarti biasa saja / sedang / bisa diterima (*okay*), 3 berarti baik (*good*), dan 4 sangat baik (*excellent*). Detail selengkapnya dapat dilihat pada tabel III.7 (halaman selanjutnya).

**Tabel III.6 Detail Penilaian**

Metrik	Nilai	Keterangan
Kefasihan ( <i>Fluency</i> ) (Fl)	1	Tidak tepat secara susunan bahasa dan atau terdapat bagian yang tidak dapat dipahami.
	2	<i>Caption</i> mengandung minimal subjek predikat, sedikit / tidak ada kesalahan eja
	3	<i>Caption</i> mengandung minimal subjek predikat objek, kalimat relatif sederhana.
	4	<i>Caption</i> mengandung minimal subjek, predikat, objek; disertai keterangan atau beberapa objek atau kata sifat pendukung. Baik secara susunan, dapat dipahami dan menjelaskan dengan detail
Ketepatan <i>caption</i> ( <i>Caption's relevance</i> ) (Cr)	1	Tidak tepat ( <i>incorrect</i> )
	2	<i>Caption</i> memiliki bagian yang benar ( <i>somewhat correct</i> )
	3	<i>Caption</i> sebagian besar sudah benar (ada sedikit yang kurang tepat) ( <i>almost perfect</i> )
	4	<i>Caption</i> sangat menjelaskan dan tepat ( <i>perfect</i> )
Kualitas penambahan konteks ( <i>Context's quality</i> ) (Ctx)	1	Tidak berhasil menambahkan dan cara penambahannya salah, kalimat menjadi tidak dimengerti.
	2	Penambahan berhasil, cara penggunaan tidak terlalu tepat, namun bisa diterima.
	3	Penambahan berhasil, cara penggunaan relatif baik, namun belum terlalu sesuai dengan gambar.
	4	Penambahan berhasil, cara penggunaan tepat, kalimat menjadi lebih baik dan sesuai dengan gambar.

## BAB IV

### IMPLEMENTASI

#### IV.1 Lingkungan Implementasi

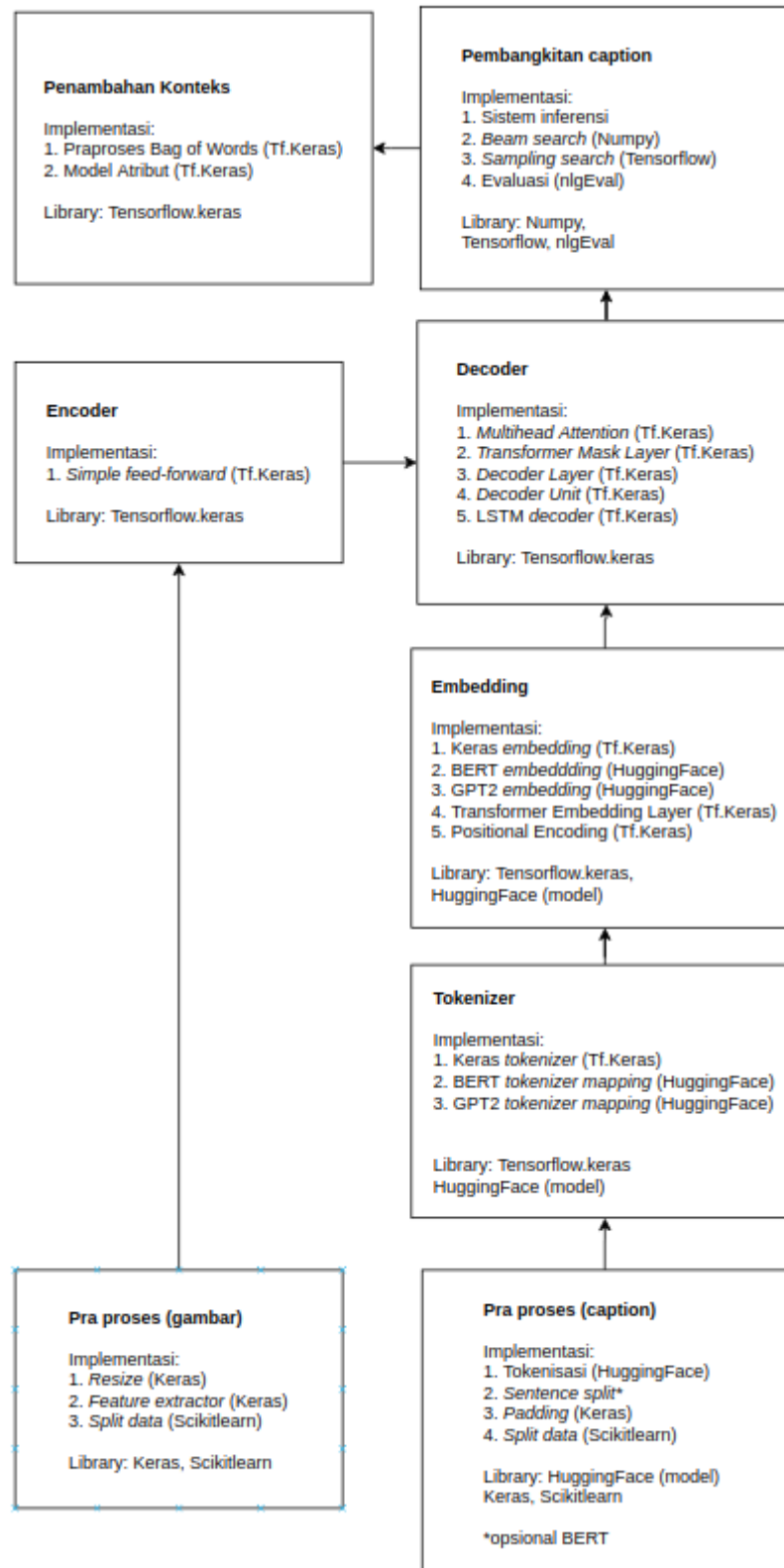
Model *image captioning* dibangun pada lingkungan implementasi yang dapat dilihat spesifikasinya pada tabel IV.1. Terdapat dua jenis perangkat keras yang digunakan, yaitu: GPU (12 GB) dan CPU (96 GB RAM) (tidak digunakan semua). Penggunaan optimal memori GPU berkisar 2-4 GB dan CPU berkisar 8 GB. Tahapan yang memerlukan model *neural network* umumnya memanfaatkan GPU sedangkan untuk tahapan lainnya (misal praproses teks, dan evaluasi) umumnya dilakukan menggunakan CPU.

**Tabel IV.1 Lingkungan Implementasi**

No.	Nama	Spesifikasi
1	Sistem operasi	Ubuntu 18.04, 64-bit
2	Bahasa pemrograman	Python 3.6
3	<i>Text editor</i>	Jupyter Notebook
4	Aplikasi	Jupyter Notebook
5	<i>Parallel computing platform</i>	CUDA 10.2
6	Perangkat keras	GPU, 12 GB RAM & CPU, 96 GB RAM (menggunakan server ITB)

#### IV.2 Implementasi

Berikut rincian modul / komponen yang diimplementasikan sesuai rancangan solusi yang telah diuraikan pada bab III.4 (gambar IV.1):



Gambar IV.1 Diagram Implementasi



### IV.2.1 Praproses Gambar (tensorflow.keras)

Komponen praproses gambar diimplementasikan menggunakan model pralatih dari Keras.

```
from keras.preprocessing.image import load_img, img_to_array
from keras.applications.xception import preprocess_input

PARAMS['image_shape'] = (299, 299, 3)
PARAMS['image_feature_size'] = 1000

def get_image_feature(image_path):

    image = load_img(image_path, target_size=PARAMS['image_shape'])
    image = img_to_array(image)
    image = preprocess_input(image)
    return image_extractor.predict(image.reshape((1,) +
image.shape[:3])).flatten()

image_extractor = keras.applications.xception.Xception(include_top=True,
weights='imagenet')

for image_path in tqdm(dataset.keys()):
    filename = IMAGE_DIR + "/" + image_path
    image_feature = get_image_feature(filename)
    np.save(filename + ".npy", image_feature)
```

### IV.2.2 Praproses Teks

#### IV.2.2.1 Tokenisasi (HuggingFace)

Modul tokenisasi menggunakan *tokenizer* yang didapat dari kakas HuggingFace. Modul tokenisasi tidak dapat dibuat terpisah karena pemetaan id harus sesuai dengan id token yang dipakai oleh model *word embedding*. Detail implementasi berupa kelas yang membungkus *tokenizer* BERT dan GPT-2 dari kakas HuggingFace. Berikut contoh untuk *tokenizer* BERT:

```

class BertTokenizerWrapper(BertTokenizer):

    def use_custom_mapping(self, use_mapping=True, vocab_size=5000):
        self.use_mapping = use_mapping
        self.cust_vocab_size = vocab_size
        self.mapping_initialized = False

    def texts_to_sequences(self, texts):
        texts = self._add_start_end_seq_token(texts)
        return [self.convert_tokens_to_ids(self.tokenize(x)) for x in texts]

    def _add_start_end_seq_token(self, texts):
        return ["{} {} {}".format(START_TOKEN, x, END_TOKEN) for x in texts]

    def convert_tokens_to_ids(self, tokens):
        bert_ids = self._get_bert_ids(tokens)
        if not self.use_mapping:
            return bert_ids
        if not self.mapping_initialized:
            raise Exception("mapping not initialized")
        return self._convert_bert_id_to_custom_id(bert_ids)

    def convert_ids_to_tokens(self, token_ids, **kwargs):
        if self.use_mapping and self.mapping_initialized:
            bert_ids = self._convert_custom_id_to_bert_id(token_ids)
        else:
            bert_ids = token_ids
        bert_tokens = super().convert_ids_to_tokens(bert_ids, **kwargs)
        return bert_tokens

    def initialize_custom_mapping(self, texts):
        bert_ids = [self._get_bert_ids(self.tokenize(x)) for x in tqdm(texts)]
        self._build_occurence_table(bert_ids)
        self._build_custom_mapping_table()
        self.mapping_initialized = True

    def _get_bert_ids(self, tokens):

```

```

        return super().convert_tokens_to_ids(tokens)

def _build_occurence_table(self, tokenized_captions):
    self.occurence_table = {}
    for caption in tqdm(tokenized_captions):
        for token in caption:
            if token not in self.occurence_table:
                self.occurence_table[token] = 0
            self.occurence_table[token] += 1

def _build_custom_mapping_table(self):

    _special_token = ['[PAD]', '[UNK]', '[CLS]', '[SEP]']
    _actual_vocab_size = self.cust_vocab_size - len(_special_token)

    sorted_occurence = {k: v for k, v in sorted(
        self.occurence_table.items(), reverse=True, key=lambda item: item[1]
    )}

    used_tokens = sorted(list(sorted_occurence)[:_actual_vocab_size])
    mapping_size = min(len(used_tokens), _actual_vocab_size)

    _bert_pad = 0
    _bert_oov = 100
    _bert_bos = 101
    _bert_eos = 102
    self._custom_pad = 0
    self._custom_oov = 1
    self._custom_bos = 2
    self._custom_eos = 3

    self.bert_id_to_custom_id = {
        _bert_pad: self._custom_pad,
        _bert_oov: self._custom_oov,
        _bert_bos: self._custom_bos,
        _bert_eos: self._custom_eos,
    }
    self.custom_id_to_bert_id = {
        self._custom_pad: _bert_pad,
        self._custom_oov: _bert_oov,

```

```

        self._custom_bos: _bert_bos,
        self._custom_eos: _bert_eos,
    }

    for i in range(0, mapping_size):
        bert_token = used_tokens[i]
        self.bert_id_to_custom_id[bert_token] = i + len(_special_token)
        self.custom_id_to_bert_id[i + len(_special_token)] = bert_token

    sorted_occurence_count = list(sorted_occurence.values())
    used_tokens_count = sum(sorted_occurence_count[:_actual_vocab_size])
    total_tokens_count = sum(sorted_occurence_count)

    def _convert_bert_id_to_custom_id(self, token_ids):
        return [self.bert_id_to_custom_id[x] if x in self.bert_id_to_custom_id
                else self._custom_oov for x in token_ids]

    def _convert_custom_id_to_bert_id(self, token_ids):
        return [self.custom_id_to_bert_id[x] for x in token_ids]

```

#### IV.2.2.2 Membangun Kosakata (HuggingFace)

Pembangunan kosakata dilakukan dengan cara mengambil *top-k* token yang paling sering muncul dari korpus *caption* yang telah diubah menjadi token BERT / GPT-2. Berikut data statistik dari kosakata yang dibentuk berdasarkan nilai *k* dan *dataset* (tabel IV.2) (halaman berikutnya).

#### IV.2.2.3 Penambahan *Padding* (tensorflow.keras)

Penambahan *padding* bertujuan menyeragamkan panjang sekuens *caption*. *Padding* yang ditambahkan adalah token [PAD] masing-masing *tokenizer* (id 0 untuk BERT, id 50257 untuk GPT-2). Berikut data statistik panjang *caption* dalam satuan token sebelum ditambahkan *padding* (tabel IV.3):

**Tabel IV.2 Statistik Kosakata**

<i>Dataset</i>	MSCOCO				Flickr-8k		
<i>Tokenizer</i>	GPT-2	GPT-2	BERT	Keras	GPT-2	BERT	Keras
Token unik yang digunakan	15000	5000	5000	5000	5000	5000	5000
Jumlah token unik (dalam <i>dataset</i> )	20468	20468	14533	23684	8517	7421	8496
Token yang digunakan	4929241	4851022	5674504	4334576	495897	575256	437623
Jumlah token (dalam <i>dataset</i> )	4935190	4935190	5720081	4391968	500284	577925	441276
Persen token unik	73.27 %	24.41 %	34.39 %	21.11%	58.66 %	67.35 %	58.85%
Persen penggunaan token	99.88 %	98.29 %	99.20 %	98.69%	99.12 %	99.54 %	99.17%

**Tabel IV.3 Statistik Panjang *Caption***

Panjang <i>caption</i> (satuan: token)	MSCOCO		Flickr-8k	
	GPT-2	BERT	GPT-2	BERT
Rata-rata ( <i>mean</i> )	11.91	11.81	12.36	12.28
Standar deviasi ( <i>std</i> )	2.85	2.81	4.04	4.04

**IV.2.2.4 Pemisahan Kalimat**

Pemisahan kalimat menjadi pecahan sekuens kalimat dilakukan untuk membangun *dataset* bagi model *image captioning* berbasis rekuren. Proses ini dilakukan

bersamaan dengan penambahan *padding*. Komponen ini bersifat opsional untuk model berbasis *transformer*.

### IV.2.3 Pembagian Data (sklearn)

Proses pembagian data menjadi data latih, evaluasi dan data uji dengan kaskas *train\_test\_split* sklearn. Pertama-tama gambar dibagi untuk data latih-evaluasi sebanyak 80% dan data uji 20%, kemudian data latih-evaluasi dibagi lagi menjadi data latih sebesar 80% dan data evaluasi 20%. Pembagian dilakukan secara acak namun menggunakan *seed* yang sama untuk setiap percobaan.

