BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini berisi spesifikasi dan kemampuan dari sistem yang akan diimplementasi, pendekatan dan sumber riset yang akan dipelajari untuk menunjang pembuatan sistem, serta rancangan tahapan yang akan dilakukan dalam pengimplementasian sistem. Pemilihan *teknologi* atau *framework*, proses pembersihan, pembagian data, metode pelatihan, dan pengujian akan dijabarkan pada bab ini.

3.1 Fitur dan Spesifikasi Penelitian

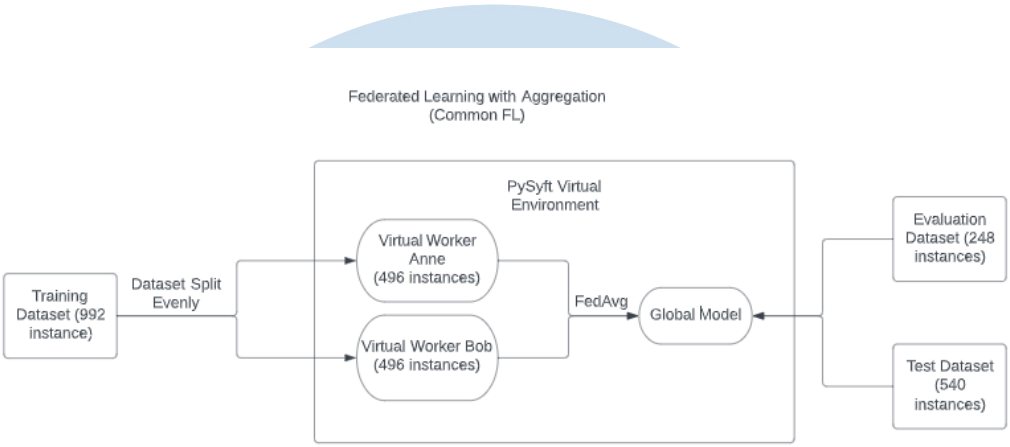
Fitur dan spesifikasi dari SMS Scam Detector yang dilatih dengan cara Federated Learning adalah sebagai berikut.

1) Model yang dapat melakukan pengklasifikasian pesan SMS berupa teks menjadi dua kelas: *scam* dan *non-scam.*
2) Pelatihan secara lokal dari dua *client* yang dilakukan agregasi dan memperbaharui *global model* setiap *epoch*-nya.
3) Pelatihan secara *daisy-chaining* dari dua *client* yang tidak melakukan agregasi dan menghasilkan *global model* di akhir pelatihan.

3.2 Studi Literatur

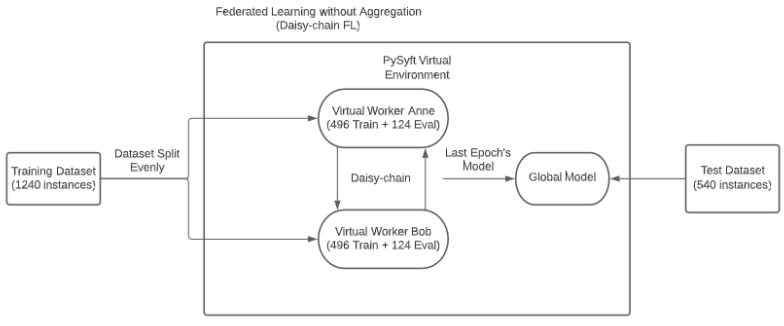
Penelitian dilakukan dengan cara mencari dan mempelajari berbagai referensi terkait dengan Federated Learning dan pendukung lainnya. Referensi yang dipelajari berupa laporan karya ilmiah milik peneliti internasional, dokumentasi daring, dan berbagai artikel dari situs Medium. Beberapa *framework* Federated Learning juga memiliki halaman repositori GitHub yang memberikan contoh cara penggunaan *framework* dan penjelasan berbentuk berkas *readme*.

