BAB III
METODOLOGI PENELITIAN

Pada penelitian ini, metodologi yang digunakan berupa pengembangan model machine learning menggunakan *universal sentence encoder* (USE) dan *Tensorflow Recommendation* (TFRS) untuk meningkatkan akurasi pada sistem rekomendasi. Terdapat beberapa tahapan proses yang perlu dilakukan untuk mencapai hasil yang diharapkan, diantaranya sebagai berikut.

Studi literatur yang dilakukan bertujuan untuk mencari informasi mengenai *machine learning*, *deep learning* dengan metode *Convolution Neural Network* menggunakan *Universal Sentence Encoder* (USE) dan *TF-Recommendation*, penerapan hasil rekomendasi *user* kedalam aplikasi Collabolio berbasis *android*. Collabolio akan merekomendasikan *user* yang sejenis dengan apa yang diinginkan pengguna dengan mencocokkan parameter *skill* dan *job interst*. *Universal Sentence Encoder* dan *TF-Recommendation* memiliki kemampuan untuk menghasilkan representasi kalimat yang dapat digunakan secara *transferable* dalam berbagai tugas klasifikasi teks. Dengan menerapkan teknik *transfer learning* terdapat representasi kalimat yang dihasilkan oleh USE dapat meningkatkan kinerja klasifikasi teks, terutama pada dataset dengan sumber daya terbatas. Sehingga, USE memiliki potensi yang besar sebagai alat yang dapat digunakan dalam berbagai aplikasi pemrosesan bahasa alami.

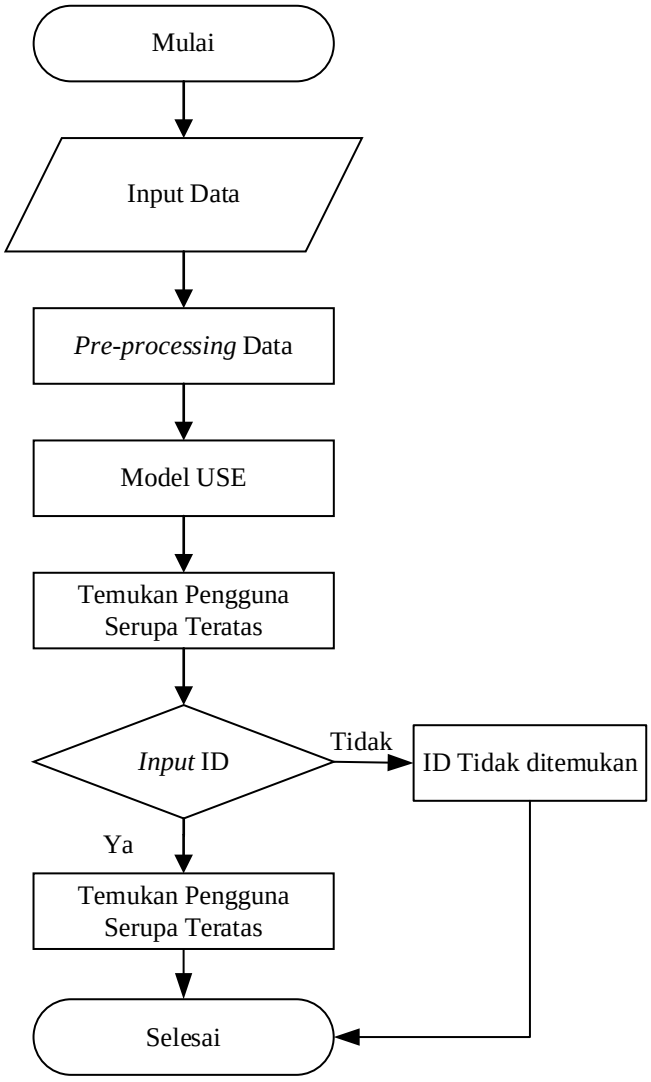
3.1 Perancangan Algoritma

Perancangan algoritma dilakukan untuk membuat prosedur atau langkah-langkah guna menyelesaikan suatu masalah. Pada penelitian ini terdapat dua buah metode yang digunakan yaitu USE dan *TF-Recommendation*.

