Image Captioning For Mobile Application

Affan Taufiqur, Dhomas Hatta

Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia

18523100@students.uii.ac.id, 085230103@uii.ac.id

Abstrak--Image Captioning adalah proses untuk memproduksi teks deskripsi dari suatu gambar secara otomatis. Penggunaan image captioning dapat diaplikasikan ke banyak bidang seperti virtual assistant, image indexing, social media,* dan *berbagai macam aplikasi yang memanfaatkan Natural Language Processing. Image captioning memiliki banyak manfaat yang dapat dipakai dalam kehidupan sehari-hari, seperti edukasi untuk anak kecil, dan membantu orang yang memiliki pengelihatan kurang. Tekonologi image captioning sudah lama dikembangkan, tetapi, penggunaan dalam skala publik masih kurang. Dalam penelitian ini, saya mengembangkan aplikasi image captioning yang dapat digunakan dalam ponsel sehari-hari. Saat ini lebih dari 80% dari semua populasi di dunia memiliki ponsel pintar, yang berarti aplikasi image captioning dapat digunakan sehari-hari secara praktis. Aplikasi bekerja dengan cara pengguna memilih gambar dari ponsel mereka, dan cukup menekan satu tombol saja, caption dari gambar tersebut langsung keluar. Implementasi aplikasi image captioning langsung pada ponsel membutuhkan kinerja komputasi yang cukup tinggi, yang dimana kinerja ponsel pintar sekarang masih kurang mampu, karena itu sekarang masih sedikit aplikasi image captioning yang dapat digunakan di ponsel pintar. Pada penelitian ini, saya mengembangkan aplikasi image captioning yang dapat digunakan di ponsel dengan menggunakan model yang sudah dikembangkan dari keras, model tersebut menggunakan CNN sebagai yang mengambil fitur pada gambar, transformer sebagai input dan output, dan menggunakan Flickr8K_dataset sebagai gambar untuk melatih model, hasil latihan dari model ini akan digunakan sebagai model akhir yang digunakan untuk menghasilkan caption. Model ini mendapatkan BLEU score sebesar 0.31 setelah di test menggunakan 15 gambar yang diambil dari validation sets. Model akan diunggah ke server dan aplikasi android akan melakukan komunikasi dengan model yang ada di server untuk menghasilkan caption. Dengan itu aplikasi image captioning di ponsel bisa digunakan oleh semua orang waulupun memiliki kinerja komputasi yang berbeda-beda karena semua kalkulasi di lakukan di server dan aplikasi android hanya perlu mengirim dan menerima data saja.

Kata kunci—Image captioning, Deep learning, Deep learning di smartphone, Image captioning di smartphone, Artificial Intellegence.

I. PENDAHULUAN

Image Captioning adalah proses untuk memproduksi teks deskripsi dari suatu gambar secara otomatis. *Image captioning* merupakan salah satu area di bidang *Artificial Intelligence* (AI) yang memahami gambar dan deskripsi dari gambar tersebut [1] Penggunaan *image captioning* dapat diaplikasikan ke banyak bidang seperti *virtual assistant, image indexing,* membantu orang yang memiliki pengelihatan kurang, *social media,* dan berbagai macam aplikasi yang menggunakan *Natural Language Processing* (NLP) [2].

Pengembangan aplikasi *image captioning* membutuhkan kinerja komputasi yang cukup tinggi, yang dimana kinerja ponsel pintar sekarang masih kurang mampu [3]. Hal itu menyebabkan sedikitnya aplikasi *image captioning* di ponsel pintar. Pengembangan fokus ke bentuk aplikasi *mobile* karena pengguna dan kinerja ponsel pintar yang terus bertambah dan berkembang setiap tahunnya, menyebabkan ponsel pintar menjadi suatu bagian penting dalam kehidupan manusia, dan mempermudah kehidupan penggunannya, smartphone juga *portable* dan praktis, yang memungkinkan untuk digunakan kapanpun dan dimanapun. Implementasi *image captioning* ke dalam bentuk aplikasi *mobile* dapat memudahkan orang yang memiliki pengelihatan kurang untuk membantu meningkatkan kualitas kehidupan sehari-hari. Mengimplementasikan kompleks *deep learning* model ke dalam suatu ponsel pintar memiliki banyak rintangan seperti kinerja komputasi, *latency*, energi, memori rendah, dan privasi [3]. Faktor lain yang perlu diperhatikan saat mengembangkan sebuah aplikasi yang menggunakan AI sebagai dasarnya, adalah kemampuan dari smartphone itu sendiri, karena setiap orang memiliki ponsel dengan spesifikasi yang berbeda-beda, perbedaan dalam kinerja ini nantinya akan mempengaruhi *user experience* dalam memakai aplikasi di ponsel mereka. Menerapkan arsitektur yang tepat untuk mengimplementasikan *image captioning* ke dalam aplikasi *mobile* dapat mengurangi masalah yang ada dalam pengembangan *image captioning* untuk ponsel pintar.