### IV.2.4 Model (tensorflow)

Sebagian besar model diimplementasi menggunakan tensorflow.keras *functional* API, baik untuk pembuatan *layer* maupun model secara keseluruhan. Model tersusun atas delapan komponen.

#### IV.2.4.1 Contextual Word Embedding (HuggingFace)

Kedua model *contextual word embedding* dibungkus dalam kelas *wrapper* masing-masing dengan tujuan untuk menyeragamkan *input* dan cara pemanggilan. Karena token *input* yang digunakan diberi *padding*, maka saat memanggil model *contextual word embedding*, perlu diberikan vektor *mask* yang menandakan token mana saja yang berupa *padding*.

```
from tensorflow.keras import Layers
from transformers import TFBertModel, TFGPT2Model

class BertEmbedding(Layers.Layer):

    def __init__(self, **kwargs):
        super(BertEmbedding, self).__init__(**kwargs)
        self.embedding = TFBertModel.from_pretrained('bert-base-uncased')
        self.embedding.trainable = False
        self.embedding_dim = self.embedding.config.hidden_size

    def call(self, inputs):
        is_sentence = tf.cast((inputs == 0), tf.int32)
```

```

        hidden_states, _ = self.embedding(inputs=inputs,
token_type_ids=is_sentence)
        return hidden_states

    def get_config(self):
        config = super(BertEmbedding, self).get_config()
        return config

class GPT-2Embedding(Layers.Layer):

    def __init__(self, **kwargs):
        super(GPT-2Embedding, self).__init__(**kwargs)
        self.embedding = TFGPT-2Model.from_pretrained('distilgpt2')
        self.embedding.trainable = False
        self.embedding_dim = self.embedding.config.hidden_size

    def call(self, inputs):
        attention_mask = tf.cast((inputs != PAD_VALUE), tf.int32)
        hidden_states, _ = self.embedding(inputs=inputs,
attention_mask=attention_mask)
        return hidden_states

    def get_config(self):
        config = super(TFGPT-2Model, self).get_config()
        return config

```

#### IV.2.4.2 Embedding Layer (tensorflow.keras)

Untuk memudahkan penggantian jenis *embedding* saat pengujian, dibuatlah sebuah *layer embedding* yang mengakomodasi BERT, GPT-2, dan Keras *embedding layer*. *Layer* ini juga memanggil fungsi *positional encoding* yang dibuat sebagai fungsi terpisah.

#### IV.2.4.3 Positional Encoder (tensorflow.keras)

*Positional encoder* yang digunakan dibuat dalam bentuk fungsi dan mengacu referensi tensorflow. *Positional encoder* berfungsi untuk menambahkan informasi urutan atau posisi token bagi model berbasis *transformer*.

```
def get_angles(pos, i, d_model):
    angle_rates = 1 / np.power(10000, (2 * (i//2)) / np.float32(d_model))
    return pos * angle_rates

def positional_encoding(position, d_model):
    angle_rads = get_angles(np.arange(position)[:, np.newaxis],
                            np.arange(d_model)[np.newaxis, :],
                            d_model)

    # apply sin to even indices in the array; 2i
    angle_rads[:, 0::2] = np.sin(angle_rads[:, 0::2])

    # apply cos to odd indices in the array; 2i+1
    angle_rads[:, 1::2] = np.cos(angle_rads[:, 1::2])

    pos_encoding = angle_rads[np.newaxis, ...]
    return tf.cast(pos_encoding, dtype=tf.float32)
```

#### IV.2.4.4 Multihead Attention Block (tensorflow.keras)

Komponen ini dibuat dengan menggunakan referensi tensorflow yang mengacu implementasi asli Vaswani dkk. (2017). *Multihead attention* berfungsi sebagai inti dari komponen perhitungan bobot *decoder* berbasis *transformer*. Komponen ini terdiri dari tiga buah matriks bobot yaitu *query*, *key*, dan *value* seperti pada implementasi asli Vaswani dkk. (2017).

#### IV.2.4.5 Transformer Mask (tensorflow.keras)

Komponen ini dibuat dengan menggunakan referensi tensorflow yang mengacu implementasi asli Vaswani dkk. (2017). Komponen *mask* berfungsi untuk



menandakan token yang bersifat *padding*, serta mencegah *decoder* untuk mengetahui informasi token  $t + 1$  dst. pada tahap ke- $t$ . Komponen *mask* umumnya digunakan oleh *decoder* dan *encoder* pada model *transformer*, namun karena *input encoder* adalah gambar maka komponen *mask* tidak perlu digunakan.

#### IV.2.4.6 Decoder Layer (tensorflow.keras)

Komponen ini dibuat dengan menggunakan referensi tensorflow yang mengacu implementasi asli Vaswani dkk. (2017). *Decoder* pada model berbasis *transformer* tersusun dari beberapa *decoder layer*. Sebuah *decoder layer* terdiri dari dua *multihead attention layer* dan sebuah *feedforward layer* seperti implementasi asli Vaswani dkk. (2017).

#### IV.2.4.7 Decoder (tensorflow.keras)

*Decoder* yang diimplementasikan adalah *decoder* berbasis *transformer*. Dalam penelitian *decoder* berbasis model rekuren, model menggunakan komponen LSTM yang telah diimplementasikan oleh tensorflow.keras. Implementasi *decoder* mengacu kepada tutorial tensorflow dan arsitektur asli Vaswani dkk. (2017) namun dengan sedikit perubahan yaitu adanya *residual connection* antar *decoder layer*. Penambahan ini dilakukan karena model tidak berhasil mencapai konvergensi saat dilatih. Berikut cuplikan perubahan yang ditambahkan untuk membuat *residual connection*.

```
from tensorflow.keras.layers import Dropout, LayerNormalization

class Decoder(tf.keras.layers.Layer):
    def __init__(**kwargs):
        ...

    def build(self, input_shape):
        ...
        self.layernorms = [LayerNormalization() for _ in range(self.num_layers)]

    def call(self, x):
        prev_x = x * 0
        for i in range(self.num_layers):
```

```

        x = self.layernorms[i](x + prev_x)
        prev_x = x
        x = self.decoder_layers[i](x)

    return x

```

#### IV.2.4.8 Keseluruhan Sistem (tensorflow.keras)

Keseluruhan model dibangun dengan menggunakan *functional* API yang disediakan tensorflow.keras. Dalam implementasi, terdapat perbedaan masukan dan keluaran untuk model berbasis rekuren dan *transformer*. Masukan yang diterima model rekuren adalah vektor gambar serta token yang telah dihasilkan, sedangkan keluaran model adalah distribusi probabilitas untuk (satu) token berikutnya. Sebagai contoh jika masukan adalah “A brown” maka keluaran yang diharapkan adalah probabilitas yang memaksimalkan nilai token “dog”.

Pada model *transformer*, *input* yang diterima adalah vektor gambar dan *caption* keseluruhan yang digeser satu langkah ke kanan (metode *teacher forcing*), dan keluaran model adalah distribusi probabilitas token untuk masing-masing ‘posisi’ yang diisi oleh suatu token untuk membentuk kesatuan *caption* yang utuh (dari posisi ke 0 sampai panjang maksimum *caption*). Sebagai contoh, jika masukan adalah “[CLS] A brown”, maka keluaran yang diharapkan adalah distribusi yang memaksimalkan kata ‘A’ di posisi ke 0, ‘brown’ pada posisi 1, dan ‘dog’ pada posisi 2, sehingga sebagai kesatuan *output*-nya adalah ‘A brown dog’.

```

from tensorflow.keras.models import Model
from tensorflow.keras.layers import Masukan, Dense, BatchNormalization,
LeakyReLU, Embedding, Bidirectional, LSTM, Concatenate,
GlobalAveragePooling1D, Reshape, Flatten, Masking

# ===== input image =====
input_image = Masukan(shape=(PARAMS["image_feature_size"],),
name="input_image")

```

```

# ===== input caption =====
input_caption = Masukan(shape=(PARAMS["max_caption_length"],),
name="input_caption", dtype="int32")
x_caption = Masking(mask_value=0)(input_caption)

# ===== masking =====
# !! force mask to have shape (batchsize, 1) instead of (batchsize, 768)
_dummy_image = tf.reduce_sum(tf.expand_dims(input_image, axis=1), axis=-1)
enc_padding_mask, combined_mask, dec_padding_mask =
TransformersMask()(_dummy_image, input_caption)

# ===== system ===== #
layer_caption_embedding = TransformerEmbedding(embedding_dim=768,
embedding_type="GPT-2", vocab_size=5000)
emb_caption = layer_caption_embedding(x_caption)

layer_decoders = Decoder(num_layers=6, d_model=768, num_heads=8, dff=1024,
rate=0.1)
dec_output, attention_weights = layer_decoders(emb_caption, input_image,
training=True, look_ahead_mask=combined_mask, padding_mask=None)

output = Dense(PARAMS["vocab_size"], activation='softmax')(dec_output)

# ===== compile ===== #
loss = masked_sparse_cce_loss
model = Model(inputs=[input_image, input_caption], outputs=output)
model.compile(loss=loss, optimizer='adam')
model.summary()

```

#### IV.2.5 Strategi Pemilihan Kata

Berikut implementasi beberapa jenis strategi untuk memilih kata yang dibangkitkan berdasarkan probabilitas (keluaran) *dense layer* terakhir. Alternatif strategi yang dapat digunakan terdiri dari metode *greedy*, *sampling* dan *beam search*.

```

from numpy import array, log
from numpy import argmax

START_TOKEN = caption_tokenizer.encode(['[CLS]'])[0]
END_TOKEN = caption_tokenizer.encode(['[SEP]'])[0]

def choose_word_ids(vocab_proba, strategy="greedy", seq_index=0):

    vocab_proba = vocab_proba[:, :seq_index + 1]

    if strategy == "beam_search":
        predicted_ids, _ = beam_search_decoder(vocab_proba, 3,
return_best=True)

    elif strategy == "sample":
        predicted_ids = sampling_search_decoder(vocab_proba, k=3)

    elif strategy == "greedy":
        predicted_ids = np.argmax(vocab_proba, axis=-1)

    else:
        raise Exception("not implemented")

    token_ids = [target_tokenizer._convert_custom_id_to_gpt2_id(x) for x in
predicted_ids]
    return np.array(token_ids)

```

```

def beam_search_decoder(data, k, return_best=True):

    _shape = data.shape

    sequences = np.zeros((_shape[0], k, _shape[1]), dtype="int32")
    sequences_score = np.ones((_shape[0], k,))

    # for each sequence
    for row in tqdm(range(_shape[1])):

```

```

# calculate score
_score = np.expand_dims(sequences_score, 2)
_prob = np.expand_dims(data[:, row], 1)
_score = _score * -log(_prob)
_reshape_score = _score.reshape(_shape[0], -1)

# get top 3 indices for each batch:
_c = np.empty((_shape[0], k), dtype="int32")
for i in range(_shape[0]):
    _c[i] = np.unique(_reshape_score[i], return_index=True)[1][:k]

_choosen_seqs = _c // _shape[2]
_choosen_words = _c % _shape[2]

# replace sequence with chosen seq
for i in range(_choosen_seqs.shape[0]):
    sequences[i] = sequences[i, _choosen_seqs[i]]

# replace score with new score
for i in range(_shape[0]):
    sequences_score[i] = _score[i, _choosen_seqs[i],
_choosen_words[i]]

# append new chosen word
sequences[:, :, row] = _choosen_words

# if return best only:
if return_best:

    seq_idx = sequences_score.argmax(axis=-1)
    for i in range(_shape[0]):
        sequences[i] = sequences[i, seq_idx[i]]
        sequences_score[i] = sequences_score[i, seq_idx[i]]

    sequences = np.mean(sequences, axis=1, dtype="int32")
    sequences_score = np.mean(sequences_score, axis=1)

return sequences, sequences_score

```

```
def sampling_search_decoder(predictions, k=3):

    # shape = batch_size, seq_len, vocab_size
    shape = predictions.shape

    predictions = tf.reshape(predictions, (-1, shape[-1]))

    # sampled_proba & sampled_ids => (batch_size * seq_len, sampling_k)
    sampled_proba, sampled_ids = tf.math.top_k(predictions, k)

    # chosen_sampled_col => (batch_size * seq_len, )
    chosen_sampled_col = tf.squeeze(tf.random.categorical(sampled_proba, 1))
    chosen_sampled_col = tf.reshape(chosen_sampled_col, (-1,))

    # create row idx to zip with chosen_sampled_col
    row_idx = tf.range(predictions.shape[0], dtype=chosen_sampled_col.dtype)
    row_col_idx = tf.stack([row_idx, chosen_sampled_col], axis=1)

    # predicted_ids => (batch_size, seq_len, )
    predicted_ids = tf.gather_nd(sampled_ids, row_col_idx)
    predicted_ids = tf.reshape(predicted_ids, shape[:-1])

    return predicted_ids.numpy()
```

#### IV.2.6 Pembangkitan *Caption*

Berikut potongan kode untuk membangkitkan *caption* dari sebuah gambar. Fungsi menerima *input* berupa gambar dan model yang digunakan, dan mengeluarkan *caption* yang berhasil dibangkitkan model.

```
from IPython.display import clear_output

def generate_caption(model, image, strategy=("greedy", "append")):
    ...

    image.shape = (1,4462)
    ...
```

```

image = np.array([image])
sequence = [START_TOKEN]

for i in (range(PARAMS["max_caption_length"])):

    input_caption = np.array([sequence])
    input_caption = pad_sequences(input_caption, value=PAD_VALUE,
maxlen=PARAMS["max_caption_length"], padding="post")

    yhat = model.predict([image, input_caption], verbose=0)

    choosen_ids = choose_word_ids(yhat, strategy=strategy[0],
seq_index=i)[0]
    sequence = append_choosen_word(sequence, choosen_ids,
strategy=strategy[1], seq_index=i)

    if sequence[-1] == END_TOKEN:
        break

prediction_text = caption_tokenizer.decode(sequence,
skip_special_tokens=True)
return(prediction_text)

```

#### IV.2.7 Penambahan Konteks

Modul yang digunakan untuk menambahkan konteks pada model *image captioning* diimplementasikan dengan nama PPLM (sesuai referensi Dathathri dkk. 2019). Dalam model yang diimplementasikan, terdapat empat jenis parameter yang dapat diatur untuk mempengaruhi hasil yang didapat, diantaranya:

1. *Iteration*

Parameter *iteration* menandakan berapa kali proses perubahan vektor *decoder* dilakukan untuk setiap proses pembangkitan kata. Pergeseran / perubahan distribusi bertambah besar apabila jumlah iterasi ditingkatkan.

2. *Lr*

Lr sebanding dengan pembobotan *loss*, apabila Lr mendekati 1, maka *loss* dari

model atribut sepenuhnya digunakan untuk mencari gradien. Sebaliknya apabila  $lr$  mendekati 0, maka  $loss$  tidak digunakan dan gradien yang dihasilkan mendekati 0.