3.1.1 *Universal Sentence Encoder (USE)*

Perancangan algoritma pada metode USE digunakan untuk mengoptimalisasikan hasil rekomendasi yang diinginkan dalam masalah klasifikasi

teks atau pencocok kalimat. Pada perancangan algoritma ini menggunakan bahasa *python* serta menggunakan *open library* seperti *Tensorflow*, *keras*, dan *pandas*. Sumber data yang digunakan pada perancangan ini berasal dari data yang di-*generate* pada *website* Mockaroo dan terdapat dua buah parameter yang dijadikan acuan saat rekomendasi yaitu *job* dan *interest*. Berikut ini merupakan diagram alir penelitian menggunakan metode USE dapat dilihat pada Gambar 3.1 berikut.



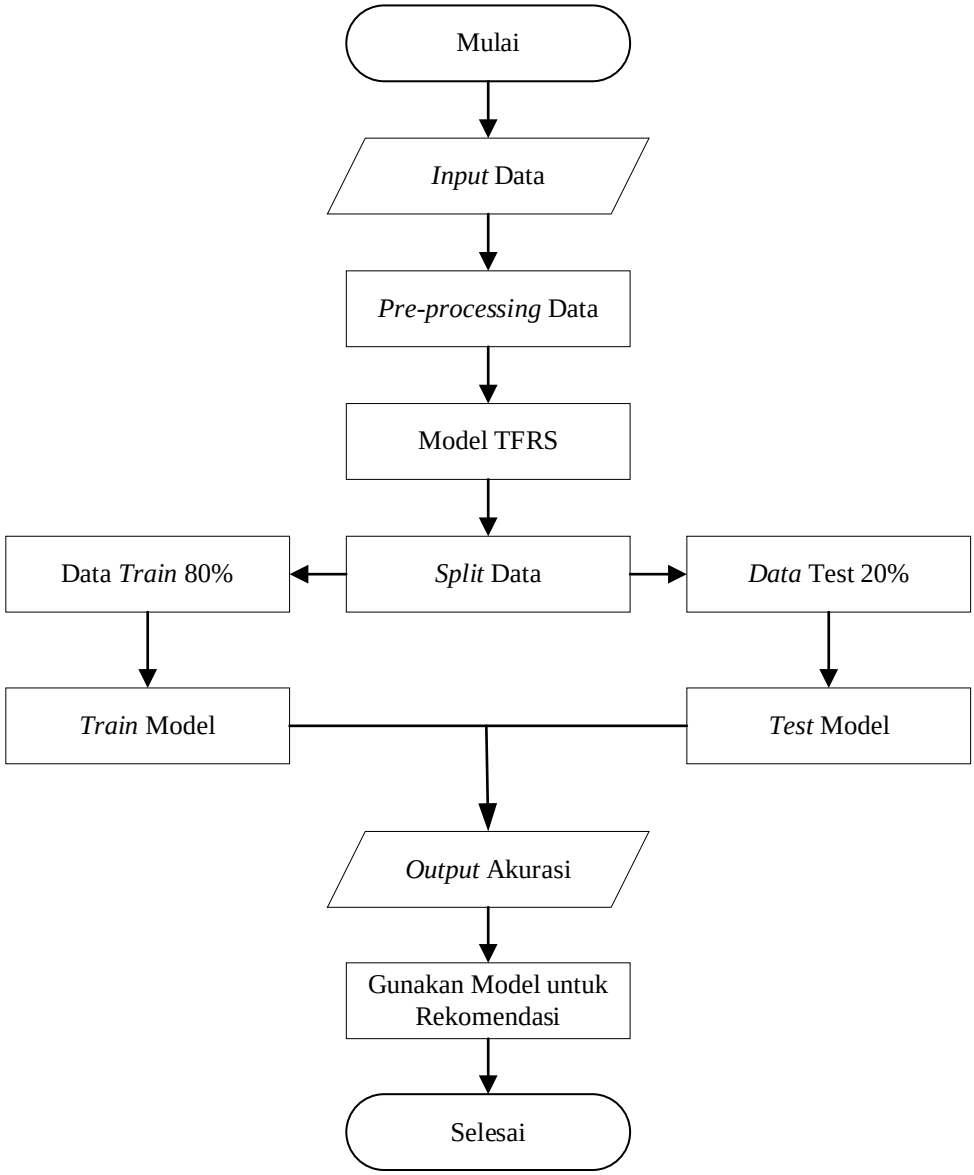
Gambar 3.1 Rancangan Proses Algoritma USE