Pada penelitian ini saya akan mengembangkan sebuah aplikasi *image captioning* yang dapat digunakan di ponsel. Pengembangan dimulai dengan memilih apakah ingin mengembangkan model sendiri atau memakai model yang sudah ada seperti Tensorflow *image captioning* atau keras *image captioning.* Saya memilih menggunakan model dari keras yang menggunakan CNN, *transformer,* dan Flickr8K sebagai datasetnya. Gambar dari dataset dipisah menjadi dua untuk latihan dan validasi, dan setiap gambar memiliki 5 caption yang berbeda yang akan digunakan oleh model sebagai referensi. Gambar kemudian dikirim ke CNN untuk diambil fitur-fiturnya kemudian akan di lanjutkan ke *transformer*, di transformer ini ada 2 bagian yaitu, *transformer encoder* yang akan mengambil fitur gambar dari CNN dan mengerluarkan input baru yang nanti akan diambil oleh *transformer decoder* yang mengambil input dari *transformer encoder* dan *text data* yang didapatkan dari Flickr8K sebagai input, dan belajar dari kedua *input* itu untuk mendapat *output* yang berupa *caption*. Hasil dari latihan model akan disimpan dan digunakan lagi nanti saat pengguna mengirim gambar, sehingga model tidak perlu melakukan latihan lagi. Model akan di unggah ke server yang nanti akan melakukan dengan komunikasi dengan aplikasi android. Aplikasi android ini akan berperan sebagai penghubung antara pengguna dan model, pengguna hanya perlu memiliki koneksi internet dan memilih gambar dari ponsel mereka dan kalkulasi akan dilakukan di server, dengan ini komputasi yang dibutuhkan di android tidak besar, karena aplikasi android hanya perlu mengirim data dan menerima data kembali, dengan metode ini aplikasi android memiliki ukuran yang ringan dan gampang di jalankan, hasil akhir aplikasi dapat dilihat pada Gambar *11* Hasil akhir aplikasi

II. KAJIAN PUSTAKA

A. Image Captioning

Image captioning merupakan proses menghasilkan suatu teks deskripsi dari sebuah gambar. Menurut [4], menggunakan komputer untuk memproduksi suatu teks deskripsi yang sesuai dengan gambar yang diberikan merupakan suatu hal yang menantang, karena menggabungkan riset dari *computer vision* dan *natural languange processing*, *image captioning* tidak hanya memerlukan pemahaman yang tinggi tentang konsep bahasa dari suatu gambar, tetapi juga perlu memberikan hasil teks yang seperti ditulis manusia. [5]

B. Deep Learning

Deep learning adalah salah satu ilmu dalam *machine learning* yang fungsinya melatih komputer untuk berpikir seperti manusia. Perbedaan *deep learning* dan *machine learning* adalah *deep learning* merupakan salah satu keilmuan khusus di bidang *machine learning.* Di dalam *deep learning* terdapat *neural network* yang fungsinya meniru perilaku dari otak manusia, membuat aplikasi komputer untuk mengenali pola dan memecahkan masalah di bidang AI, *machine learning* dan *deep learning*. Salah satu pengaplikasian *deep learning* di dunia nyata adalah *self-driving car*, mampu mendeteksi rambu lalu lintas seperti tanda stop, lampu merah, dan marka jalan. Deep learning bekerja dengan menggunakan *neural network,* model *deep learning* akan dilatih dengan menggunakan data yang besar dan *neural network* belajar langsung dari data tersebut. Salah satu arstiktur yang terkenal di dalam *deep learning* adalah *convolutional neural network* (CNN). CNN adalah salah satu arsitektur di *deep learning* yang belajar langsung dari data, CNN biasanya digunakan untuk mencari pola di dalam gambar, seperti wajah, dan objek, salah satu aplikasi dari CNN adalah *face recognition*.

C. Artificial Intellegence di ponsel pintar

Perkembangan *smartphone* setiap tahunnya semakin baik, dengan adanya *smartphone* yang bertambah kencang, kuat, dan semakin canggih dalam hampir segala aspek. Seiring dengan perkembangan *smartphone*, AI juga berkembang secara pesat dalam konteks komputasi yang dapat membuat ponsel bekerja secara pintar. Beberapa aplikasi yang menggunakan AI sebagai dasarnya adalah *voice assistant*, *personal recommendation*, *facial recognition*, fotografi, dan *object recognition*. [6] memprediksi bahwa dalam tahun 2022, 80% *smartphone* akan diluncurkan dengan AI yang sudah tertanam di dalamnya.

Salah satu tantangan yang ada di dalam topik riset ini adalah kemampuan untuk memproses data dan meningkatkan perilaku dari aplikasi dengan memberikan informasi sekitar seperti *temporal context, spatial context, social context, environtmental* atau *device-related context*. Kemampuan dari aplikasi AI yang ada di *smartphone* bergantung pada data, dan tugas AI yang mempunyai peran penting untuk membuat model yang efektif [7]. Dalam konteks ini, *image captioning* termasuk salah satu AI. Dengan menerapkan *image captioning* ke dalam ponsel, teknologi ini dapat digunakan secara umum, tidak hanya perusahaan besar dan para profesional yang dapat menggunakan teknologi *image captioning*. Salah satu penggunaan *image captioning* yang dapat dimanfaatkan oleh masyarakat umum adalah untuk membantu orang yang memiliki pendengaran, dan pengelihatan kurang, untuk membantu mengidentifikasi lingkungan sekitar dengan menggunakan fitur seperti *text-to-speech*, dan edukasi anak kecil, seperti latihan membaca dari *caption* tentang benda atau aktifitas dalam lingkungan sekitar mereka.