### 3. *Optimizer\_lr*

Dalam atribut model digunakan *optimizer* (Adam) untuk menghitung gradien dan *optimizer* ini memiliki parameter internal  $lr$  (*learning rate*). Parameter ini mempengaruhi proses perhitungan gradien vektor yang dihasilkan dari model atribut.

### 4. *Gamma*

*Gamma* adalah konstanta yang digunakan sebagai bobot saat menjumlahkan distribusi asli dan distribusi setelah dilakukan pergeseran dengan vektor. Konstanta ini dapat dipandang sebagai rasio atau perbandingan distribusi awal dan hasil pergeseran. Jika *gamma* mendekati 0 maka hasil pergeseran sepenuhnya diabaikan, sebaliknya *gamma* mendekati 1 maka distribusi asli diabaikan.

```
from tensorflow.keras.losses import CategoricalCrossentropy
from tensorflow.keras.optimizers import Adam

class PPLM(tf.keras.layers.Layer):

    def __init__(self, model, iteration=3, lr=0.003, optimizer_lr=0.01,
ratio=0.8, **kwargs):
        super(PPLM, self).__init__()

        self.model = model
        self.pplm_iteration = iteration
        self.pplm_lr = lr
        self.optimizer_lr = optimizer_lr
        self.pplm_rasio = ratio

        self.ori_weights = None
        self.input_pert = None
```



```

self.loss_function = CategoricalCrossentropy(from_logits=False)
self.optimizer = Adam(lr=optimizer_lr)

def call(self, x, support_vector=None):

    if support_vector is None:
        return dec_output

    self._init_ori_weights()
    self._init_input_perturbation(x)

    ori_prediction, pplm_prediction = self._apply_pplm(x, support_vector)
    output = self._merge_output(ori_prediction, pplm_prediction)

    self._restore_weights()
    return output, ori_prediction

def _init_ori_weights(self):
    self.ori_weights = [tf.identity(x) for x in self.model.weights] # deep
copy

def _init_input_perturbation(self, x):
    self.input_pert = tf.Variable(tf.zeros(x.shape), trainable=True)

def _merge_output(self, ori_prediction, pplm_prediction):
    return (pplm_prediction * self.pplm_ratio) + (ori_prediction * (1 -
self.pplm_ratio))

def _restore_weights(self):
    self.model.set_weights(self.ori_weights)

def _apply_pplm(self, x, support_vector):

    pplm_results = []
    for _ in range(self.pplm_iteration + 1):

        with tf.GradientTape() as pplm_tape:

            dec_output = self._pert_input(x)

```

```

        prediction = self.model(dec_output)
        pplm_results.append(prediction)

        pplm_loss = self._calculate_loss(support_vector, prediction)

        trainable_variables = self.model.trainable_variables +
[self.input_pert]
        gradients = pplm_tape.gradient(pplm_loss, trainable_variables)
        self.optimizer.apply_gradients(zip(gradients,
trainable_variables))

    return pplm_results[0], pplm_results[-1]

def _pert_input(self, x):
    return x + self.input_pert

def _calculate_loss(self, real, pred):
    """
    real => (batch_size, vocab_size)
    pred => (batch_size, seq_len, vocab_size)
    """

    seq_len = pred.shape[1]

    real = tf.expand_dims(real, axis=1)
    real = tf.tile(real, [1, seq_len, 1])

    pplm_loss = self.loss_function(real, pred, self.pplm_lr)

    return pplm_loss

```

#### IV.2.8 Praproses Model Atribut (*Bag of Words*)

Sebelum model atribut jenis *Bag of Words* dapat digunakan, perlu melalui beberapa tahap praproses, diantaranya:

1. Tokenisasi: mengubah kata-kata dalam BoW menjadi bentuk token menggunakan *tokenizer* yang sama untuk *caption* dan *contextual word embedding*.
2. Membuang kata-kata yang mengandung token [UNK]. Token [UNK] menandakan terdapat bagian dari kata yang tidak dikenal oleh *tokenizer* (di luar kosakata atau OOV).
3. Menggabungkan semua token menjadi satu vektor *one hot encoding*. contoh: token tersisa adalah [3, 5], maka dibuat menjadi [[0,0,0,1,0,0,...], [0,0,0,0,0,1,...]]
4. Hasil *one hot encoding* digabungkan menjadi satu representasi BoW dengan cara dijumlahkan.  
  
contoh: [[0,0,0,1,0,0,...], [0,0,0,0,0,1,...]] dijumlahkan menjadi [0,0,0,1,0,1,...].

```
pplm_vocab = open("vocabs/vocab_military.txt").read().splitlines()

# change word to id,
pplm_vocab = [target_tokenizer.encode(x) for x in pplm_vocab]

# remove OOV id
pplm_vocab = list(filter(lambda x: UNK_TOKEN not in x, pplm_vocab))

# flatten
pplm_vocab = [item for sublist in pplm_vocab for item in sublist]

# remove duplicate
pplm_vocab = sorted(set(pplm_vocab))

# one hot then merge
support_vector = tf.reduce_sum(to_categorical(pplm_vocab), axis=0)
```

```
pad = PARAMS["vocab_size"] - support_vector.shape[0]

# add padding to match vocab_size
support_vector = tf.concat([support_vector, tf.zeros(pad)], axis=0)
support_vector = tf.expand_dims(support_vector, axis=0)
```

#### IV.2.9 Evaluasi (nlgeval)

Kakas nlgeval digunakan untuk evaluasi hasil pembangkitan *caption*. Metrik yang dipakai terdiri dari Bleu\_1, Bleu\_2, Bleu\_3, Bleu\_4, METEOR, dan ROUGE\_L. Dalam penelitian juga digunakan metrik penilaian relatif seperti *Kullback–Leibler divergence* dan *delta BLEU* untuk membandingkan perubahan *caption*. Metrik sederhana lainnya yang digunakan adalah *recall*. *Recall* digunakan untuk menghitung jumlah kemunculan kata dalam kosakata model atribut pada *caption* yang telah dibangkitkan.

## BAB V

### EKSPERIMEN DAN PENGUJIAN

#### V.1 Eksperimen

Setelah model diimplementasikan, penelitian dilanjutkan dengan eksperimen untuk menjawab rumusan masalah dan tujuan penelitian. Beberapa tujuan dari eksperimen mencakup:

1. Menentukan jenis *contextual word embedding* dengan kinerja terbaik untuk *task* pembangkitan *caption*.
2. Menentukan kombinasi parameter model atribut yang dapat membangkitkan *caption* berkonteks dengan kualitas baik.

##### V.1.1 Data Eksperimen

*Dataset* utama eksperimen yang digunakan terdiri atas 80% data MSCOCO. Mencakup 66226 gambar unik dengan jumlah total *caption* unik sebanyak 331250 *caption*. Data eksperimen ini dibagi lagi 80% untuk data latih dan 20% untuk data evaluasi. Dalam eksperimen juga digunakan *dataset* pembandingan Flickr 8K yang diambil 80% data untuk eksperimen. Detail pembagian data ditunjukkan pada tabel V.1.

**Tabel V.1 Statistik Pembagian Data**

<i>Dataset</i>	Jenis	Data Latih	Data Eval	Data Uji	Total
Flickr 8k	Gambar	6472 (80%)		1619 (20%)	8091 (100%)
	<i>Caption</i>	25888 (~64%)	6472 (~16%)	8095 (~20%)	40455 (100%)
MSCOCO	Gambar	66226 (80%)		16557 (20%)	82783 (100%)
	<i>Caption</i>	265024 (~64%)	66257 (~16%)	82831 (~20%)	414112 (100%)

Dalam eksperimen yang melibatkan model rekuren, *caption* dari gambar dipisahkan menjadi  $n$  data (dengan  $n$  adalah jumlah token dalam satu *caption*) untuk mensimulasikan setiap tahap pembangkitan token.

## V.1.2 Eksperimen *Contextual Word Embedding*

### V.1.2.1 Skenario Eksperimen

Eksperimen diawali dengan proses implementasi ulang model *image captioning* berbasis rekuren (LSTM) dan *transformer*. Kedua jenis model ini kemudian dilatih dan dievaluasi berdasarkan hasil pembangkitan *caption*. Beberapa variabel bebas yang digunakan dalam eksperimen adalah *dataset* dan jenis *word embedding* yang digunakan. Parameter masing-masing model dianggap sebagai variabel kontrol sehingga parameter kedua model yang digunakan disamakan untuk setiap kasus pengujian. Untuk menentukan parameter model yang digunakan, dilakukan pengujian terpisah untuk mendapatkan parameter terbaik. Terdapat total 12 kombinasi model ( $2 \text{ model} \times 3 \text{ jenis embedding} \times 2 \text{ dataset}$ ) yang dilatih dan dievaluasi, detail parameter ditunjukkan pada tabel V.2. Dalam proses evaluasi, setiap model membangkitkan 1500 *caption*, dan hasilnya dibandingkan dengan *caption* asli *dataset* menggunakan metrik BLEU-1, BLEU-4, METEOR dan ROUGE-L.

**Tabel V.2 Detail Kombinasi Parameter**

Jenis varian	Varian	Jumlah varian
<i>Dataset</i>	Flickr 8k, MSCOCO	2
Model	LSTM, <i>Transformer</i>	2
<i>Embedding</i>	Keras <i>Embedding</i> , BERT, GPT-2	3

Model berbasis rekuren dan *transformer* memiliki cara pembelajaran yang berbeda sehingga data latih dan evaluasi disesuaikan formatnya seperti disebutkan pada sub bab V.1.1.

### V.1.2.2 Hasil Eksperimen

Eksperimen dilakukan terhadap dua jenis model dengan rincian:

1. LSTM (mewakili jenis model rekuren): menggunakan 768 unit; *dropout* 0.1; Pelatihan dilakukan sampai model konvergen (sekitar 12-15 *epoch*), dengan ukuran *batch* 64.
2. *Transformer*: menggunakan 6 *layer decoder*; 768 unit yang dibagi menjadi 8 *head* (*multi head attention*), masing-masing *head* terdiri dari 96 unit ( $8 \times 96 = 768$ ); *batchnorm* dan *dropout* 0.1 diberikan antar *decoder layer*. *Training* dilakukan selama sampai model konvergen (sekitar 8-12 *epoch*), dengan ukuran *batch* 64.

Parameter model yang digunakan dalam eksperimen didapatkan dari referensi penelitian terkait serta hasil eksperimen yang dilakukan untuk mencari kombinasi arsitektur dan parameter terbaik. Hasil eksperimen ditunjukkan pada tabel V.3 dan V.4.

**Tabel V.3 Hasil Eksperimen pada MSCOCO**

	<i>Dataset: MSCOCO</i>					
No.	Jenis model		Metrik			
	Model	<i>Embedding</i>	BLEU-1	BLEU-4	METEOR	ROUGE-L
1	LSTM	Keras	0.572	0.159	0.191	0.380
2	LSTM	BERT	0.536	0.134	0.199	0.373
3	LSTM	GPT-2	0.611	0.169	0.207	0.431
4	<i>Transformer</i>	Keras	0.581	0.164	0.196	0.382
5	<i>Transformer</i>	BERT	0.044	0.000	0.022	0.055
	<i>Transformer</i>	BERT*	0.500	0.113	0.179	0.349
6	<i>Transformer</i>	GPT-2	0.619	0.187	0.212	0.438

BERT\*: *embedding* BERT namun metode pelatihan dilakukan secara sekuensial (*caption* dipecah untuk setiap tahap pembangkitan).

**Tabel V.4 Hasil Eksperimen pada Flickr 8K**

	<i>Dataset: Flickr 8K</i>					
No.	Jenis model		Metrik			
	Model	<i>Embedding</i>	BLEU-1	BLEU-4	METEOR	ROUGE-L
1	LSTM	Keras	0.438	0.094	0.154	0.299
2	LSTM	BERT	0.425	0.084	0.178	0.303
3	LSTM	GPT-2	0.531	0.121	0.182	0.363
4	<i>Transformer</i>	Keras	0.507	0.136	0.181	0.349
5	<i>Transformer</i>	BERT	0.082	0.000	0.031	0.073
	<i>Transformer</i>	BERT*	0.418	0.087	0.175	0.294
6	<i>Transformer</i>	GPT-2	0.535	0.152	0.202	0.387

### V.1.2.3 Analisis Hasil Eksperimen

Berdasarkan hasil eksperimen, didapatkan bahwa penggunaan *word embedding* kontekstual GPT-2 menghasilkan model dengan kinerja yang lebih baik terlepas arsitektur model yang digunakan. Terlihat pada tabel V.3 dan V.4, baik model berbasis rekuren (LSTM) maupun *transformer*, model dengan *word embedding* GPT-2 selalu lebih unggul dibandingkan model lainnya. Disisi lain, penggunaan *transformer* dengan BERT justru memberikan hasil yang lebih buruk dibandingkan GPT-2 maupun *embedding* non-kontekstual KE (Keras *Embedding*).

Hasil dari eksperimen semakin memperkuat hipotesis bahwa BERT justru mempersulit model *image captioning* dalam membangkitkan *caption* karena sifat



dua-arahnya. Dalam eksperimen, apabila model *transformer*-BERT digunakan apa adanya, hasil pembangkitannya memiliki nilai BLEU yang mendekati 0, padahal pada proses pembelajaran, *loss* yang didapat dari model relatif lebih kecil dibandingkan model lainnya.

Penulis melakukan eksperimen lanjutan pada model *transformer*-BERT dan mendapatkan hasil bahwa apabila model diuji dengan masukan satu sekuens (seperti pada proses pembelajaran), model dapat menghasilkan sekuens yang hampir sempurna (BLEU-1 0.89). Sebagai contoh, masukkan berupa gambar dan *caption* “[CLS] a brown dog”, model dapat mengeluarkan *caption* “a brown dog [SEP]”, lalu kedua *caption* dibandingkan. Hal tersebut tidak wajar karena model berbasis *transformer* seharusnya dapat dilatih dan diuji dengan pendekatan *transformer* dan *autoregressive* (seperti *transformer*-GPT atau *transformer*-Keras-embedding).

Penulis mengasumsikan tidak terjadi kesalahan dalam implementasi model *transformer* karena pada kasus GPT-2 dan Keras *embedding* model dapat dilatih dan membangkitkan *caption* dengan baik. Penulis menyimpulkan alasan *transformer*-BERT tidak dapat dilatih dan membangkitkan *caption* dengan baik adalah karena sifat dua-arah BERT tidak cocok dengan *task autoregression*.

Salah satu alternatif jika tetap ingin menggunakan *transformer*-BERT adalah dengan memecah kalimat *caption* seperti pada proses pelatihan model rekuren. Namun pendekatan ini menghilangkan tujuan utama penggunaan model *transformer* yaitu melakukan paralelisasi dan mempercepat proses pembelajaran. Walaupun sudah dilatih secara sekuensial, hasil pembangkitan *caption transformer*-BERT\* tetap memiliki kualitas dibawah model *transformer*-KE. Kinerja *transformer*-BERT yang rendah ini dipengaruhi faktor cara pembelajaran model *transformer* yang tidak lazim. Kesimpulan tersebut dibuat berdasarkan fakta bahwa pada model rekuren, kinerja LSTM-BERT tidak berbeda jauh dibandingkan LSTM-KE, yang artinya BERT dapat digunakan dalam domain *image captioning*, namun model harus dilatih secara sekuensial dan tidak bisa memanfaatkan model berbasis *transformer*.