Pada Gambar 3.1 terdapat skema pelaksanaan algoritma USE yang akan diterapkan dalam penelitian ini. Langkah pertama yaitu melakukan *input* data ke

dalam *Firebase* menggunakan tools *JetAdmin*. Proses pengolahan dataset menggunakan bahasa pemrograman Python, dengan menerapkan teknik *preprocessing* dalam pemrosesan bahasa alami (NLP), seperti tokenisasi, transformasi huruf, dan pengkodean teks melalui *word embedding* menggunakan USE. Selanjutnya, pembuatan model dengan menggunakan *word embbeding* dan transfomator *Deep Average Network* (DAN). Proses selanjutnya ialah menemukan pengguna yang serupa atau *similar* dengan menggunakan *word embedding* yang disebutkan di atas. Setelah pengujian model, algoritma USE akan menghasilkan rekomendasi pengguna lain yang paling serupa dengan pengguna saat ini, berdasarkan nilai *similarity* antara *vektor representasi*. Jika, *user* tidak ditemukan maka masuk ke kondisi yang memunculkan pesan "*user not found*". Kondisi tersebut terjadi saat *user* memasukkan atau mencari *user* ID yang belum terdaftar atau tidak ada pada *database*. Rekomendasi yang dihasilkan mencakup dua parameter, yaitu keterampilan (*skill*) dan minat pekerjaan (*job interest*). *Output* rekomendasi akan disajikan dalam bentuk daftar pengguna lain beserta nilai *similarity*.

3.1.2 *Tensorflow Recommendation (TF-RS)*

Perancangan algoritma TF-RS digunakan untuk memprediksi kebutuhan pengguna berdasarkan data yang telah di-*preprocessing*. Pada perancangan algoritma ini menggunakan bahasa python serta menggunakan *open library* seperti *Tensorflow*, *keras*, dan *pandas*. Sumber data yang digunakan pada perancangan ini berasal dari data yang di ganerate pada *website* Mockaroo dan terdapat dua buah parameter yang dijadikan acuan saat rekomendasi yaitu *job* dan *interest*. Perancangan TF-RS melibatkan beberapa proses utama yang dapat dilihat pada Gamabr 3.2 di bawah ini.



Gambar 3.2 Diagram Alir Proses *Tensorflow Recommendation*

Pada Gambar 3.2 terdapat skema pelaksanaan algoritma TF-RS yang akan diterapkan dalam penelitian ini. Langkah pertama yang dilakukan ialah peng-*inputan* data yang sudah diolah melalui Microsoft Excel. Proses pengolahan *dataset* menggunakan bahasa pemrograman Python, dataset dibagi menjadi dua bagian utama yaitu data pelatihan (*training set*) dan data pengujian (*data test*) sebesar 4:1 atau 80% *data train* dan 20% *data test*. Pembuatan model TF-RS menggunakan