D. Penelitan Sebelumnya

Pada tahun 2019, [8] melakukan penelitan tentang mengembangkan sebuah aplikasi *image captioning* di ponsel pintar untuk membantu orang yang memiliki pengelihatan dan pendengaran kurang. Penelitan mereka menggunakan arsitektur VGG16 dan LSTM sebagai model untuk menghasilkan *caption,* [8] mengatakan bahwa dengan menggunakan arsitektur itu mereka mendapatkan peforma yang lebih baik dalam menghasilkan *caption*. Dataset yang mereka gunakan adalah MSCOCO yang berisi seratus enam puluh ribu gambar, dan setiap gambar memiliki lima caption sebagai referensi. Mereka mengembangkan aplikasi android yang Bernama "*Eye of Horus"*. Parameter yang mereka gunakan adalah 55 *Epoch* dan 1024 untuk *Batch Size. Epoch* berarti satu iterasi dari semua *training set* yang dilakukan untuk memproses gambar dan *caption*, sedangkan *Batch Size* adalah jumlah banyaknya latihan dalam satu kali iterasi.

Aplikasi mereka bekerja dengan cara pengguna mengambil gambar dari ponsel mereka dan aplikasi akan melakukan komunikasi dengan server yang memproses gambar dan mengirim hasil yang berupa *caption* kembali kepada pengguna. [8] mengatakan bahwa sistem yang mereka ajukan memiliki potensial untuk membantu orang yang memiliki pengelihatan dan pendengaran yang kurang.

Adapun penelitian lain yang membahas tentang bagaimana cara mengimplementasikan *artificial intelligence* ke dalam ponsel pintar. Dalam penelitiannya, [3] mengatakan bahwa mengimplementasikan AI kedalam ponsel pintar memiliki banyak rintangan seperti kinerja komputasi, memori yang rendah, dan privasi. Mereka membahas lima arsitektur yang sering dipakai untuk mengimplementasikan AI kedalam ponsel.

Yang pertama adalah *Cloud inference without training*. Arsiterktur ini menerapkan sistem *cloud* yang berarti model dari AI tersebut berada di *cloud* dan aplikasi di ponsel menggunakan API untuk melakukan komunikasi, jadinya aplikasi di ponsel tidak perlu melakukan latihan. [3] mengatakan bahwa arsitektur ini merupakan solusi yang paling cepat dan paling mudah untuk menerapkan AI kedalam aplikasi di ponsel. Solusi ini hanya bisa bekerja jika menggunakan data yang umum, karena pengembang aplikasi ponsel tidak bertanggung jawab untuk melatih model dan mengupdatenya.

Kedua adalah *Both inference and training in the cloud.* Arsitektur ini menerapkan model yang melatih model langsung di server, arsitektur ini hampir sama dengan yang pertamanya sedangkan yang ini pengembang aplikasi ponsel dapat langsung berinterkasi dengan model dari AI. Arsiterktur ini memiliki risiko privasi.

Ketiga adalah *On-device inference with pre-trained model*. Arsitektur ini berguna untuk aplikasi yang memerlukan respon yang instan, seperti *self driving car*

yang memerlukan data secara *real time*. Model akan yang sudah dilatih akan di taruh di perangkat pengguna dan melakukan kalkulasi di perangkat mereka sendiri.

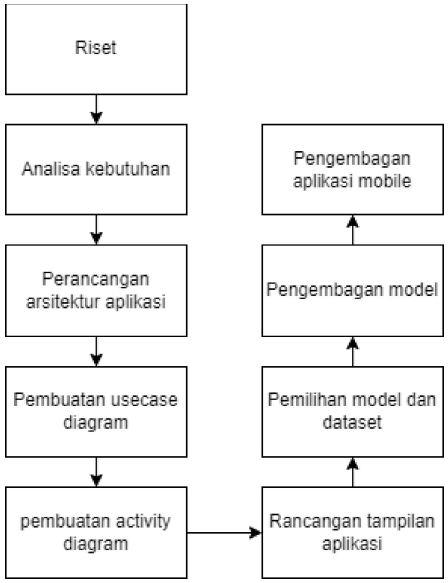
Keempat adalah *Both inference and training on device*. Arsitektur ini melakukan latihan dan kalkulasi di perangkat pengguna, dengan ini model dapat terus terupdate dengan data yang disediakan oleh user, arsitektur ini hanya dapat bekerja dengan AI yang dasar karena keterbatasan kinerja dan memori dalam ponsel.

Yang terakhir adalah *Hybrid.* Model akan dilatih di perangkat pengguna dan di server. Dengan itu, pengembang dapat menggunakan data dari pengguna untuk menyesuaikan model. Arsitektur ini kompleks dan mahal untuk di pelihara.

Dalam kesimpulannya [3] mengatakan bahwa *Cloud base Architecture* merupakan cara implementasi yang paling popular, tetapi dengan perkembangan di dalam dunia teknologi, *machine learning* langsung di dalam perangkat genggam dapat menjadi standart. Selebihnya, dalam masa sekarang melatih dan melakukan komputasi AI di dalam perangkat genggam masih kurang mampu.

III. METODE

Tahap ini akan berisi tahap-tahap atau alur yang digunakan dari awal sampai akhir dalam proses penelitian, dimulai dari melakukan riset, menentukan kebutuhan, perancangan dll. Seluruh alur pengerjaan dapat dilihat pada Gambar *1* Alur metode penelitian.



Gambar 1 Alur metode penelitian

1. Riset

Mengumpulkan data dari riset, buku, dan jurnal tentang aplikasi seputar *image captioning*. Tahap pertama pada penelitian ini adalah melakukan riset, seperti membaca jurnal, penelitan, membaca *conference paper* tentang hal-hal yang berkaitan dengan *image captioning, deep learning* dan implementasi AI kedalam ponsel. Dengan melakukan riset kita dapat mendapat *insight* atau pengetahuan baru dengan tujuan dari penelitian ini.