Terdapat keuntungan yang signifikan penggunaan model *transformer* dibandingkan model rekuren. Salah satunya dari segi kecepatan pembelajaran, derajat paralelisasi, dan memori yang digunakan. Salah satu faktor utamanya karena model tidak perlu dilatih untuk setiap tahap pembangkitan. Dari segi hasil pembangkitan *caption*, model berbasis *transformer* memiliki kinerja yang lebih baik daripada model berbasis rekuren.

Penambahan data dapat meningkatkan kinerja model. Terlihat hasil pembangkitan model yang dilatih menggunakan MSCOCO cenderung lebih baik daripada model yang dilatih dengan Flickr 8K. Jumlah kosakata yang digunakan mempengaruhi tingkat variasi *caption* yang dapat dibangkitkan. Dalam penelitian digunakan kosakata sebesar 5000 karena secara statistik kosakata 5000 cukup untuk mengakomodasi 99% token yang ada (detail pada tabel IV.2).

### **V.1.3 Eksperimen Penambahan Konteks**

#### **V.1.3.1 Skenario Eksperimen**

Model terbaik yang didapat dari eksperimen sebelumnya (*Transformer-GPT*) digunakan untuk mengevaluasi hasil dari penambahan konteks pada *caption* yang dibangkitkan. Untuk menambahkan konteks, digunakan model atribut berupa kumpulan kata (*bag of words*). *Bag of words* yang digunakan diperoleh dari penelitian Dathathri dkk. (2019).

Dalam eksperimen terdapat beberapa variabel bebas, diantaranya 4 parameter model penambah konteks (PPLM). Karena jumlah kombinasi parameter yang diujikan dapat menghasilkan permutasi dengan orde  $x^5$ , maka jumlah variasi nilai parameter dikurangi dan disesuaikan dengan kebutuhan. Detail parameter dijabarkan dalam tabel V.5.

Setiap konfigurasi model digunakan untuk menambahkan konteks pada 25 *caption* dan kemudian diamati berdasarkan beberapa metrik penilaian. Beberapa diantaranya:

1. BLEU-1 *caption* setelah ditambahkan konteks.
2. *Delta* BLEU-1 antara *caption* awal dan *caption* hasil penambahan konteks.
3. *Delta* penambahan kata dalam *bag of words* yang berhasil dimunculkan.
4. *Delta* perbandingan kemunculan kata dalam *bag of words* dengan kemunculan kata lainnya (*recall*).

Metrik tersebut dianalisis lebih lanjut untuk memilih konfigurasi parameter model yang dianggap terbaik.

**Tabel V.3 Detail Kombinasi Parameter**

Jenis parameter	Varian	Jumlah varian
<i>iteration</i>	3, 5	2
<i>lr</i>	0.005, 0.01, 0.05, 0.1, 0.3, 0.5	6
<i>optimizer_lr</i>	0.001, 0.005, 0.01, 0.03	4
<i>gamma</i>	0.6, 0.8, 0.9	3
Total		144

#### V.1.3.2 Hasil Eksperimen

Hasil eksperimen ditunjukkan pada tabel V.6. Terdapat total 144 kombinasi yang dieksperimenkan, lalu diseleksi berdasarkan beberapa kriteria yang dimulai dari 0 dan ditingkatkan sedikit demi sedikit untuk mengurangi kandidat kombinasi parameter. Kriteria akhir yang digunakan adalah sebagai berikut:

1. Nilai BLEU setelah diberi konteks  $> 0.5$ ; tujuannya untuk menjamin kualitas *caption* agar tetap baik setelah ditambahkan konteks.
2. Batasan perubahan BLEU dibandingkan BLEU sebelumnya maksimal  $-15\%$ ; dengan tujuan mencari model yang dapat menambahkan kata dengan baik tanpa risiko penurunan BLEU yang terlalu besar.

- Persen kemunculan kata baru dalam *vocab* konteks yang ditambahkan minimal 35%; tujuannya agar kalimat yang dibangkitkan cenderung menambahkan kata dalam BoW yang diinginkan, tidak sekedar mengubah *caption* yang ada.

**Tabel V.6 Hasil Eksperimen II**

No.	Parameter	B ↑	dB↑	dB%↑	dOc%↑	dRc↑	dRc%↑
1	<i>iter 3 lr 0.005 optlr 0.03 gamma 0.8</i>	0.539	-0.077	-12.59	36.0	<b>0.026</b>	<b>689.00</b>
2	<i>iter 3 lr 0.05 optlr 0.03 gamma 0.8</i>	<b>0.579</b>	<b>-0.019</b>	<b>-3.33</b>	36.0	-0.001	-15.16
3	<i>iter 3 lr 0.1 optlr 0.01 gamma 0.8</i>	0.533	-0.081	-13.29	36.0	<b>0.015</b>	<b>138.03</b>
4	<i>iter 3 lr 0.1 optlr 0.03 gamma 0.8</i>	0.551	-0.067	-10.94	<b>44.0</b>	-0.004	-39.63
5	<i>iter 3 lr 0.5 optlr 0.03 gamma 0.8</i>	0.551	-0.042	-7.2	<b>44.0</b>	0.005	45.42
6	<i>iter 5 lr 0.005 optlr 0.01 gamma 0.8</i>	0.560	-0.039	-6.51	<b>48.0</b>	<b>0.013</b>	<b>174.99</b>
7	<i>iter 5 lr 0.01 optlr 0.01 gamma 0.8</i>	<b>0.563</b>	-0.021	-3.76	<b>44.0</b>	-0.000	-4.22
8	<i>iter 5 lr 0.05 optlr 0.01 gamma 0.8</i>	<b>0.576</b>	<b>0.008</b>	<b>1.45</b>	<b>60.0</b>	0.005	45.1
9	<i>iter 5 lr 0.3 optlr 0.01 gamma 0.8</i>	<b>0.562</b>	<b>-0.004</b>	<b>-0.88</b>	<b>60.0</b>	<b>0.009</b>	<b>232.22</b>
10	<i>iter 5 lr 0.5 optlr 0.005 gamma 0.8</i>	<b>0.580</b>	<b>-0.009</b>	<b>-1.65</b>	36.0	0.002	37.85
11	<i>iter 5 lr 0.5 optlr 0.01 gamma 0.8</i>	0.534	<b>-0.017</b>	<b>-3.18</b>	36.0	<b>0.006</b>	<b>169.78</b>

\*Nilai yang dicetak biru tebal adalah 5 nilai terbaik dari masing-masing kategori.

**B** (BLEU): metrik penilaian kemiripan teks secara otomatis.

**dB** (*delta BLEU*): perubahan BLEU sebelum dan sesudah ditambahkan konteks.

**dB%** (*delta BLEU (%)*): perubahan BLEU dalam persen.



**dOc%** (*delta occurrence (%)*): perubahan jumlah kemunculan kata dalam BoW sebelum dan sesudah *caption* ditambahkan konteks, dibagi dengan jumlah *caption* yang dibangkitkan. Sebanding dengan perubahan ekspektasi kemunculan kata dalam BoW untuk setiap *caption* yang dibangkitkan, sebelum dan sesudah *caption* ditambahkan konteks.


**dRc** (*delta recall*): perubahan rasio token dalam BoW dan token di luar BoW pada *caption*, sebelum dan sesudah *caption* ditambahkan konteks.

**dRc%** (*delta recalls (%)*): *delta recall* dalam persen.

Seperti telah disebutkan sebelumnya pada bab III.2.1, berbagai macam kasus dapat terjadi saat menambahkan konteks. Beberapa contoh hasil *caption* yang dibangkitkan sebelum dan sesudah menambahkan konteks untuk masing-masing jenis kasus ditunjukkan pada tabel V.7:

**Tabel V.4 Hasil Penambahan Konteks**

Kasus 1: Berhasil dengan baik	
	<p>sebelum: <i>A kitchen with a table, chairs and a table.</i></p> <p>setelah (<i>BoW positive words</i>): <i>A kitchen with a <b>classy</b> trutharded table and a <b>large</b> window.</i></p>
	<p>sebelum: <i>A bathroom with a window and a window.</i></p> <p>setelah (<i>BoW positive words</i>): <i>A <b>charming</b> bathroom with a <b>lavish</b> window</i></p>

Kasus 2: Konteks dimunculkan namun <i>caption</i> menjadi kurang relevan	
	<p>sebelum: <i>A group of chefs preparing food in a kitchen.</i></p> <p>setelah (BoW positive words): <i>A group of people <b>entertaining</b> a <b>fabulous</b> woman in a kitchen</i></p>
	<p>sebelum: <i>A group of people sitting at a table with food.</i></p> <p>setelah (BoW positive words): <i>A group of people <b>admire</b> a table with a pizza and a knife.</i></p>
Kasus 3: Konteks tidak berhasil dimunculkan namun <i>caption</i> relevan	
	<p>sebelum: <i>A bird sitting on a wooden bench.</i></p> <p>setelah (BoW positive words): <i>A bird sitting on a wooden bench.</i></p>
	<p>sebelum: <i>A group of horses are standing in a field.</i></p> <p>setelah (BoW positive words): <i>A group of people on a horse in a field.</i></p>

Kasus 4: Konteks tidak berhasil dimunculkan, dan <i>caption</i> tidak relevan	
	<p>sebelum: <i>A plane is parked on the runway.</i></p> <p>setelah (BoW religion): <i>A Suinate Retral planeer Retpect Graorthvert Canon YearCom ForceResumblelife Chavenespherdquest.</i></p>
	<p>sebelum: <i>A man riding a motorcycle on a dirt road.</i></p> <p>setelah (BoW religion): <i>A mananing on a motorcyclelelines down a street.</i></p>

### V.1.3.3 Analisis Hasil Eksperimen

Pemilihan dan penentuan model terbaik cukup sulit untuk dilakukan karena kualitas bersifat subjektif dan metrik yang digunakan bersifat perbandingan antar model. Maka dari itu, diusulkan cara pemilihan dengan cara menyaring model dengan beberapa batasan, diantaranya memanfaatkan BLEU, kemunculan kata, dan *recall* yang dapat dihitung secara otomatis.

Berdasarkan eksperimen, dapat disimpulkan dampak perubahan masing-masing parameter dalam tabel V.8. Parameter yang dinilai paling berdampak sampai paling tidak berdampak sesuai urutannya adalah *gamma*, *optimizer\_lr*, iterasi, dan *lr*. Penambahan nilai dari semua parameter menurunkan nilai yang terkait BLEU dan menaikkan nilai yang terkait kemunculan kata dalam BoW, artinya model atribut telah berhasil mengarahkan model *image captioning* untuk membangkitkan teks yang lebih ‘diinginkan’.

Berdasarkan eksperimen juga ditemukan parameter yang memiliki nilai sama untuk ke-11 kasus ‘terbaik’ yang memenuhi kondisi; yaitu *gamma*. *Gamma* yang diperoleh memiliki nilai 0.8, hal ini sejalan dengan penelitian Dathathri dkk. (2019) yang mengusulkan nilai optimal *gamma* berkisar 0.8-0.9.

Pemilihan nilai parameter yang digunakan sangat menentukan kualitas hasil dari penambahan konteks. Penggunaan nilai yang terlampau besar menghasilkan *caption* yang menggambarkan konteks tapi tidak relevan, atau bahkan merusak *caption* secara keseluruhan. Jika nilai terlalu kecil, cenderung gagal menambahkan konteks.

**Tabel V.5 Dampak Perubahan Parameter**

No.	Parameter	B ↑	dB ↑	dB% ↑	dOc%↑	dRc ↑	dRc% ↑
1	Iterasi ↑	↓	↓	↓	↑↑	↑	↑
2	lr ↑	-	-	-	↑	↑	-
3	<i>optimizer_lr</i> ↑	↓↓	↓↓	↓↓	↑↑	↑↑	↑↑↑
4	<i>gamma</i> ↑	↓↓	↓↓↓	↓↓↓	↑↑↑	↑↑	↑↑

Dalam eksperimen ditemukan permasalahan dimana kata dalam *bag of words* (BoW) yang digunakan, seringkali belum terdaftar dalam kosakata yang digunakan sehingga harus dibuang dari BoW. Alternatif solusinya adalah menggunakan ukuran kosakata yang lebih besar (misalnya 15k), agar kata dalam BoW dapat terakomodasi.

Kualitas penambahan konteks juga dipengaruhi pemilihan konteks yang digunakan. Penggunaan beberapa jenis BoW relatif lebih efektif dibandingkan BoW lainnya, misalnya BoW kata-kata positif, lebih mudah digunakan oleh model untuk dimunculkan dibandingkan BoW yang berisi kata-kata terkait politik maupun militer. Hal ini terjadi karena pada dasarnya model atribut hanya ‘menggeser’ distribusi probabilitas kata untuk dibangkitkan. Apabila pada distribusi awal, nilai kata dalam BoW sangat kecil, maka model atribut pun kesulitan untuk mengubah distribusinya sembari menjaga ketepatan *caption*. Apabila model ‘dipaksa’ untuk



memunculkan kata dalam BoW dengan cara memperbesar nilai-nilai parameter, maka *caption* yang dihasilkan tidak lagi selaras dan *caption* yang dibangkitkan memiliki kualitas buruk (BLEU rendah).

## **V.2 Pengujian**

Tahap terakhir adalah proses menguji model yang sudah dibangun. Tujuan dari pengujian ini adalah sebagai berikut:

1. Menguji hasil pembangkitan *caption* model *image captioning* yang menerapkan *contextual word embedding* dibandingkan *non-contextual word embedding*.
2. Menguji kualitas *caption* berkonteks yang dibangkitkan model atribut.

### **V.2.1 Data Pengujian**

Data pengujian utama yang digunakan terdiri dari 20% data MSCOCO, meliputi 16557 gambar unik dan 82831 *caption*. Dalam proses pengujian juga digunakan *dataset* pembandingan Flickr 8K. Sebanyak 20% dari jumlah data yang dimiliki Flickr 8k dipisahkan untuk keperluan pengujian. Detail pembagian data ditunjukkan pada tabel V.1.

### **V.2.2 Pengujian Kinerja *Contextual Word Embedding***

#### **V.2.2.1 Skenario Pengujian**

Pengujian ini bertujuan menjawab rumusan masalah kedua. Pengujian dilakukan terhadap model terbaik yang didapatkan pada tahap eksperimen, dan dibandingkan kinerjanya dengan model *baseline* yang menggunakan *non-contextual word embedding*. Pada pengujian ini, variabel bebas yang digunakan adalah jenis *embedding* (*contextual* atau *non-contextual*) yang digunakan model. Perbandingan kinerja dilakukan dengan metode seperti pada evaluasi V.1.2. dimana model membangkitkan masing-masing 1500 *caption* yang dinilai berdasarkan nilai BLEU, METEOR, dan ROUGE-L.