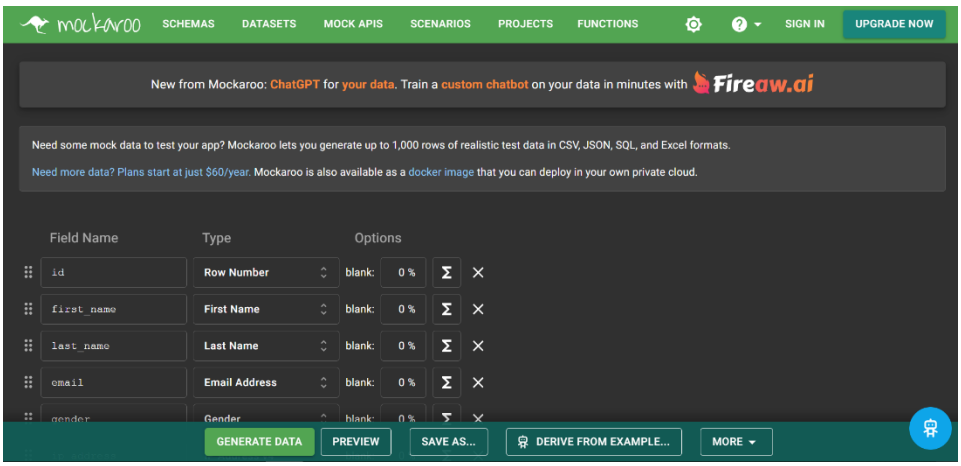
tensorflow ini akan menggunakan *collaborative filtering*. Rekomendasi yang dihasilkan akan berupa ID pengguna menggunakan indeks pencarian *brute-force*.

3.2 Pengambilan Dataset

Pada penelitian yang akan dilakukan menggunakan dua buah metode yang akan dibandingkan yaitu metode USE dan metode TF-RS. Kedua metode tersebut menggunakan dataset dan data test yang sama. Dataset yang digunakan terdiri dari beberapa atribut antara lain:

a. ID *user*
b. *Name*
c. *Email*
d. *Gender*
e. *Username*
f. *Phone*
g. *City*
h. *Avatar*
i. *University*
j. *Skill* 1
k. *Skill* 2
l. *Skill* 3
m. *Skill* 4
n. *Birth*
o. Mbti
p. *Age*
q. *Password*
r. *Job interest*

Dataset di atas dibuat menggunakan *website* Mockaroo. Berikut merupakan tampilan dari website Mockaroo dapat dilihat sebagai berikut.



Gambar 3.3 Tampilan *Website* Mockaroo

Pada Gambar 3.3 merupakan tampilan *website* Mockaroo. Mockaroo ialah alat *daring* gratis yang digunakan untuk menghasilkan kumpulan data CSV, JSON, SQL, dan Excel secara kustom untuk melakukan pengujian. Alat ini menyediakan berbagai jenis data bawaan dan kemampuan untuk mengunggah data referensi kustom atau membuat data secara kustom menggunakan API formula Mockaroo.

Data *test* digunakan untuk menguji kinerja suatu model atau sebuah algoritma. Data *test* yang digunakan pada penelitian ini dibuat menggunakan *website* yang sama seperti pembuatan dataset yaitu, Mockaroo. Namun, dataset yang digunakan hanya berisi dua buah atribut yaitu ID dan *Job Interest*. Sehingga, ID yang terdapat pada dataset dan data *test* menjadi *primary key* pada penelitian ini.

3.3 *Training* Dataset

Pada umumnya *training set* digunakan untuk melatih model *machine learning* atau sebuah algoritma. *Training* set umumnya dipisahkan dari data validasi dan data *test*. Berikut ini merupakan alur dalam proses *training set* untuk melatih model yang sudah dirancang dapat dilihat pada Gambar 3.4.