Dari riset ini kita dapat mengambil poin-point penting yang dapat digunakan sebagai referensi dan masukan terhadap penelitian yang akan dikerjakan. Riset juga dapat membantu kita untuk mencari tahu apakah ada metode atau solusi yang sudah digunakan sebelumnya dengan kasus yang sama dengan penelitian yang akan dikerjakan, dengan adanya informasi tersebut kita dapat memikirkan hal apa saja yang menjadi solusi dan masalah.

2. Analisa Kebutuhan

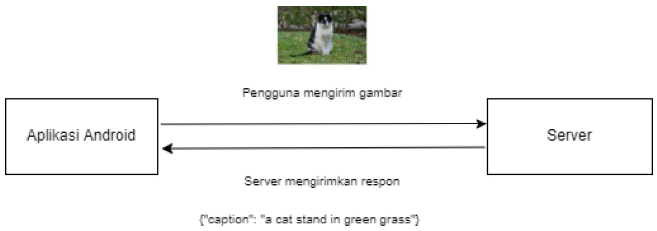
Berdasarkan data yang kita dapat dari riset, kita dapat melakukan Analisa tentang apa saja yang akan dibutuhkan untuk mencapai solusi dari penelitian yang dikerjakan. Analisa kebutuhan dapat berupa apa saja yang diperlukan untuk membangun sebuah aplikasi, Bahasa pemograman yang diperlukan, arsitektur yang digunakan, alur proses aplikasi, model dan dataset yang digunakan, dan metode yang digunakan.

3. Perancangan Arsitektur Aplikasi

Tahap ini keseluruhan arsitektur aplikasi akan dirancang, tahap ini dilakukan di awal karena berdasarkan data yang didapat dari riset, mengimplementasikan AI kedalam aplikasi yang dapat di gunakan di ponsel bukanlah sebuah hal yang mudah.

Arsitektur yang akan dipakai dalam aplikasi ini adalah *Cloud Inference Without Training* yang berarti semua kalkulasi dilakukan di server, dengan ini aplikasi di ponsel tidak memiliki beban yang berat, karena tugas dari aplikasi di ponsel hanyalah mengirim dan menerima data, dan menampilkan data kembali kepada pengguna. Arsitektur ini juga arsitektur yang paling popular digunakan dalam mengimplementasikan AI ke dalam perangkat genggam[3].

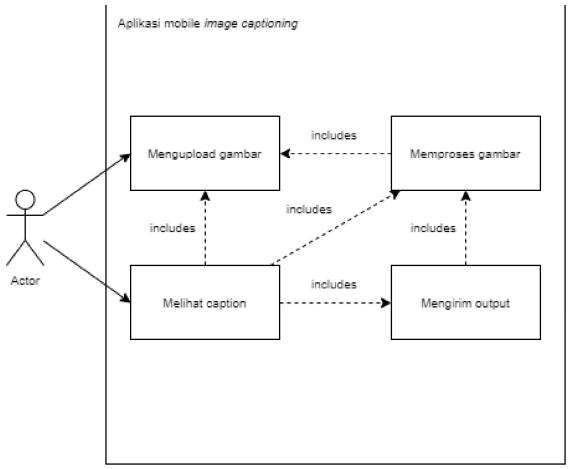
Cara kerja dari arsitektur ini adalah model yang sudah dilatih akan diunggah ke server dan nantinya akan melakukan komunikasi dengan menggunakan API. Pengguna akan memilih gambar dari ponsel mereka dan server akan mengolah data tersebut dan mengirim hasil yang berupa *caption* untuk ditampilkan kepada user, bisa merujuk Gambar *2* Arsitektur aplikasi



Gambar 2 Arsitektur aplikasi

4. Pembuatan Usecase Diagram

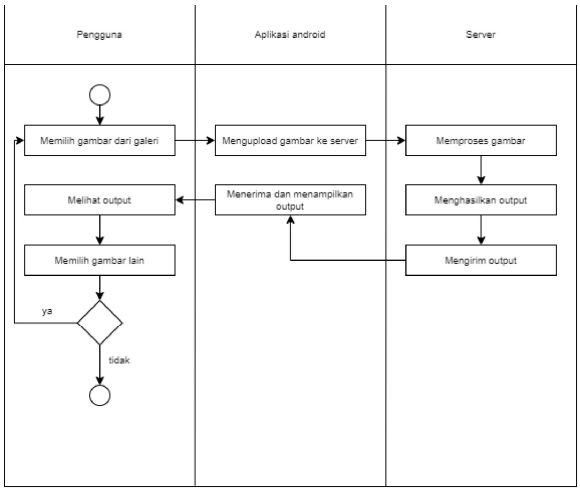
Pembuatan usecase diagram bertujuan untuk menetukan bagaimana actor akan berhubungan dengan aplikasi. Dengan membuat usecase diagram, kita akan tahu interaksi seperti apa yang akan terjadi antara actor dan aplikasi. Dalam aplikasi ini aktor hanya berjumlah satu, yaitu si pengguna itu sendiri. Usecase diagram dapat dilihat pada Gambar *3* Usecase diagram aplikasi.



Gambar 3 Usecase diagram aplikasi

5. Pembuatan Activity Diagram

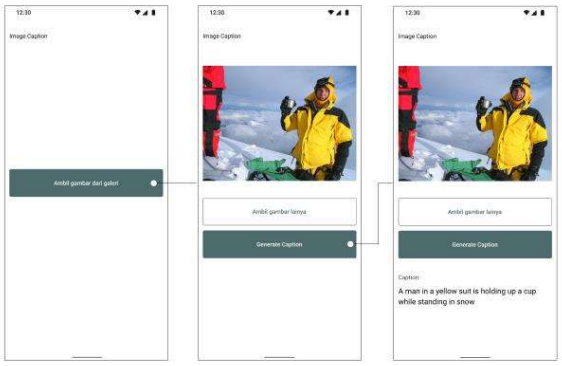
Tahap ini akan dibuat activity diagram, yang bertujuan untuk melihat workflow atau tahapan tahapan yang terjadi di dalam aplikasi. Diagram ini akan menampilkan proses yang terjadi. Dengan adanya diagram ini kita melihat proses yang terjadi saat pengguna menggunakan aplikasi ini. Untuk diagram dapat dilihat pada Gambar *4* Acitvity diagram.