#### **V.2.2.2 Hasil Pengujian**

Berikut hasil pengujian terhadap model terbaik pada eksperimen I (tabel V.9):

**Tabel V.6 Hasil Pengujian Model Terbaik pada Eksperimen I**

	<i>Dataset: MSCOCO</i>					
No.	Jenis model		Metrik			
	Model	<i>Embedding</i>	BLEU-1	BLEU-4	METEOR	ROUGE-L
1	LSTM	GPT-2	0.600	0.181	0.201	0.432
2	<i>Transformer</i>	Keras <i>Embedding</i>	0.589	0.175	0.193	0.396
3	<i>Transformer</i>	GPT-2	0.613	0.189	0.206	0.440

### V.2.2.3 Analisis Hasil Pengujian

Jika dibandingkan dengan tabel hasil eksperimen, semua metrik memiliki nilai yang hampir sama. Perbedaan hanya berkisar 0.00-0.02 terutama pada metrik BLEU-4. Nilai metrik BLEU-1, METEOR dan ROUGE-L konsisten antara eksperimen dan pengujian. Urutan kinerja model juga konsisten baik dalam eksperimen maupun pengujian. Kinerja terbaik diperoleh model *Transformer*-GPT-2, diikuti LSTM-GPT-2 pada urutan kedua serta *Transformer*-KE pada urutan ketiga. Hal tersebut menunjukkan ketiga model sudah berhasil memperoleh kinerja yang baik dan bersifat umum untuk data yang belum pernah dilihat. Hasil pengujian menekankan kembali hipotesis dan kesimpulan yang didapatkan dari proses eksperimen. Penggunaan *word embedding* kontekstual GPT-2 meningkatkan kinerja terlepas jenis model *image captioning*-nya.

Penggunaan *transformer* tidak selalu berhasil meningkatkan kinerja pembangkitan dari segi kualitas, namun dari segi kecepatan dan efisiensi saat pelatihan *transformer* jauh lebih unggul dibandingkan model rekuren. Pada dasarnya model *transformer* menggunakan konsep atensi, sehingga apabila komponen *word embedding* yang digunakan sudah berbasis *transformer* (BERT, GPT), maka perubahan dari model rekuren ke *transformer* tidak berdampak signifikan dari segi kualitas, namun untuk *word embedding* non-kontekstual (KE), penggunaan model

berbasis *transformer* berpengaruh signifikan karena dapat menambahkan informasi atensi dari *caption* yang digunakan.

Dalam pengujian model belum dapat mencapai nilai yang mengalahkan *state-of-the-art*. Hal ini salah satunya karena model yang digunakan tidak memiliki mekanisme atensi pada gambar. Gambar yang digunakan diproses oleh model *encoder* CNN tanpa mempertimbangkan *caption* yang telah dibangkitkan. Penelitian terkait menunjukkan pembuatan fitur gambar berdasarkan tahap pembangkitan *caption* dapat meningkatkan kinerja model.

Alasan mendasar yang menjadi batasan dari penggunaan atensi gambar adalah karena *hidden state decoder* pada tahap  $t - 1$  ( $H_{t-1}$ ) yang biasanya digunakan sebagai *input* model atensi gambar pada tahap  $t$ , ( $A_t = \text{atensi}(\text{gambar}, H_{t-1})$ ) tidak dapat digunakan. Hal ini disebabkan karena *decoder* dari model *transformer* dieksekusi secara ‘paralel’ ( $H_{t-1}$  dan  $H_t$  didapatkan pada tahapan yang sama) sehingga tidak bisa dilakukan pertukaran *hidden state* antar modul atensi gambar dan *decoder* layaknya pembelajaran secara sekuensial. Pada model sekuensial, komponen atensi gambar tidak bisa dilakukan karena model dibangun dan dilatih dengan Keras *Functional* API dan kondisi *dataset* yang sudah diacak. Data yang sudah diacak tidak lagi memiliki keterhubungan sekuensial. Penggunaan Keras *functional* API tidak memungkinkan pertukaran *hidden state* antar *batch* maupun antar data. Jika ingin menggunakan atensi pada gambar, maka pembelajaran harus dilakukan secara sekuensial pada setiap gambar satu per satu dan tidak dapat memanfaatkan Keras *Functional* API.

### **V.2.3 Pengujian Kualitas *Caption* Berkonteks**

#### **V.2.3.1 Skenario Pengujian**

Pengujian dilakukan terhadap dua model PPLM dengan parameter ‘terbaik’ yang dipilih berdasarkan eksperimen pada sub bab V.1.3. Model tersebut membangkitkan masing-masing 175 *caption* yang kemudian dinilai secara manual oleh manusia berdasarkan kriteria kualitas *caption* dan keberhasilan model menambahkan konteks. Penilaian dilakukan oleh tiga orang dan diambil nilai rata-rata yang dibulatkan ke angka terdekat (*majority voting*) seperti yang dilakukan Dathathri,

dkk. (2019). Metrik yang digunakan adalah *fluency* (Fl), ketepatan *caption* (Cr), dan kualitas penambahan konteks (Ctx). Perbandingan dilakukan pada *caption* yang tidak ditambahkan konteks berdasarkan metrik penilaian yang sama. Kriteria penilaian dapat dilihat pada sub bab III.5.4 dan contoh penilaian dapat dilihat pada gambar lampiran 1.

Untuk menjaga kualitas *caption* yang digunakan dalam pengujian, serta menghindari kegagalan penambahan *caption* karena kesalahan dari model *image captioning*, maka terdapat beberapa kriteria / batasan yang harus dipenuhi:

1. Gambar yang digunakan harus dapat dibangkitkan *caption*-nya dengan baik oleh model *image captioning* dengan atau tanpa model PPLM.
2. *Caption* tidak berkonteks dari gambar yang dipilih memiliki nilai BLEU-1  $\geq 0.7$  dan BLEU-4  $\geq 0.3$ .
3. *Caption* berkonteks harus memunculkan minimal satu kata dalam *bag of words* yang digunakan sebagai model atribut.

### V.2.3.2 Hasil Pengujian

Berikut hasil pengujian pada model PPLM dengan dua konfigurasi ‘terbaik’ pada tabel V.10 dan V.11.

**Tabel V.7 Hasil pengujian II (1)**

<i>iter 5 lr 0.05 optlr 0.01 gamma 0.8</i>					
Jenis BoW	<i>Caption</i> asli		Penambahan Konteks		
	Fl	Cr	Fl	Cr	Ctx
<i>Military</i>	3.34	3.24	3.04	2.40	2.88
<i>Religion</i>	3.12	2.88	3.15	2.4	2.72
<i>Legal</i>	3.33	3.23	3.18	2.63	2.70
<i>Positive Words</i>	3.26	2.86	2.91	2.66	2.66

**Tabel V.8 Hasil pengujian II (2)**

<i>iter 5 lr 0.3 optlr 0.01 gamma 0.8</i>					
Jenis BoW	<i>Caption</i> asli		Penambahan Konteks		
	Fl	Cr	Fl	Cr	Ctx
<i>Military</i>	3.12	3.04	3.04	2.37	2.89
<i>Religion</i>	3.40	2.85	3.20	2.35	2.72
<i>Legal</i>	3.23	3.23	2.95	2.65	2.60
<i>Positive Words</i>	3.43	3.14	2.89	2.66	2.71

### V.2.3.3 Analisis Hasil Pengujian

Berdasarkan pengujian, penambahan konteks menurunkan nilai *fluency* dan ketepatan *caption*. Hal ini terjadi karena jenis konteks yang ditambahkan sebenarnya tidak terlalu relevan / sesuai dengan kondisi gambar dari *dataset* yang cenderung objektif.

Penurunan nilai Fl berkisar 0.1-0.3, dan nilai Fl setelah ditambahkan konteks berkisar dari 2.9-3.1. Artinya model masih dapat menghasilkan *caption* yang secara struktur sudah baik, memiliki subjek predikat objek, namun relatif sederhana (*good*). Hasil ini mengimplikasikan walaupun terjadi pergeseran distribusi kata pada saat pembangkitan *caption*, model tetap bisa memilih kata yang paling sesuai dan menentukan waktu yang tepat untuk memunculkan kata dari BoW yang diinginkan dengan baik.

Secara intuitif dapat diprediksi terjadi penurunan nilai ketepatan *caption*. Hal ini dikarenakan model ‘dipaksa’ mengeluarkan kata yang tidak seharusnya. Terbukti dari hasil pengujian, dimana nilai ketepatan turun cukup signifikan, dengan rata-rata penurunan sebesar 0.5 (setengah indeks). Artinya *caption* yang tadinya hampir sempurna (*almost perfect*), menjadi kategori ‘ada benarnya’ (*somewhat correct*)

dikarenakan adanya penambahan kata yang ‘tidak seharusnya’ berada di dalam *caption*.

Berdasarkan nilai kualitas penambahan konteks, untuk keempat jenis BoW, model rata-rata memperoleh nilai 2.6-2.8. Artinya model berhasil menambahkan kata yang diinginkan dalam BoW. Cara penggunaan kata yang dipilih sudah relatif baik, namun belum terlalu sesuai dengan gambar (tidak ada keterkaitan antara gambar dan kata yang dipilih). Ketidaksesuaian ini disebabkan sifat *dataset* yang relatif objektif, general, dan tidak ada relevansinya dengan BoW yang digunakan.

Dari keempat jenis BoW yang diujikan, terdapat BoW yang relatif spesifik (*Military, Religion, Legal*) dan BoW yang bersifat *general (positive words)*. Namun saat dilihat pada nilai penambahan konteks, BoW yang bersifat *general* justru memiliki nilai yang lebih rendah daripada BoW lainnya. Hal ini kemungkinan disebabkan jumlah kata dalam BoW yang terlalu besar (800~). Jumlah yang besar ini menimbulkan ‘kebingungan’ saat model membandingkan distribusi yang seharusnya dengan distribusi BoW (terlalu banyak kata yang ingin dimunculkan).

Pergeseran yang terjadi juga lebih signifikan apabila banyak kata yang ingin dimaksimalkan probabilitasnya. Akibatnya model cenderung menjadi sulit merumuskan kalimat *caption* yang baik (F1 tinggi) dan tepat (Cr tinggi). Berbeda dengan BoW yang bersifat spesifik, nilai kata yang perlu dimaksimalkan tidak banyak (50~), sehingga model relatif mudah mempertahankan ‘*fluency*’ dari suatu *caption*. Variasi dari nilai F1 dan Cr untuk *caption* yang belum ditambahkan konteks dapat terjadi karena variasi gambar yang digunakan (gambar yang digunakan dapat berbeda / sama tergantung hasil penyaringan dan batasan yang dijelaskan pada sub bab V.2.3.1).

Dalam penelitian juga ditemukan batasan dari model atribut BoW. Apabila terlalu banyak token dalam BoW yang ingin dimaksimalkan, model menjadi kesulitan menentukan kata terbaik untuk dimunculkan. Hal ini terjadi karena setiap token dalam BoW memiliki bobot dan kepentingan yang sama. Permasalahan ini dapat ditangani oleh model atribut yang bersifat diskriminator (*classifier*). Pada model diskriminator, model tidak lagi membandingkan distribusi kata-per-kata (seperti

BoW) namun model membandingkan kategori atau kelas dari distribusi kata. Dalam BoW, setiap kata hanya direpresentasikan dengan 0 (tidak diinginkan) atau 1 (diinginkan); sedangkan pada kenyataanya, setiap kata dapat memiliki rentang 0-1. Walaupun tidak memiliki representasi untuk setiap kata, model atribut diskriminator memiliki ‘pembobotan’ berbeda untuk masing-masing kata dalam suatu kelas. Abstraksi ini mengakibatkan gradien yang dihasilkan oleh model atribut lebih *robust* untuk kebanyakan kasus.

## **BAB VI**

### **KESIMPULAN DAN SARAN**

#### **VI.1 Kesimpulan**

Kesimpulan yang dapat diambil berdasarkan hasil analisis yang telah dilakukan adalah sebagai berikut:

1. Berdasarkan hasil eksperimen dan pengujian, penggunaan *contextual word embedding* dari model bahasa GPT-2 meningkatkan kualitas *caption* yang dihasilkan terlepas dari arsitektur model *captioning* yang digunakan. Kinerja terbaik memiliki nilai BLEU 0.613 pada *dataset* MSCOCO menggunakan model berbasis *transformer* dengan *contextual word embedding* GPT-2.
2. Model *image captioning* yang dapat membangkitkan *caption* berkonteks berhasil dibangun dengan menambahkan komponen model atribut berupa *Bag of Words* pada model *image captioning*.
3. Penggunaan *contextual word embedding* yang bersifat *bidirectional* (BERT) tidak sesuai untuk *task autoregression* seperti pembangkitan *caption*.
4. Berdasarkan hasil pengujian, penambahan konteks dengan model atribut *Bag of Words* dapat memunculkan kata dalam konteks (BoW) dengan baik, namun menurunkan nilai BLEU dan ketepatan konteks. Kualitas akhir penambahan konteks sangat bergantung pada gambar, jenis konteks yang digunakan dan parameter model atribut yang digunakan.

#### **VI.2 Saran**

Berikut adalah beberapa saran terkait topik tugas akhir untuk pengembangan lebih lanjut.

1. Menambahkan mekanisme atensi pada *encoder* agar proses pembangkitan *caption* pada setiap tahap memiliki fokus yang berbeda-beda. Untuk model berbasis *transformer*, perlu digunakan metode tambahan untuk mengenali representasi gambar sebagai sebuah sekuens.



2. Mencoba model atribut lainnya (misalnya model diskriminator) untuk membandingkan hasil penambahan konteks.
3. Membuat daftar atribut model yang bisa dipilih secara otomatis oleh model *image captioning* (model dapat menyarankan pilihan konteks yang sesuai tanpa perlu dipilih secara eksplisit oleh pengguna).
4. Memperbesar ukuran kosakata dan kedalaman model agar semakin banyak kata-kata yang terakomodasi dan meningkatkan kualitas kinerja pembangkitan *caption* dari model.

## DAFTAR PUSTAKA

- Banerjee, S., & Lavie, A. (2005, June). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization (pp. 65-72).
- Bianco, S., Cadene, R., Celona, L., & Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6, 64270-64277.
- Bianco, S., Cadene, R., Celona, L., & Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6, 64270-64277.
- Chunseong Park, C., Kim, B., & Kim, G. (2017). Attend to you: Personalized image captioning with context sequence memory networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 895-903).
- Clark, K., Luong, M. T., Manning, C. D., & Le, Q. V. (2018). Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*.
- Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., ... & Liu, R. (2019). Plug and play language models: a simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2625-2634).
- Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

- Johnson, J., Karpathy, A., & Fei-Fei, L. (2016). Denscap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4565-4574).
- Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3128-3137).
- Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., & Socher, R. (2019). Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.
- Kiros et al. (2014a). Multimodal Neural Language Models. *Proceedings of the 31st International Conference on Machine Learning*, PMLR 32(2):595–603.
- Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74-81).
- McCann, B., Bradbury, J., Xiong, C., & Socher, R. (2017). Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems* (pp. 6294-6305).
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).
- Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 311-318). Association for Computational Linguistics.
- Pennington, J., Socher, R., & Manning, C. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. URL <https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf>.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog, 1(8).