Gambar 3.4 *Training Process* Model

Pada Gambar 3.4 *input* dataset yang digunakan untuk proses *training* pada penelitian ini didapatkan dari *website* Mockaroo yang sudah di lakukan proses *data cleaning*. *Data cleaning* dilakukan untuk mengidentifikasi dan mempebaiki atau menghapus data yang tidak akurat, rusak, tidak lengkap, dan lain-lain. Hal ini dilakukan bertujuan untuk memastikan bahwa data yang digunakan untuk melakukan analisis atau pengambilan keputusan memiliki kualitas yang bagus dan dapat dipercaya. Data *train* yang digunakan sebesar 80% dan data *test* yang

digunakan sebesar 20% dikarenakan pada penelitian sebelumnya perbandingan 80:20 banyak menghasilkan hasil yang memuaskan. Proses *training* dilakukan menggunakan model yang sudah dibuat menggunakan dua metode yang digunakan pada penelitian yang akan menghasilkan model berupa *recommendation result*.

3.4 *Loss Function*

Loss function merupakan sebuah fungsi kerugian yang dimana terdapat matriks untuk mengukur seberapa baik model *machine learning* (ML) yang digunakan guna meningkatkan keakuratan. Pada sistem rekomendasi *loss function* digunakan untuk mengukur sejauh mana prediksi sistem rekomendasi yang cocok dengan preferensi atau perilaku pengguna [45]. Tujuan utama *loss function* ialah mengoptimalkan model rekomendasi agar memberikan rekomendasi yang paling relevan dan sesuai dengan kebutuhan pengguna.

Ada beberapa jenis *loss function* yang umum digunakan dalam sistem rekomendasi. Berikut jenis *loss function* yang sering digunakan.

1. *Mean Squared Error* (MSE) merupakan jenis *loss fuction* yang digunakan dalam pengembangan model ML. MSE digunakan untuk menghitung perbedaan kuadrat antara *output* yang dihasilkan oleh model dan *output* yang di harapkan. MSE mengukur tingkat kesalahan prediksi model dalam meningkatkan kinerja model terhadap kesalahan prediksi. MSE akan lebih sensitif terhadap sebuah *outlier* karena menggunakan formula selisih kuadrat [45]. Cara menghitung MSE yaitu dengan menghitung perbedaan kuadrat antar *output* yang dihasilkan model dengan *output* yang diharapkan dibagi dengan jumlah *output*.
2. *Cross-Entropy Loss* mengukur sebuah perbedaah antara disribusi probabilitas prediksi dan distribusi probabilitas yang sebenarnya. *Cross-Entropy Loss* menggabungkan kedua kontribusi kesalahan untuk menghasilkan nilai kesalahan keseluruhan. Nilai *loss* yang lebih rendah menunjukkan prediksi model lebih dekat dengan distribusi probabilitas yang benar. Model yang baik memiliki nilai *cross-entropy loss* 0. Terdapat 3 jenis *Cross-Entropy* yaitu *Binary Cross-Entropy* yang menyelesaikan

tugas biner, *Categorical Cross Entropy* digunakan untuk tugas binear dan *multiclass*. *Categorical Cross-Entropy* membutuhkan label untuk dikodekan sebagai sebuah kategori. *Spare Cross-Entropy* mirip dengan *categorical cross-entropy* yang menjadi pembeda ialah *spare cross-entropy* membutuhkan label serupa bilangan bulat.

3. *Rangking Losses* dirancang untuk perangkingan. Tujuan utama dari *rangking losses* adalah mengurutkan *item* berdasarkan preferensi. Terdapat beberapa jenis *rangking losses* seperti *pairwise rangking loss*, *listwise rangking loss*, *mean average precision, normalized discounted cumulative gain,* dan *hinge loss*. Berbagai jenis *rangking losses* dapat dipilih sesuai dengan kebutuhan data dan tujuan dari tugas. *Loss fuction* ini digunakan dalam proses pelatihan model untuk mengoptimalkan peringkat dan kualitas hasil peringkat.