Gambar 4 Acitvity diagram

6. Rancangan Tampilan Aplikasi

Tahap ini akan membuat rancangan tampilan aplikasi yang nanti pada saat aplikasi dibuat, tampilan aplikasi akan merujuk kesini. Tampilan aplikasi dapat dilihat pada Gambar *5* Rancangan tampilan aplikasi.



Gambar 5 Rancangan tampilan aplikasi

7 Pemilihan Model dan Dataset

Setelah semua rancangan selesai dibuat, langkah selanjutnya adalah memilih atau mengembangkan model *image captioning*. Untuk ini saya memilih menggunakan model yang sudah dikembangkan oleh keras. Model ini menggunakan CNN dan *transformer* sebagai arsitekturnya. Untuk dataset, akan menggunakan Flickr_8K yang berisi 8000 gambar, dan setiap gambar memiliki 5 pasang *caption* sebagai referensi untuk melatih model. Contoh gambar yang diambil dari dataset bisa dilihat pada.

Gambar 6 Contoh gambar dari Flickr_8K dataset

Setiap gambar memiliki lima pasang caption yang digunakan sebagai referensi, pada Gambar *6* Contoh gambar dari Flickr_8K dataset, terdapat referensi seperti yang ditunjukkan pada Tabel *1* Refensi caption.

1	*A black dog and a spotted dog are fighting*
2	*A black dog nad tri-colored dog playing wqith each other on the road*
3	*A black dog and a white dog with brown spots are staring at each other in the street*
4	*Two dogs of different breeds looking at each other on the road*
5	*Two dogs on pavement movid toward each other*

Tabel 1 Refensi caption

7. Pengembangan Model

Setelah pemilihan model dan dataset selesai, langkah selanjutnya adalah mengembangakan model. Keras menggunakan arsitektur CNN dan *transformer* sebagai modelnya. Model yang digunakan pada penelitian ini berbeda dari penelitian yang dilakukan oleh [8], yang dimana mereka menggunakan VGG16 dan LSTM sebagai arsitekturnya, sedangkan dalam penelitian ini menggunakan EfficientNetB0 dan Transformer sebagai arsitekturnya. Dataset yang digunakan pun berbeda, dalam penelitian ini menggunakan Flickr_8K, sedangkan [8] menggunakan MSCOCO.

Tahap pertama dalam pengembangan model adalah menyiapkan dataset, bisa dilihat pada Gambar *6* Contoh gambar dari Flickr_8K dataset dan Tabel *1* Refensi caption, sebagai contoh gambar dan referensi captionnya. Gambar dari dataset ini akan di konfigurasi seperti dimensi gambar yang diinginkan, untuk ini akan di set menjadi (299x299). Selajutnya adalah parameter *Epoch* dan *Batch Size*. *Epoch* disini sebanyak 30 dan *Batch Size* sebesar 64.

Setelah parameter gambar selesai di konfigurasi, gambar di dalam dataset akan dibagi menjadi 2 yaitu *train sets*, yang digunakan untuk melakukan latihan, dan *validation sets*, yang digunakan sebagai tes validasi, dengan rasio 80-20, yang berarti gambar yang dilakukan untuk latihan adalah sebanyak 80% dari total gambar dari dataset dan 20% sisanya digunakan untuk melakukan validasi.

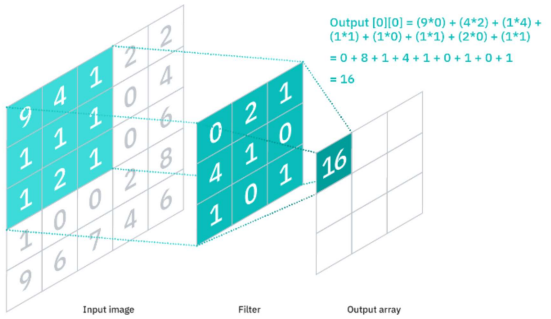
Langkah selanjutnya adalah membangun modelnya, pada bagian ini keras menggunakan arsitektur yang memiliki 3 bagian, yaitu CNN, *TransformerEncoder,* dan *TransformerDecoder*.

CNN adalah satu tipe dari *neural networks*. *Neural network* sendiri merupakan kumpulan algoritma yang meniru perilaku otak manusia, memungkinkan computer untuk mengenali pola, dan menyelesaikan masalah di bidang AI, *machine learning* dan *deep learning. Neural Network* berisi dari *node layers*, yang berisi *input layer*, satu atau lebih *hidden layer*, dan *output layer*. Setiap node berhubungan dengan *node* yang lain. *Neural Network* sangat bergantung pada data latihan untuk belajar dan berkembang. CNN berbeda dengan tipe *neural network* lainya karena memiliki peforma lebih baik pada bagian gambar, *speech,* dan *signal audio inputs.*

CNN terdiri dari 3 layer, yaitu: (1) *Convolutional Layer*, (2) *Pooling Layer,* (3) *Fully-connected layer*

Convolutional Layer adalah bagian utama dari CNN, dalam bagian ini memerlukan beberapa komponen seperti, *input filter,* dan *feature map*. Sebagai contoh, ibaratkan *input* berupa gambar, yang memiliki *matrix* 3 dimensi yang berisi panjang x tinggi x channel yang menyesuaikan gambar.