Sharma, P., Ding, N., Goodman, S., & Soricut, R. (2018, July). Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 2556-2565).

Sow, D., Qin, Z., Niasse, M., & Wan, T. (2019, May). A sequential guiding network with attention for image captioning. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 3802-3806). IEEE.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).

Tanti, M., Gatt, A., & Camilleri, K. P. (2017). What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator? arXiv preprint arXiv:1708.02043.

Tanti, M., Gatt, A., & Camilleri, K. P. (2018). Where to put the image in an image caption generator. Natural Language Engineering, 24(3), 467-489.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., & Saenko, K. (2015). Sequence to sequence-video to text. In Proceedings of the IEEE international conference on computer vision (pp. 4534-4542).

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3156-3164).

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning (pp. 2048-2057).

## LAMPIRAN

### Lampiran A. Contoh Penilaian Pengujian 2

			
" Two dogs running on the grassy field "			
	Fl	Cr	
Resp 1	3	4	
Resp 2	4	3	
Resp 3	4	4	
result	4	4	
BoW: 'Military'			
" Two dogs <b>fighting</b> on grassy field "			
	Fl	Cr	Ctx
Resp 1	3	3	3
Resp 2	4	3	3
Resp 3	3	2	3
result	3	3	3

Responden ditunjukkan pasangan gambar dan *caption* yang akan diberi nilai sesuai kriteria pada tabel III.7. Nilai tersebut diambil rata-rata dan dibulatkan pada angka terdekat (*majority voting*).

ID	Original	FL	Crct	PPLM (BoW: Legal)	FL	Crct	Ctx
	A man riding a motorcycle on a street.	3	4	A man riding on a dorderation hazy motorcycle escorting a lawbench.	3	2	2
	A red and white bus driving down a street.	3	3	A prison bus is driving down a street.	3	2	3
	A bowl of soup and a spoon of soup.	3	2	A bowl of soup with a spoon judge isriance.	3	2	2
	A fire hydrant is sitting on the side of a road.	3	4	A fire hydrant vacony on a sidewalk near aian estate.	3	3	3
	A man riding a motorcycle on a street next to a building.	4	3	A man vacates a motorcycle on a city street.	4	3	3
	A bike is parked on the side of a road.	3	4	A bike appeal sign sitting on a capital street.	3	2	3
	A street sign with a street sign on top of it.	3	3	A street sign with a title on it on it.	3	3	3
	A man is skateboarding down a street.	3	4	A man vacates his skateboard on a sidewalk.	4	3	3
	A man riding a bike down a dirt road next to a building.	4	4	A man vacates on a bike evidence estatement bench.	4	2	4
	A man riding a wave on top of a surfboard.	3	4	A man vacates on a surfboard in the ocean.	4	3	3
	A stop sign with a street sign on it.	3	2	A stop sign with a title on ither minorony.	3	2	3
	A man holding a camera in front of a mirror.	4	4	A manequeth a titlehiprobevlitsecated a toothbrush trustbench holding a toothbrush in his hands.	1	1	1
	A woman walking down a street holding an umbrella.	4	4	A womanequins holding an umbrella record trust while walking down a street.	1	2	1
	A man is riding a bike down a street.	3	3	A man vacates a bike on a sidewalk.	4	3	3
	A man is walking on the beach with a surfboard.	4	4	A man vacates on the beach with a surfboard.	4	4	4
	A fire hydrant on a sidewalk in front of a building.	3	4	A fire hydrant vac resolutionress on a sidewalk near a record resolution.	3	2	3
	A man is riding a wave on a surfboard.	3	4	A man in a wetsuitment riding a wave in the ocean.	4	3	3
	A group of skiers standing on top of a snow covered slope.	4	4	A group of people standing on top of a snow covered slope.	4	4	4
	A clock is on the side of a building.	3	2	A clock estate with a clock on the side of it.	3	2	3
	A man is riding a motorcycle on a street.	3	4	A man vacates a motorcyclebench on a city street.	3	3	3
	A group of people standing around a table with a cake.	4	4	A group of people standing around a table with a cake.	4	4	2
	A man riding a motorcycle down a street next to a building.	4	3	A motorcycle parked on a street next to a capital building.	4	3	3
	A man sitting in a car with a surfboard on top of a car	4	4	A manexex title title resolution title in a parking lot with a surfboard on top of it.	2	3	2
	A man riding a horse drawn carriage down a street.	3	3	A man riding a horse drawn down a street next to a estate vacu claim.	4	3	2
	A man riding a motorcycle on a street with a large crowd.	4	3	A manequins on a motorcycleorder on a street with a title on it.	3	2	2
	A bowl of oranges sitting on top of a white plate.	3	3	Aex of oranges sitting on a plate with a jailrob appearance.	1	1	1
	A plane is flying in the air with a smoke trail.	3	2	A plane flying in the sky over aparlaw.	3	3	2
	A man riding a motorcycle on a dirt road.	3	3	A man riding a motorcycle down a street next to a resolution.	4	3	3
	A large airplane sitting on top of a runway.	3	3	A large resolution airplane vacates on a runway.	3	3	3
	A giraffe standing next to a tree branch.	3	4	A giraffe standing next to a tree branch in a forest.	4	3	3
	A computer monitor sitting on top of a desk.	3	4	A computer monitor title and a mouse on a desk.	3	3	3
	A man sitting on a motorcycle in the street next to a motorcycle.	4	2	A man vacates a motorcycle on a street corner.	3	2	3
	A large jetliner flying high over a blue sky.	3	3	A large resolution airplane flying in the sky over a capital.	4	4	4
	A cat sitting on a keyboard next to a keyboard.	2	2	A computer keyboard and a mouse on a bailorder.	3	3	2
	A man sitting on a bench with a skateboard.	4	2	A man vacates a bench on a benchtfoot.	3	3	3
	A man is riding a boat on a lake.	3	2	A man vaconylaw defense on a boat in the waterhip.	3	2	3
	A clock tower with a clock on top of it.	3	4	A clock tower estate with a clock tower on the side of it.	4	3	4
	A man is sitting on a motorcycle in a parking lot.	4	3	A manequinsequinsarrates to a motorcycle defense officer on a motorcycle.	2	2	2

ID	Original	FL	Crct	PPLM (BoW: Legal)	FL	Crct	Ctx
	A man laying on a bed with a laptop computer on top of a bed.	4	2	Arial and whitej stay on a bed in a living room.	3	2	2
	A motorcycle parked on a sidewalk next to a building.	3	3	A motorcycle parked on a street rulesury in a parking lot.	3	2	2
		3.33	3.23		3.18	2.63	2.70
	A plate of food with a salad and a glass of wine.	3	4	A bail resolution with aarsle of food and aianrobuses on it.	1	1	1
	A man on a motorcycle is sitting on a motorcycle.	3	2	A manequinsbench with a title on the back of a motorcycle.	2	2	2
	A large jetliner flying high in the sky.	3	4	A large bail plane flying high over a citybench.	3	3	3
	A motorcycle parked on a street next to a building.	3	3	A motorcycle appealexleges a motorcycle parked on a sidewalk.	2	3	2
	A double decker bus is driving down the street.	3	4	Aeloot reversement bus is parked on a city street.	2	2	2
	A large commercial airplane flying in the sky.	3	4	A large resolution airplane is flying in the air.	3	3	3
	A dog is looking out the window of a car.	3	4	A dog vacates a rear view mirror of a car.	3	3	3
	A dog is sitting on a bed with a white blanket.	4	3	A dog vacles on a bed prison bed with a white dog.	2	2	2
	A large airplane is on the runway at an airport.	3	4	A large passenger jetexonyparral plane on a runway estate.	3	2	3
	A man riding a motorcycle down a street next to a crowd.	4	3	A maneques a motorcycle with a resolution on the back of his motorcycle.	3	2	3
	A giraffe standing on top of a lush green field.	3	4	A giraffe standing in a grassy field.	3	4	2
	A man riding a motorcycle down a street next to a motorcycle.	3	3	A man vaconyting a motorcycle on a street with a title on it.	4	3	3
	A plate with a sandwich and a salad on it.	3	2	A plate of food with abery boiledict minor estateichony and arial of a judge.	4	2	3
	A man is riding a motorcycle in the street.	3	3	A man vacates his patrol motorcycle on a street corner.	4	3	3
	A cat is sitting on a bed in a room.	3	2	A catequal cat is sitting on a jail chair.	3	2	3
	A motorcycle parked on a road next to a building.	4	3	A motorcycle parked on the side of a road vacictced by a bailer.	4	2	3
	A man riding a motorcycle on a street.	3	4	A manequed to a prison motorcycle on a street.	3	3	4
	A chocolate cake with a slice of chocolate on a plate.	4	4	A chocolate cake with a judgeroboony reverse facecase on top of a plate.	3	3	3
	A giraffe standing in the grass next to a tree.	3	3	A giraffe standing in the grass near a tree.	3	3	2
	A giraffe standing in a grassy field with a small giraffe in the background.	4	3	A giraffe standing in the grass near a tree.	3	4	3
	A street scene with a traffic light on the side of a road.	4	4	A capital street with a lot of cars and a lot of traffic.	4	3	4
	A giraffe standing in the grass near a tree.	3	3	A giraffe standing in the grass near a tree.	3	3	3
	A red fire hydrant sitting on the side of a road.	3	4	A red fire hydrant sitting on a sidewalk capital in a city.	4	4	4
	A double decker bus driving down a street next to a tall building	4	3	A prison bus driving down a street next to a building.	4	3	3
	A giraffe standing next to a small giraffe.	3	3	A group of giraffes standing next to each other.	3	3	3
	A giraffe standing next to a tree in a field.	3	2	A giraffe standing next to a tree prison.	3	2	2
	A red and white bus driving down a street.	3	3	A sequ prison bus is driving down a street.	3	3	3
	A giraffe standing next to a tree in a zoo enclosure.	4	4	A giraffe is standing in the grass near a fence.	4	4	3
	A cat sitting on a wooden bench in a chair.	3	3	A cat sitting on a sequitylaw chair in a sequsy resolution.	2	3	2
	A giraffe standing in a grassy field with trees	3	4	A giraffe standing in the grassparorder with aparafest in its mouth.	3	2	2
	A bench with a bench in the grass next to a tree.	3	3	A bench prison with aance of a judge and a sequocket of trees.	2	2	2
	A bus is driving down a street with a bus on the side.	2	3	Ariality evidence of a bus vacoringhipment on a city street.	1	2	1
	A man standing next to a fire hydrant.	3	2	A man standing next to a bailethcourt bailortprosecurnter.	2	1	1



ID	Original	FL	Crct	PPLM (BoW: Legal)	FL	Crct	Ctx
	A bus is driving down a street with a car in the backgrou	3	3	Arial buses parked on a street near a building with a title.	3	2	2
	A bus is driving down a street near a building.	3	3	A prison bus driving down a street next to a building.	3	2	3
	A double decker bus driving down a street.	3	4	A red double decker bus driving down a street capital.	4	4	3
	A plane is flying over the water with a plane in the background.	4	3	A plane is flying in the air with a title on it.	3	3	3
	A traffic light is on a pole in a city.	3	3	A stop light evidence estate with a title and aher left.	3	3	2
	A street sign and a traffic light on a city street.	3	3	A capitalian venue with a title and abar with a title and a traffic light.	3	3	3
	A man is standing on a stage with a skateboard.	4	3	A man vacates a bailescorder on a sequrobocket.	3	2	2
		3.23	3.23		2.95	2.65	2.60

ID	Original	FL	Crct	PPLM (BoW: Military)	FL	CR	CTx
	A man riding a motorcycle on a street	4	3	A man riding a motorcycle down a street next to a sailor	3	2	3
	A man in a suit is holding a smart phone.	3	2	A man is holding a sailor in the cavalry's hand.	3	2	3
	A man is riding a bike in the street.	3	3	A man riding a bike down a street next to a combat sign.	4	2	3
	A man is standing on a bike with a surfboard.	4	3	A man is standing on a sailor's combatsection.	2	2	3
	A man walking down a street with a skateboard.	4	4	A man walking down a street next to a commander.	3	2	4
	A man is standing on a sidewalk next to a building.	4	3	A man is standing on a sidewalk next to a fire hydrant.	3	3	2
	A man is riding a motorcycle on a street.	4	3	A man is sitting on a motorcycle with a mortar in his hands.	4	2	3
	A fire hydrant is sitting on the side of a road.	4	4	A fire hydrantmaster is standing on a sidewalk.	3	2	3
	A man walking down a street while talking on a cell phone.	3	3	A man standing on a sidewalk next to a weapon.	3	2	2
	A cat laying on a bed with a blanket on top of it's head.	4	3	A cat sitting on top of a bed next to a mortar squad.	4	2	2
	A man is skateboarding down a street.	3	4	A man is mortarting a skateboard on a sidewalk.	2	2	3
	A man riding a bike down a dirt road next to a building.	4	3	A man riding a bike down a street next to a sailor.	4	3	3
	A baseball player swinging a bat at a pitch.	3	3	A baseball player swinging a bat at a ball.	3	2	3
	A man holding a camera in front of a mirror.	4	3	A man holding aotiter in his hand in front of a mirror	2	2	2
	A man is riding a bike down a street.	4	4	A man riding a bike down a street next to a sailor.	3	2	3
	A fire hydrant on a sidewalk in front of a building	3	4	A fire hydrant sitting on top of a battery box.	4	2	3
	A man is riding a wave on a surfboard.	3	4	A man commander in a wetsuit riding a wave in the ocean.	3	3	3
	A group of people sitting at a table with wine glasses.	3	3	A group of people sitting at a table with wine glasses.	4	3	3
	A man brushing his teeth with a toothbrush in his mouth.	4	3	A mortar with a toothbrush and toothbrush in it's mouth.	2	2	2
	A man in a red shirt is on a skateboard.	4	3	A man combatfight on a skateboard in a park.	3	2	4
	A man holding a pair of scissors up to his face.	4	3	A man holding aadmarks of a submarine.	3	2	3
	A clock is on the side of a building.	3	4	A clock on a pole with a commander on it.	3	2	3
	A man riding a motorcycle down a street next to a building.	3	4	A man riding a motorcycle down a street next to a battery.	3	2	1
	A living room with a couch, chair, and television.	3	3	A living room with a couch, couch, and a window.	1	3	3
	A plane is flying in the air with a smoke trail.	4	3	A plane is flying in the air with smoke coming from it.	4	2	3
	A large commercial airplane on a runway with a large airport in the background.	4	3	A large jetliner sitting on top of a runway.	3	3	3
	A small airplane is flying high in the air.	3	3	A small airplane is flying in the air.	3	3	3
	A man riding a motorcycle on a street with a large crowd.	4	3	A man riding a motorcycle down a street next to a sailor.	4	2	3
	A plane is flying in the air with a smoke trail	3	3	A plane flying over a bomb spear hospital.	3	3	4
	A man is riding a bike in the dirt.	3	3	A man riding a bike down a street next to a sailor.	4	3	3
	A man is talking on a cell phone while talking on a cell phone.	3	3	A man combatmus talks on a cell phone while sitting on a couch.	3	2	3
	A giraffe standing next to a tree branch.	4	3	A giraffe standing next to a tree weapon in a forest.	3	2	3
	A cat sitting in a window looking out of a window.	4	4	A cat sitting in a window looking out of a window.	2	2	1
	A motorcycle is parked on a street near a building.	3	3	A motorcycle parked on a street near a building with a weapon on it.	4	3	3
	A dog laying on a bed with a white blanket.	3	4	A dog laying on a bed with a insignizedjet's paw on it's head.	3	3	3
	A cat sitting on a keyboard next to a keyboard.	2	3	A mortar and a computer monitor on a desk.	3	2	3
	A man sitting on a bench with a skateboard.	3	2	A man sitting on a bench in a park with a weapon on his back.	3	3	3
	A man riding a bike down a dirt road	3	3	A man riding a bike down a street next to a weapon.	4	2	3
	A man laying on a bed with a laptop computer on top of a bed.	4	4	A woman laying on a bed with a mortar in her hand.	3	3	4