3.5 *Similar Score*

Similar score merupakan salah satu langkah dalam membangun sistem rekomendasi yang menghasilkan skor kesamaan atau kemiripan antar *item* yang terdapat pada dataset. *Similar score* berfungsi sebagai pembanding preferensi pengguna dengan *item* yang ada pada dataset sehingga akan menghasilkan skor kemiripan tertinggi dapat direkomendasikan kepada pengguna. Dengan kata lain, *user* atau pengguna yang memiliki skor kemiripan tertinggi akan mendapatkan peringkat yang lebih tinggi dalam daftar rekomendasi. Hal ini dilakukan guna pengguna menerima hasil rekomendasi yang lebih relevan dan akurat berkat penggunaan hasil rekomendasi menggunakan *similar score*.

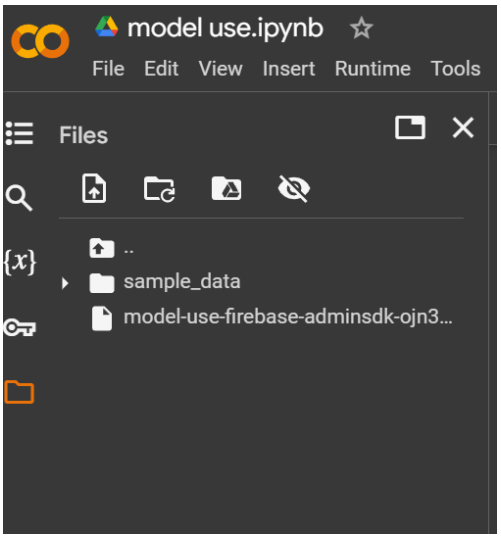
Dalam konteks *collaborative filtering* penggunaan *similar score* terdapat cara mendapatkan *similar score* yaitu dalam *user-based collaborative filtering* diukur menggunakan matriks seperti *cosine similarity*. Sedangkan dalam *item-based* didapatkan dari kemiripan preferensi pengguna. Pada metode TFRS tidak ada metode tunggal yang secara khusus digunakan dalam menghitung *similar score*. Sehingga, dalam proses sistem rekomendasi disesuaikan dengan karakteristik dataset, konteks aplikasi, hingga preferensi dari pengembang.

Terdapat beberapa metode untuk menghitung *similar score* seperti, *cosine similarity* yang mengukur kesamaan antar dua buah vektor dalam ruang multidimensi. *Jaccard similarity* menghitung dari jumlah elemen yang ada. Selain itu, terdapat cara menggunakan *pearson correlation* dengan mengukur korelasi *linier* antar dua buah variabel. Banyak metode lainnya seperti *euclidean distance*, *manhattan distance*, dan lain-lain.

Pada penelitian ini menggunakan cara *cosine similarity* dalam menghasilkan *similar score*. Matriks *similarity* dihitung dengan perkalian matriks. Sehingga, nilai *similar score* akan merepresentasikan tingkat kesamaan antar vektor-vektor pengguna. Nilai yang lebih tinggi akan menunjukkan tingkat kesamaan yang tinggi, sedangkan nilai yang lebih rendah begitupula menunjukkan tingkat kesamaan yang rendah. Dengan kata lain, nilai *similar score* berbanding lurus dengan tingkat kesamaan.

3.6 *Deployment*

Proses *deployment* yang dilakukan pada penelitian ini melibatkan dua buah metode yang berbeda yaitu metode USE menggunakan *firebase* sebagai platform pengembangan aplikasi. *Firebase* merupakan *platform* pengembangan aplikasi seluler dan *web* yang memiliki berbagai fitur seperti penyimpanan data, otentikasi pengguna, analitik, dan *machine learning*. Berikut ini menunjukkan lokasi file *firebase* pada Gambar 3.5 berikut.



Gambar 3.5 Lokasi File *Firebase*

Pada Gambar 3.5 merupakan lokasi file Firebase pada Google Colab yang digunakan untuk melakukan komputasi. *Firebase* digunakan pada penelitian ini sebagai *Cloud* untuk menyimpan dan menyinkronkan data aplikasi hingga aturan keamanan. Selain itu, pada penelitian ini menggunakan fitur layanan *machine learning* untuk menerapkan model *machine learning* ke dalam aplikasi Collabolio.