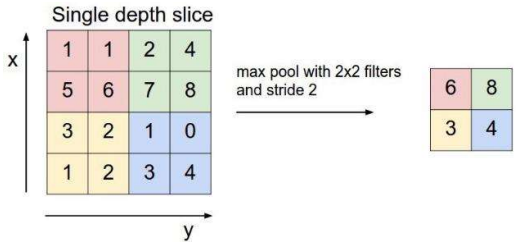
Pertama, di dalam convlayer, terdapat matrix yang membentuk sebuah filter dengan panjang, tinggi. dalam *pixels. Filter* itu akan digunakan untuk menganalisa seluruh bagian gambar, dan akan menghasilkan *output* yang biasa disebut *feature map*, ilustrasi dari kerja convlayer bisa dilihat pada Gambar 7 Ilustrasi kerja pada *convoluional layer*



Gambar 7 Ilustrasi kerja pada *convoluional layer*

Terdapat beberapa *hyperparameter* yang bisa di konfigurasi pada layer ini, seperti berapa banyak filter yang digunakan, *stride* yang fungsinya menentukan jumlah pergeseran filter, dengan menggunakan nilai *stride* yang kecil, informasi yang didapatkan lebih banyak, namun memiliki dampak pada peforma komputasi, dan *padding* yang berguna untuk menambah pixel yang bernilai 0 di sisi input gambar, dengan tujuan memanipulasi dimensi dari *output* atau *feature map*.

Layer selanjutnya adalah *Pooling layer*, tujuan dari layer ini adalah melakukan filter pada *feature map* atau *output* dari convlayer, dengan berkurangnya dimensi dari *feature map*, akan mempercepat komputasi. Terdapat 2 tipe pada layer ini yaitu, *Max Pooling* dan *Average Pooling*. *Max pooling* akan melakukan filter dari *input* yang didapat, dan dari matrix tersebut akan diambil nilai yang paling besar, contoh dalam *input layer* berisi matrix 4x4, dan menggunakan max pooling 2x2 dengan stride 2, pada bagian matrix 2x2 pertama, berisi nilai 1,1,5,6 maka nilai yang diambil adalah 6 karena dari matrix 2x2 tersebut nilai 6 adalah yang paling besar. Agar lebih jelas bisa lihat pada Gambar *8* Ilustrasi max pooling.



Gambar 8 Ilustrasi max pooling

Sedangkan *average pooling* akan mengambil nilai rata-rata dari matrix.

Layer terakhir pada convlayer adalah *Fully-Connected Layer* (FC Layer). *Layer* ini melakukan tugas untuk mengklasifikasi dari *feature map*. *Feature map* akan di ubah bentuk nya agar bisa digunakan sebagai *input* untuk FC Layer, biasanya input dari FC Layer bernilai dari 0 sampai 1.

Transformer model adalah sebuah *neural network* yang belajar dari konteks, dan hubungan dari data. Model ini menggunakan teknik matematik yang bernama *attention* atau *self-attention*, yang berguna untuk mendeteksi bagaimana suatu elemen di data mempengaruhi satu sama lain. Sebagai contoh "*A big cat is drinking water from a bowl until it was empty"* dalam kalimat itu, model *transformer* dapat menghubungkan konteks dan relasi, seperti "*big"* dengan

"*cat*", "*it*" dengan "*water*". *Attention layer* akan menghasilan kata individual dan relasinya kepada kata lain. Input dari transformer biasanya berupa teks, yang setiap kata dari teks akan di pisah dan diberikan ID. Selanjutnya input akan ditujukkan ke *embedding* layer untuk merubah input menjadi vector. Selanjutnya akan menuju layer *multi-head attention*, layer ini akan menghasilkan beberapa relasi dari kata individual. Kemudian menuju ke *decoder* layer yang akan menerjemahkan input yang didapat dari *attention*. Dalam masa latihan, *decoder* akan memiliki vektor dari layer *encoder* dan hasil yang di harapkan. *Encoder* akan berusaha memprediksi kata yang akan digunakan selanjutnya dan memperbaiki kesalahan berdasarkan perbedaan dari hasil dan hasil yang diharapkan.

Untuk mendapat hasil yang maksimal, diperlukan data latihan dan parameter yang banyak. Seluruh arsitekur nya dapat dilihat pada Gambar 9 Arsitektur model transformer.

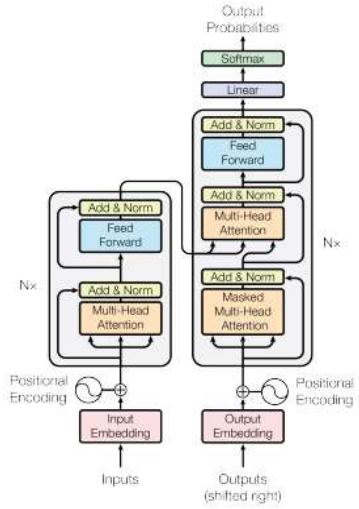


Figure 1: The Transformer - model architecture.