ID	Original	FL	Crct	PPLM (BoW: Military)	FL	CR	CTx
	A motorcycle parked on a sidewalk next to a building.	3	3	A motorcycle is parked on a street with a weapon on it.	4	3	3
	A plate of food with a salad and a glass of wine.	3	4	A table topped with a plate with a weapon of food and afireforce.	3	3	4
	A large airplane is parked on the runway.	4	3	A large commercial airplane on a runway with afire coming from it's landing.	3	2	3
	A large airplane flying in the sky over a blue sky.	2	4	A large airplane is flying high in the sky.	3	3	1
	A car is parked in a lot with a car on the side of it.	3	3	A fire hydrantcar parked on the side of a road.	2	3	3
	A dog is looking out the window of a car.	3	3	A dog is combat in the street force of a car in the street.	2	3	3
	A dog is sitting on a bed with a white blanket.	3	3	A dog sitting on a bedoman's bed in front of a white dog.	3	3	2
	A vase filled with pink flowers in a vase.	2	3	A mortarinescrew with a killiterfight in a hospital.	1	1	3
	A large airplane flying in the sky over a blue sky.	2	3	A weapon is flying over a plane on a runway.	3	3	4
	A group of people riding motorcycles down a street.	3	4	A man squad of people riding amphibers down a street.	4	2	3
	A motorcycle parked on a sidewalk next to a car.	4	3	A motorcycle parked on a sidewalk next to a squad of combat bikes.	3	3	3
		3.34	3.24		3.04	2.4	2.88
	A large airplane is on the runway at an airport.	3	4	A large commercial airplane on a runway with aparidding on it.	3	2	3
	A man is sitting on a motorcycle in a parking lot.	3	3	A man combattrobow is capture on a motorcycle.	3	3	2
	A cake with a red and white frosting on it.	4	2	A veteran mortar marines cake with a bomb on it.	3	2	4
	A table with a wine glass and a wine glass.	2	3	A table topped with a weapon and a glass of wine.	3	2	3
	A man standing next to a bus on a street.	4	4	A cavalry marines and a weapon shoot at a red bus.	3	2	3
	A man riding a motorcycle down a street next to a crowd.	3	3	A man riding a motorcycle on a street next to a captain.	4	2	4
	A man standing next to a large airplane.	3	4	A man standing next to a large airplane on top of a cement field.	3	2	3
	A man riding a motorcycle down a street next to a motorcycle.	2	3	A man squad riding on a motorcycle on a street.	4	2	4
	A cat sitting on a window sill looking out of the window	3	4	A cat is looking out of the window.	3	2	3
	A plate with a slice of cake and a spoon on a plate.	3	3	A plate with a missile and a spoon with a spoon.	3	2	3
	A giraffe standing in a grassy field with a small giraffe in the background.	3	3	A giraffe standing in the grass with a weapon in its mouth.	3	2	3
	A street scene with a traffic light on the side of a road.	3	3	A traffic lightotformation with a red light on the side.	3	3	3
	A red fire hydrant sitting on a sidewalk next to a sidewalk.	2	4	A red and white fire hydrant sitting on a sidewalk.	4	2	2
	A fire hydrant in front of a building.	4	3	A fire hydrant sitting on a sidewalk next to a veteran.	2	3	3
	A red fire hydrant sitting on the side of a road.	4	4	A red fire hydrantmaster is standing on a sidewalk.	2	3	1
	A double decker bus driving down a street next to a tall building.	3	3	A convoy is driving down a city street.	3	3	3
	A man standing on a sidewalk next to a building.	4	4	A man is walking down a sidewalk squad.	4	2	3
	A giraffe standing next to a small giraffe.	3	3	A giraffe standing next to a weapon in a field.	3	2	3
	A fire hydrant on a sidewalk near a city street	3	3	A bench with a weapon on it is on the side of a road.	4	2	3
	A giraffe standing next to a tree in a zoo enclosure.	3	3	A giraffe standing next to a weapon in a field.	3	2	3
	A man standing next to a fire hydrant.	3	4	A mortarmaster is standing in front of a fire hydrant.	3	2	3
	A dog is sitting on a bed in the snow.	2	3	A dog is standing on a cavalry cart squad of people in the snow.	3	2	4
	A man standing next to a fire hydrant.	3	4	A man standing next to a fire hydrant.	4	4	3
	A large airplane flying in the sky with a sky in the background.	4	3	A large commercial airplane is parked on the runway.	4	3	3
	A red and white bus is driving down a street.	3	3	A bus is driving down a street with a sailor on it.	4	2	3
	A large passenger jet sitting on top of an airport tarmac	3	3	A large jetliner sitting on top of an airport tarmac.	3	3	3

ID	Original	FL	Crct	PPLM (BoW: Military)	FL	CR	CTx
	A man standing next to a fire hydrant near a fire hydrant.	4	3	A commander is standing next to a fire truck.	4	2	3
	A giraffe standing in the grass next to a tree.	3	3	A giraffe standing in the grass with a weapon in its mouth.	3	3	3
	A man sitting on a bench with a skateboard.	3	3	A man sitting on a benchplainmaster bench next to a weapon.	2	2	3
	A large white and blue bus driving down a street.	4	3	A commander is standing in front of a sailor on a street.	3	3	4
	A group of people sitting on top of a bench.	3	4	A man sitting on a bench with a mortar in his hand.	4	3	3
	A large airplane is parked on the runway at an airport.	3	3	A large commercial airplane on a runway with a mortar on it.	3	3	4
	A giraffe standing in the middle of a field with a small giraffe.	3	4	A giraffe standing in the grass with its neckshot's head in the air.	3	2	3
	A street scene with a traffic light and a street sign.	4	3	A street commander directing traffic at a cross walk.	3	3	3
	A bus is driving down a busy street.	4	3	A bus is driving down a street with a weapon on it.	4	2	3
	A woman walking down a street with an umbrella.	3	3	A woman holding an umbrella squad walking down a street.	2	3	3
	A street with a lot of traffic on it.	3	3	A street with a commander directing traffic at a crosswalk.	3	2	3
	A fire hydrant sitting on the side of a road.	3	3	A mortar and a fire hydrant on a sidewalkhip.	2	3	3
	A man sitting on a bench with a skateboard.	3	3	A man sitting on a bench next to a weapon.	3	3	3
	A man sitting on a bench next to a bench.	2	3	A man sitting on a bench in a park with a weapon on top of it.	3	3	3
	A street sign with a street sign on it.	3	3	A mortar and street sign with a street sign on it.	3	3	3
	A man is standing next to a fence in a park.	3	3	A cavalry is standing in a field with a weapon on the back.	3	3	3
	A man is standing on a stage with a skateboard.	3	3	A man is mortarting a sailor on a stage.	3	2	3
	A fire hydrant on a sidewalk next to a building.	4	3	A veteran sailor is standing on a bench near a building.	4	3	3
	A bird is perched on a tree branch.	3	3	A bird sitting on a branchcrew on a branch.	2	3	2
	A man walking down a street with a skateboard.	3	3	A manic squad is walking down a street with a sailor on the back of a bike	3	2	3
	A small bird is perched on a branch of a tree.	4	2	A bird is sitting on aadmaster's commander's hand.	2	1	3
	A train is going down the tracks in a rural area.	4	3	A train is combatreating a cavalryteam on a railroad track.	2	2	3
	A man laying on a bed with a remote control.	3	3	A man laying on a bed with aicscented combat gear.	4	3	3
	A man is standing on a street corner next to a stop sign.	3	3	A man is battlemarks on a stop sign.	3	2	3
	A man wearing a hat talks on a phone while talking on a cell phone.	2	3	A man combat in a quarter of asection of a battle on a cell phone.	3	3	3
	A train is traveling down the tracks near a building.	3	3	A train is traveling down the tracks commandermarks.	2	3	3
	A train is traveling down the tracks near a building.	3	2	A train is commander on the tracks near a train station.	4	1	3
	A swans swimming in the water near a lake.	3	2	A duck combat swimming in the water with a swans carrier in the background	3	2	3
	A train is traveling down the tracks near a train station.	2	3	A train is shelled by a train yard.	4	3	2
	A group of people walking down a street with umbrellas.	3	3	A group of peoplecrafting mortar andcrew on a city street.	2	2	3
	A train is on the tracks near a platform.	4	3	A train is traveling down the tracks near a captain.	3	2	3
	A train is traveling down the tracks near a building.	3	2	A train is going down the tracks with a captain.	3	2	3
	A bird sitting on a branch of a tree branch.	4	3	A bird is perched on a branchcrew	3	3	3
	A fire hydrant sitting on top of a sidewalk next to a street.	3	3	A fire hydrant sitting on the side of a road.	3	2	2
	A blue and white bird standing on a beach.	3	2	A blue and black bird standing on aadformation.	3	2	1
	A dog with a hat laying on a bed.	3	3	A dog laying on a couch with a mortar in it's mouth.	2	2	3
	A street sign is on a corner of a street.	3	2	A commander is walking draftoman on a street corner.	3	1	3
	A stop sign with a street sign on it.	3	3	A street signot on a pole in front of a building	3	2	1
	A stop sign on a street corner with a stop sign on it.	2	2	A stop sign on a pole in acrafttro leave a parking lot.	3	1	3

ID	Original	FL	Crc	PPLM (BoW: Military)	FL	CR	CTx
	A train is traveling down the tracks near a large mountain.	3	3	A train is combatly traveling down the track.	4	3	1
	A street sign with a street sign on it.	3	3	A street sign with afire on it.	2	3	3
	A street sign with a street sign on it.	3	3	A street sign with a veteran capture of a street sign on it.	3	3	3
	A train is traveling down the tracks near a building.	4	3	A train is on the tracks near a captain.	3	3	3
	A man walking down a street with a bicycle	3	3	A veteranvertot shop with a lot of people walking down a city street.	3	2	3
	A train is on the tracks near a building.	3	2	A train is going down the tracks near a captain.	4	2	3
	A bird is sitting on a wooden bench.	3	3	A bird is sitting on a cavalry chair spear.	2	3	3
	A street sign with a red arrow on it.	3	3	A street sign with a red arrow on it.	3	2	3
	A train is traveling down the tracks near a building.	4	4	A train is going down the tracks near a weaponally large veteran.	2	2	3
	A street sign with a street sign on it on it.	2	3	A street sign with afire on it.	3	2	2
		3.11	3.04		3.04	2.37	2.89

ID	Original	FL	Crct	PPLM (BoW: Positive words)	FL	Crct	Ctx
	A kitchen with a sink and a sink in it.	3	2	A kitchen with a stove top oven and a sink.	3	2	1
	A kitchen with a sink and a sink.	3	2	A kitchen with a sink, sink, and a window.	3	2	1
	A man in a suit is holding a smart phone.	4	3	A man in a suit is holding a smart phone.	4	3	4
	A man riding a wave on top of a surfboard.	3	3	A man riding a wave on top of a surfboard.	3	3	3
	A white and blue cat is sitting on a window sill.	4	3	A cat sitting on a window sill in a house.	3	3	3
	A man riding a wave on top of a surfboard.	4	3	A man riding a wave on top of a surfboard.	3	3	4
	A street sign with a street sign on top of it.	3	2	A stop sign with a street sign on it.	3	2	3
	A man standing next to a black and white photo of a truck	3	3	A man standing next to a black and white photo of a truck.	3	3	3
	A man riding a bike down a dirt road next to a building.	4	3	A man riding a bike down a street next to a trendy building.	4	3	4
	A man riding a horse drawn carriage down a street.	4	3	A man riding a horse down a street next to a trendy building.	4	3	3
	A man riding a bike on top of a bike.	2	2	A man riding a bike down a street next to a trendy woman on a bicycle.	4	4	3
	A kitchen with a stove top oven and a microwave.	3	3	A kitchen with a variety ofAAAAAA stove and a	2	2	2
	A kitchen with a sink and a sink in it.	3	3	A kitchen with a stove and a trophy.	3	2	3
	A washer in a bathroom with a toilet in it.	3	3	A small bathroom with a white refrigerator and a fast washer.	3	2	3
	A man wearing a hat talks on a phone while talking on a cell phone.	3	3	A man happilyently talking on a cell phone.	2	3	3
	A man riding a wave on top of a surfboard.	4	4	A man engaging on a surfboard in the ocean.	4	3	4
	A bicycle is parked on a sidewalk near a building.	3	3	A bicycle is available enough on a sidewalk.	2	3	2
	A man riding a motorcycle on a street next to a building.	4	2	A sleek stylA stylized motorcycle worked clearly clearly on the side of a motorcycle.	2	2	1
	A horse standing next to a horse in a field.	2	2	A stabledesA horse led to a horseralperole in a field.	1	1	1
	A man riding a motorcycle on a street.	3	3	A man riding a patriotic motorcycle down a street.	4	4	3
	A toilet with a toilet and a roll of toilet paper on it.	3	3	A toiletainapationuredains shinefast in a bathroom with a toiletaseger.	1	1	2
	A living room with a couch, chair, and television.	3	3	A living room with a couchworkured window and aasA deliciousionureduredpleasAiA	1	2	2
	A man riding a motorcycle on a street.	3	3	A man riding a patriotic motorcycle down a road.	4	3	3
	A desk with a laptop computer and a laptop computer.	2	3	A tidy desk with a laptop owsestain and a laptop.	2	2	2
	A large building with a clock on top of it.	4	3	A clock tower shineening over a lovely building.	3	3	2
	A giraffe standing next to a tree branch.	3	3	A giraffe affectionized a tree in a feA treeub	2	2	2
	A large airplane is flying in the air.	3	3	A small airplane is flying patriotic in the air.	4	2	3
	A motorcycle is parked on a street near a building.	4	4	A motorcycle is parked neatly worked on a sidewalk.	3	3	3
	A man riding a motorcycle on a street.	4	4	A man riding a patrioticized motorcycle down the road.	4	4	3
	A cat sitting on a keyboard next to a keyboard.	3	2	A sleek laptop computer sitting on a table.	4	3	3
	A living room with a couch, chair, and a television.	3	3	A living room with a lovelyasitty furniture and a grandpeastwork.	3	3	3
	A motorcycle parked on the side of a road.	3	2	A sleekikesirtifulness affA motorcycle is worked on a motorcycle.	1	1	1
	A giraffe standing next to a tree in a zoo enclosure.	4	3	A giraffe standing in a spacious area.	4	4	3
	A herd of sheep standing on top of a grass covered field.	4	3	A sheep isepperting peacefully in a field.	2	3	3
	A cake with a red and white frosting on it.	3	3	A delicious looking cake is sitting on a table.	4	4	4
		3.26	2.86		2.91	2.66	2.66
	A man is sitting on a motorcycle with a dog.	3	2	A man works on a motorcycleousel in a scenicidevel.	1	1	1
	A motorcycle parked on a road next to a building.	3	2	A man on a motorcycle is riding a patriotic leadruck.	3	2	2