Gambar 9 Arsitektur model *transformer*

Model *image captioning* yang digunakan pada penelitian ini menggunakan arsitektur yang bernama EfficientNetB0. EffiecientNet adalah arsiterkur yang bekerja dengan menggunakan metode *scaling* kedalaman, lebar dan resolusi dari CNN secara konstan. Jika input gambar yang diberikan berukuran besar, maka layer didalamnya pun akan bertambah untuk menyesuaikan dengan input. EfficientNet mendapat akurasi dan efesiensi lebih baik dari pada ConvNets, dengan EfficientNetB7 mendapatkan akurasi sebesar 84.3% dalam ImageNet [9].

Pada model yang dikembangkan, CNN atau *Convolutional Neural Network* digunakan untuk mengambil fitur-fitur dan mengenali objek yang ada di gambar. *TransformerEncoder* akan mengambil hasil fitur dan objek yang telah di proses oleh CNN dan menjadikan fitur dan objek sebagai input baru. Selanjutnya *TransformerDecoder* akan mengambil input dari *TransformerEncoder* dan mengambil 5 caption yang disediakan dari dataset, kedua input tersebut akan menjadi bahan *TransformerDecoder* untuk belajar dan menghasilkan caption. Konfigurasi dari model ini menggunakan input dengan ukuran 299x299x3, tidak menggunakan FC Layer dan Pooling layer, yang berarti akan menggunakan *output* langsung yang dihasilkan pada convlayer.

Model ini akan ditest dan diberi nilai yang dinamakan dengan nama Bleu score. Bleu score adalah suatu matrik untuk mengukur seberapa bagus mesin dalam melakukan translasi. Blue score dinilai dengan nilai dari 0 sampai dengan 1, dengan nilai 1 yang berarti sempurna. Di dalam Bleu score juga terdapat yang namanya n-gram, n-gram berfungsi untuk mencocokan referensi dan hasil, n-gram biasanya terdiri dari 4 individual n-gram, 1-gram berarti mencocokan 1 kata dari *caption* ke referensi, 2-gram mencocokan 2 pasang kata dari *caption* ke referensi dan seterusnya. Pada test ini saya akan menggunakan 15 gambar dari test set atau validation set untuk menguji model ini.

Sebagai contoh, Gambar *10* Gambar uji coba test Bleu akan saya gunakan sebagai mengetes bleu test.

Gambar 10 Gambar uji coba test Bleu

Setiap gambar memiliki 5 referensi *caption*, sebagai contoh dapat dilihat pada Tabel *1* Refensi caption. Pada Gambar *10* Gambar uji coba test Bleu, hasilnya mendapatkan score bleu-4 sebesar 0.4366835442847812. Dari semua gambar yang ditest, model ini mendapatkan yang bisa dilihat di Tabel 2 Hasil test bleu.

Tipe test	Rata-rata
Individual 1-gram	0.69654173333333
Individual 2-gram	0.55493713333333
Individual 3-gram	0.4537012
Individual 4-gram	0.31178571428571

Tabel 2 Hasil test bleu

Dari 15 gambar, model ini mendapatkan blue-4 skor sebesar 0.31, yang berarti dapat memahami esensi dari gambar, tetapi memiliki tatabahasa yang masih kurang baik.

Setelah model selesai belajar atau latihan, hasil latihan model ini akan disimpan, dan nantinya akan digunakan sebagai model dasar untuk menghasilkan *output* saat digabungkan dsengan aplikasi android. Untuk pelatihan model nya bisa dilakukan berulang kali, namun pada penelitian ini cukup dilakukan 1 kali iterasi saja, sesuai dengan EPOCH dan *batch size* yang sudah dikonfigurasi di awal. Model bisa di latih berulang kali menggunakan konfigurasi yang sama atau mengganti konfigurasi sesuai \ dengan yang diinginkan.

Ketika model sudah selesai latihan dan model sudah disimpan, hal yang selanjutnya akan dilakukan adalah mengembangkan API agar bisa melakukan komunikasi dengan perangkat lain melalui internet. Untuk

mengembangkan API, digunakan framework bernama Flask. Setelah pengembangan model dan API selesai selanjutnya adalah mengunggah nya ke server, server yang dipilih adalah Heroku.

Untuk melakukan *update* kepada model, bisa dilakukan dengan mengkonfigurasi parameter dari model, melakukan training ulang, sampai mendapatkan hasil yang diinginkan. Setelah itu hasil dari latihan dapat disimpan, dan cukup di *deploy* ulang ke server, dan model dari *image captioning* sudah terupdate.

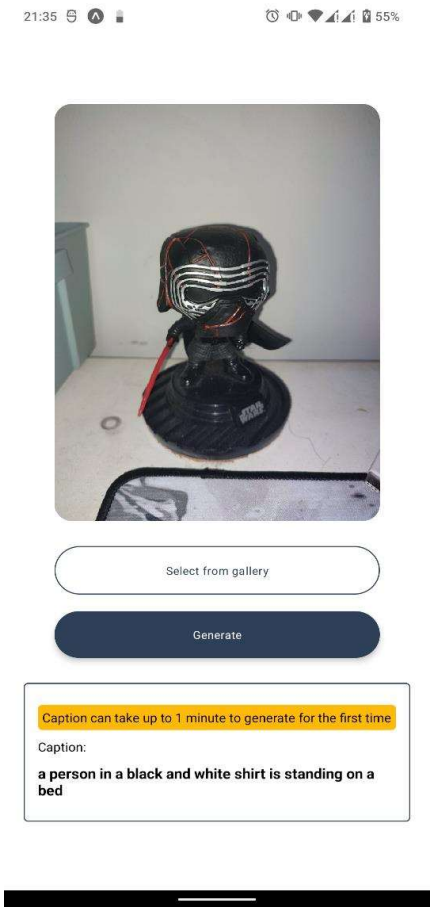
8. Pengembangan Aplikasi Android

Tahap terakhir adalah melakukan pengembangan aplikasi android dan tesnya. Untuk tahap ini saya akan menggunakan bahasa pemograman *javascript* dan menggunakan framework bernama Expo.

Tahap pertama adalah mengembangkan fitur memilih gambar, fitur ini bertujuan agar pengguna bisa memilih gambar dari galeri mereka, gambar ini nanti akan dikirim ke server untuk mendapat *caption.*

Setelah user memilih gambar, gambar akan dikirim ke server untuk di proses, setelah selesai, server akan mengirimkan data yang berupa JSON, di dalam JSON berisi caption yang nantinya akan ditampilkan kepada pengguna.

Hasil akhir bisa dilihat pada Gambar *11* Hasil akhir aplikasi



Gambar 11 Hasil akhir aplikasi

Langkah selanjutnya adalah membuat test, untuk menguji apakah aplikasi berjalan seperti seharusnya. Untuk tes akan dilakukan 2 tes yaitu Black Box test dan UAT.

Black box test akan menguji aplikasi tanpa melihat *source code* dan UAT akan mengetes aplikasi dengan pengguna langsung. Pengujian fungsional akan menguji fitur yang ada di aplikasi. Untuk pengujian black box dapat dilihat pada Tabel *3* Tabel pengujian black box

Fungsi	skenario	Hasil yang diharapkan	Berhasil	Gagal
Upload	User menguploa d gambar	Menghasilka n caption	✓	
Upload	User menguploa d selain gambar	Menghasilka n error	✓	

Tabel 3 Tabel pengujian black box

UAT test akan dilakukan dengan pengguna langsung menggunakan aplikasi dan mengisi kuisioner tentang pengalaman mereka saat menggunakan aplikasi. Aplikasi yang sudah dites, dapat di gunakan sebagaimana aplikasi pada umumnya, karena fitur yang ada pada penelitian ini hanya mencakup mengambil gambar dari galeri bisa dibilang aplikasi ini masih pada tahap alfa, dan fitur lainya seperti mengambil dari kamera, *text-to-speech* bisa di tambahkan.

IV. KESIMPULAN

Pada penelitian ini, telah ditunjukkan cara pengembangan aplikasi *image captioning* untuk ponsel pintar, aplikasi ini memanfaatkan model *image captioning* dari keras yang menggunakan arsitektur EfficientNetB0 dan *Transformer*, dan arsitektur *cloud based* untuk aplikasi androidnya, yang memungkinkan untuk mengimplementasikan *image captioning* kedalam ponsel pintar tanpa khawatir dengan kendala peforma dari ponsel, hal perlu diperhatikan adalah aplikasi ini memerlukan koneksi internet untuk dapat menghasilkan caption. Dengan menggunakan tampilan yang gampang digunakan dan pengoperasian yang gampang, cukup memilih gambar menekan tombol *generate* dan *caption* akan muncul dalam beberapa detik saja. Aplikasi ini masih dalam tahap alfa yang bisa di kembangkan lagi dengan menambah fitur-fitur baru, dan melatih model lagi dengan parameter yang berbeda untuk mendapat hasil yang maksimal. Kendala yang ada pada aplikasi ini adalah, untuk *startup* pertama, server bisa memakan waktu kurang lebih sampai 1 menit dan terkadang heroku akan mengirimkan error karena memakan waktu terlalu lama untuk menghidupkan server, karena *default* nya heroku memasang waktu maksimal 30 detik, jika lebih dari itu maka heroku akan mengirimkan error, karena menggunakan jasa gratis dari heroku, server akan dimatikan selama 6 jam dalam sehari, dengan adanya limitasi ini, sangat dianjurkan untuk menggunakan jasa server yang berbayar dan aplikasi tidak akan bisa menghasilkan *caption* jika server tidak berjalan. Selain itu tidak banyak aplikasi *image captioning* dalam ponsel yang bisa dijadikan referensi.

REFERENCES

[1] Md. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, “A Comprehensive Survey of Deep Learning for Image Captioning,” vol. 51, no. 6, pp. 1–36, Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.04020

[2] L. Srinivasan and D. Sreekanthan, “Image Captioning-A Deep Learning Approach,” 2018. [Online]. Available: http://www.ripublication.com

[3] X. Dai, I. Spasic, S. Chapman, and B. Meyer, “The State of the Art in Implementing Machine Learning for Mobile Apps: A Survey,” in *Conference Proceedings - IEEE SOUTHEASTCON*, Mar. 2020, vol. 2020-March. doi: 10.1109/SoutheastCon44009.2020.9249652.

[4] S. Bai and S. An, “A survey on automatic image caption generation,” *Neurocomputing*, vol. 311, pp. 291–304, Oct. 2018, doi: 10.1016/j.neucom.2018.05.080.

[5] S. Herdade, A. Kappeler, K. Boakye, and J. Soares, “Image Captioning: Transforming Objects into Words,” 2019.

[6] Gartner, “Gartner Highlights 10 Uses for AI-Powered Smartphones,” Jan. 04, 2018.

[7] I. H. Sarker, M. M. Hoque, M. K. Uddin, and T. Alsanoosy, “Mobile Data Science and Intelligent Apps: Concepts, AI-Based Modeling and Research Directions,” *Mobile Networks and Applications*, vol. 26, no. 1, pp. 285–303, Feb. 2021, doi: 10.1007/s11036-020-01650-z.

[8] B. Makav and V. Kılıç, “Smartphone-based Image Captioning for Visually and Hearing Impaired,” 2019.

[9] M. Tan and Q. v Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” 2019.