ID	Original	FL	Crct	PPLM (BoW: Positive words)	FL	Crct	Ctx
	A herd of sheep grazing on a lush green field.	3	3	A herd of sheepicamasaring in a spacious pasture.	2	2	2
	A truck is parked on a city street.	3	3	A truck driving down a street helping a truck.	3	3	3
	A giraffe standing in a grassy field with a small giraffe in the background.	4	3	A giraffe standing peacefully in a field of grass.	4	4	4
	A giraffe standing in the grass near a tree.	4	3	A giraffe standing peacefully in a grassy area.	4	4	4
	A red and white bus driving down a street.	4	3	A bus is neatlyige andA bus is driving down the street.	2	2	2
	A bench with a bench in the grass next to a tree.	3	2	A lovelyapAppedple-A bench affA bench is sitting on a bench in the grandwalk	2	1	1
	A dog is sitting on a bed in the snow.	3	3	A wooppy dog is sitting on a bench in a spacious area.	3	3	3
	A herd of sheep grazing on a lush green field.	3	3	A herd of sheep standing peacefully on top of a lush green field.	4	3	3
	A bench in a park with a hammock on it.	3	3	A lovelyvelumbs sitting on a benchillowed bench in a park.	2	2	2
	A giraffe standing next to a tree in a forest.	3	2	A giraffe affectionppping peacefully in a spacious area.	3	2	3
	A giraffe standing in the grass near a tree.	3	3	A giraffe standing peacefully in a spacious field.	4	3	3
	A double decker bus is parked on the side of a road.	4	3	A double decker bus is neatlyppy andruck.	3	2	2
	A boat is sitting on a sidewalk next to a sidewalk.	3	3	A boat is sitting on a smooth wooddesresqueured area.	2	3	3
	A fire hydrant on a sidewalk next to a sidewalk.	4	4	A fire hydrant shine on a sidewalk.	3	3	4
	A giraffe standing in the grass near a tree.	4	4	A giraffe standing in a field of grass tops.	3	4	3
	A city street with a lot of traffic and cars driving down it.	4	3	A street with a traffic light shine on it.	3	4	4
	A bird is perched on a tree branch.	3	4	A bird sitting on top of a peach tree.	3	3	2
	A train is going down the tracks in a rural area.	4	4	A train is worked patiently on a track	4	3	3
	A red and white bus driving down a street	4	4	A bus is driving down a quietle.	3	3	2
	A man riding a wave on top of a surfboard.	4	4	A man engagingfullyppy bestoorication on a beautifulyvy waves.	2	2	1
	A group of people walking down a street with umbrellas.	4	3	A group of people enjoy aiA stylist is walking down a busy street.	3	3	3
	A train is on the tracks near a platform.	4	3	A train is coming down the tracks shine down the tracks.	4	3	3
	A street sign with a street sign on it.	2	2	A street sign clearlypped a stop sign.	2	2	2
	A white bird is flying over a body of water.	4	4	A seagullppy bird flying peacefully in though water.	3	3	3
	A building with a clock on the side of it.	3	4	A grand building with a fine building and a clock on the side.	4	3	4
	A street sign with a street sign on it.	2	3	A street sign indicatesized street signs available.	2	2	2
	A herd of sheep standing on top of a lush green field.	4	4	A sheep stands peacefully in a field with aparagus.	4	4	4
	A black bird is perched on a rock.	3	3	A black bird is worked on in the air.	3	2	3
	A giraffe standing in the grass near a tree.	4	2	A giraffe standing peacefully on a lushuredured green field.	3	2	3
	A bird is sitting on a wooden bench.	3	4	A small bird perched on aattericA lovely birdilyery moment.	2	2	3
	A herd of sheep grazing on a lush green field.	3	3	A wooly sheep standing in a grassy field.	4	3	4
	A train traveling over a bridge over a river.	4	4	A trainardicpped over a bridge leads into a bridge.	2	3	2
	A herd of sheep grazing on a grass covered field.	4	3	A wooly pasture with a herd of sheepestissowriczy animals in the background	2	2	2
		3.43	3.14		2.89	2.66	2.71

ID	Original	FL	Crct	PPLM (BoW: Religion)	FL	Crct	Ctx
	A man is riding a bike in the street.	4	3	A man riding a bike down a street next to a Cathedral.	4	2	3
	A man is standing on a bike with a surfboard.	3	1	A Cathedral with a Bibleiver, a bicycle rack, and a bike Commode on the back of it.	1	1	2
	A man walking down a street with a skateboard.	3	3	A man walking down a street with a Cathedral in the background.	4	3	4
	A man is standing on a sidewalk next to a building.	3	3	A man is standing on a sidewalk with a Bible on it.	3	2	3
	A man riding a motorcycle on a street next to a building.	4	3	A Cathedral with a motorcycle parked on top of it.	3	2	3
	A man walking down a street while talking on a cell phone.	3	3	A man is walking down a street with a Bible on it.	4	2	3
	A cat laying on a bed with a blanket on top of it's head.	4	3	A cat laying on top of a bed with a Bible on it's head.	4	3	3
	A street sign with a street sign on top of it.	2	3	A street sign with a Bible on it on it.	3	2	2
	A man standing next to a black and white photo of a truck.	3	2	A man standing next to a black and white photo of a Bible.	4	2	3
	A man riding a bike down a dirt road next to a building.	3	3	A man is riding a bike with a Bible on it.	3	3	3
	A man riding a bike down a street next to a building.	3	3	A man riding a bike down a street Bible covered street next to a Cathedral.	2	2	3
	A stop sign with a street sign on it.	2	2	A Bible sign with a street sign on it.	2	2	2
	A man is walking on the beach with a surfboard.	3	4	A Bible Choise isations of a man on a surfboard in the ocean.	1	2	1
	A man is riding a motorcycle on a street.	3	3	A man riding a motorcycle on a street next to a Cathedral poster.	4	2	2
	A fire hydrant on a sidewalk in front of a building.	3	3	A red fire hydrant sitting on top of a Cathedral street.	4	3	3
	A knife and a knife on a cutting board.	2	2	A Bible cutting board with a knife on a cutting board.	3	2	2
	A man holding a pair of scissors up to his face.	3	3	A man with a beard holding a Bible in his mouth.	3	2	3
	A clock is on the side of a building.	3	3	A Bible clock is on a pole in a park	2	2	2
	A cat laying on a bed next to a white and white cat.	2	2	A cat laying on a bed next to a Bible.	3	2	3
	A living room with a couch, chair, and television.	4	4	A living room with a couch, a Bible and a couch.	3	3	3
	A busy city street with cars and a lot of cars.	3	3	A Cathedral with a lot of cars and a lot of people walking by.	3	3	3
	A man riding a motorcycle on a street with a large crowd.	3	3	A man riding a motorcycle on a street next to a Cathedral.	4	3	3
	A bowl of oranges sitting on top of a white plate.	4	3	A bowl of oranges sitting on a table next to a Bible.	4	3	3
	A plane is flying in the air with a smoke trail.	4	2	A plane flying over a Sunday sky filled sky.	2	2	3
	A bowl of oranges sitting on top of a white plate.	3	3	A Cathedral with a bunch of oranges on top of it.	3	2	3
	A busy city street with cars and a lot of traffic.	4	4	A Cathedral with a taxi and a lot of cars on it.	3	3	3
	A table topped with a vase of flowers on top of a table.	3	4	A Bibleiety is shown on a table with a vase of flowers.	2	3	2
	A large airplane is flying in the air.	3	3	A plane flying in the air with a Bible on it's tail.	3	3	3
	A man sitting on a motorcycle in the street next to a motorcycle.	2	2	A man sitting on a motorcycle with a Bible on it.	3	2	3
	A living room with a fireplace, a television, and a television.	3	3	A living room with a Bible, a television, and a couch.	4	4	3
	A dog laying on a bed with a white blanket.	4	3	A dog laying on a bed with a Bible on it's back.	4	3	2
	A cat sitting on a keyboard next to a keyboard	3	1	A Bible book is on a desk next to a computer	3	1	2
	A man is riding a motorcycle on a street.	3	3	A man riding a motorcycle on a street Bible store.	3	2	2
	A man is sitting on a motorcycle in a parking lot.	3	2	A man riding a motorcycle down a street next to a Cathedral.	4	3	3
	A man riding a motorcycle on a street.	3	3	A man riding a motorcycle on a street next to a Cathedral.	4	3	3
	A man riding a bike down a dirt road.	3	3	A man riding a bike down a dirt road next to a Cathedral.	4	3	3
	A man laying on a bed with a laptop computer on top of a bed.	4	2	A Bible is laid on a bed with a Bible on it.	3	2	3
	A motorcycle parked on the side of a road.	3	2	A motorcycle parked on a street with a Bible on it.	3	2	2



ID	Original	FL	Crct	PPLM (BoW: Religion)	FL	Crct	Ctx
	A motorcycle parked on a sidewalk next to a building.	3	2	A Cathedral with a motorcycle parked on the side of it.	4	2	3
	A plate of food with a salad and a glass of wine.	4	4	A Biblehedral table topped with a variety of food on it.	3	3	4
		3.125	2.775		3.15	2.4	2.725
	A large airplane is parked on the runway.	4	4	A large Bible jumbo jet sitting on top of a runway.	3	3	3
	A motorcycle parked on a street next to a building.	3	3	A Cathedral motorcycle parked on a sidewalk next to a building.	3	2	2
	A group of people riding motorcycles down a street.	3	3	A Cathedral with a lot of people walking down a street.	3	3	3
	A motorcycle parked on a sidewalk next to a car.	4	3	A motorcycle parked on a street next to a Cathedral.	3	3	3
	A man is sitting on a motorcycle in a parking lot.	3	3	A Cathedral motorcycle is parked on a sidewalk near a building.	3	2	2
	A man standing next to a bus on a street.	3	2	A Cathedral with a bus and a bus on the side.	2	1	2
	A man riding a motorcycle down a street next to a crowd.	4	3	A man riding a motorcycle on a street with a Cathedral in the background	4	3	3
	A table with a cup of coffee and a cup of coffee.	3	2	A Bible and a cup of coffee on a table next to a cup of coffee.	3	2	3
	A man riding a motorcycle down a street next to a motorcycle.	3	2	A man sitting on a motorcycle with a Bible on it.	3	2	3
	A man is riding a motorcycle in the street.	4	4	A motorcycle parked on a street with a Bible on it.	3	2	3
	A motorcycle parked on a road next to a building.	3	2	A Cathedral with a motorcycle parked on the side of a road.	3	2	3
	A man riding a motorcycle on a dirt road.	4	4	A man riding a motorcycle on a street Lightvert.	3	3	3
	A man riding a motorcycle on a street with a woman on the back.	4	3	A man is riding a motorcycle with a Bible on it.	3	2	3
	A man and woman sitting on a bench.	4	4	A man sitting on a bench in a park with a Bible on it.	4	3	3
	A red fire hydrant sitting on the side of a road	4	4	A red fire hydrant sitting on a sidewalk next to a Cathedral.	4	3	3
	A man standing on a sidewalk next to a building.	4	3	A building with a Bible on it is shown in a large city.	3	2	3
	A bench with a bench in the grass next to a tree.	2	4	A Bible sitting on a bench in a park near a park.	2	3	3
	A man standing next to a fire hydrant.	3	2	A man Musicvert is standing next to a fire hydrant.	2	2	2
	A man sitting on a bench next to a body of water.	3	2	A Cathedral bench with a Bellan tree in the background.	4	3	3
	A fire hydrant sitting on a sidewalk next to a street.	4	4	A red fire hydrant sitting on top of a Cathedral.	3	2	3
	A large white bird standing on top of a lush green field.	4	3	A Bible that is Godond sheep standing on top of a rock.	2	1	2
	A giraffe standing in the grass near a tree.	3	3	A giraffe standing in a grassy field with a Cathedral in the background	3	2	4
	A bench in a park with a hammock on it.	3	2	A bench in a park with a Bible on it.	3	2	3
	A man sitting on a bench in the snow.	4	4	A Bible book sitting on top of a bench in a park.	4	3	2
	A man standing next to a fire hydrant near a fire hydrant.	2	2	A Cathedral firetruck is parked on a street near a building.	3	2	2
	A man is skateboarding on a bridge with a train in the background.	3	1	A Bible scene with a bridge over a river.	3	2	2
	A man sitting on a bench with a skateboard.	4	1	A man sitting on a bench in a park with a Bible on his back.	4	1	3
	A large white and blue bus driving down a street.	4	3	A Salvation Angel is driving down a street.	3	3	3
	A man sitting on a bench in a park.	4	3	A Bible sitting on a bench in a park.	3	2	3
	A boat is sitting on the side of a river.	3	3	A boat is sitting on the water near a Cathedral.	4	3	3
	A sheep is standing in a grassy field.	3	3	A sheep stands in a grassy field with a Cathedral in the background.	4	3	3
	A man sitting on a bench in a park.	4	3	A man sitting on a bench in a park with a Bible on it.	4	3	3
	A street with a street sign and a traffic light.	4	3	A Cathedral with a street sign and a traffic signal on it.	4	2	3

ID	Original	FL	Crct	PPLM (BoW: Religion)	FL	Crct	Ctx
	A clock is on a pole in a city	3	4	A Bible clock is seen on a pole in a city.	2	3	3
	A yellow school bus is parked in a lot.	3	2	A yellow school bus parked next to a Cathedral.	4	2	3
	A fire hydrant is covered in snow and snow.	2	3	A Cathedral fire hydrant is shown in the snow.	3	3	3
	A man sitting on a bench near a body of water.	4	2	A Bible Chopper is sitting on a bench near a large body of water.	4	2	3
	A blue and white bird standing on a beach.	4	3	A blue and white bird standing on a Sunday.	4	2	1
	A street sign with a street sign on it.	2	2	A street sign with a Bible written on it.	3	2	1
	A sheep standing on top of a lush green field.	4	3	A sheeptom stands in a grassy field with a Cathedral in the background.	3	3	3
		3.4	2.85		3.2	2.35	2.725