3.3 Rancangan Sistem



Gambar 3. 1 Common Federated Learning System Overview Diagram

Gambar 3.1 menggambarkan garis besar dari sistem Machine Learning yang dibuat. *Dataset* sejumlah 992 dibagi rata ke kedua Virtual Worker yang diberi nama Anne dan Bob. Kedua Virtual Worker melatih menggunakan data dan memperbaharui *global model* dengan algoritma Federated Averaging. Setiap kedua Virtual Worker selesai melatih, maka *global model* dievaluasi untuk menghitung *evaluation loss* menggunakan 248 data. Jika seluruh tahap *training* telah selesai, 540 data digunakan untuk *testing* penentu performa model.



Gambar 3. 2 Daisy-chain Federated Learning System Overview Diagram

Selanjutnya proses pelatihan Federated Learning tanpa agregasi (Daisy-chain Federated Learning) dilakukan. *Training data* sejumlah 1240 dibagi secara rata ke

dua Virtual Worker. Setiap Virtual Worker melatih dan melakukan evaluasi menggunakan data tersebut. Ketika satu Virtual Worker telah menjalankan *training epoch*-nya, model dipindahtangankan ke Virtual Worker lain. Jika seluruh *epoch* telah selesai dijalankan, model pada *epoch* terakhir dijadikan *global model* yang melalui tahap *testing* menggunakan 540 data.

3.3.1 Environment Preparation

Menyiapkan berbagai macam *library* bahasa pemrograman Python yang digunakan untuk memuat model, memproses teks, dan lain-lain. *Framework* yang digunakan untuk menerapkan Federated Learning adalah PySyft yang menyediakan abstraksi berupa SyftTensor. Abstraksi tersebut merepresentasikan *state* dari sebuah data yang dimiliki oleh *client* tanpa memberikan informasi mengenai apa data tersebut. PySyft membutuhkan beberapa fungsi PyTorch seperti pembuatan Tensor sehingga *library* PyTorch juga digunakan. PySyft dapat membuat dua macam Tensor yaitu LocalTensor yang dikirimkan dan diubah oleh *client* dan PointerTensor yang menunjuk pada LocalTensor di *client*. PySyft menyediakan metode *send* untuk mengirim Tensor dan *get* untuk mengambil kembali hasil parameter Tensor dari *client*.

Simulasi *client* digunakan dengan membuat Virtual Workers. PySyft menyediakan fitur tersebut untuk mempermudah proses *debugging* dari operasi kompleks yang ada dalam Federated Learning. Seluruh Virtual Worker yang dibuat berada pada satu mesin sehingga tidak diperlukannya konfigurasi *networking*.

3.3.2 Dataset Collection

Mengumpulkan data berupa pesan SMS yang umum diterima oleh masyarakat Indonesia. *Dataset* SMS berbahasa Indonesia dicari dari internet berasal dari *blog* Yudi Wibisono [14] memiliki 1144 *instance* dengan tiga kelas yaitu: SMS normal, SMS penipuan, dan SMS promosi. Jumlah data ditambah sebanyak 636 secara manual oleh penulis. Pengambilan secara manual dilakukan dengan cara meminta izin akses SMS dari *smartphone* orang yang

dikenal penulis dan mengambil SMS menggunakan *software* bernama Backuptrans.

3.3.3 Dataset Partition

Membagi *dataset* untuk beberapa *client* atau *virtual worker*. Setiap pembagian untuk masing-masing *client* juga dilakukan pembagian data untuk *training* dan *evaluation*. Data yang pada umumnya dimiliki oleh satu *client* tidak terlalu banyak. Jumlah *dataset* yang cenderung berjumlah sedikit digunakan untuk setiap *client*. Total jumlah *dataset* utama dibagi sesuai jumlah *virtual worker* yang dibuat. Lalu dari jumlah *dataset* yang dipegang oleh sebuah *virtual worker*, ditentukan sebuah persentase jumlah data yang dijadikan sebagai data untuk tahap *evaluation*, dan sisanya digunakan pada tahap *training*. Total *dataset* sebanyak 1780 yang terdiri dari 591 pesan *scam*, 598 pesan promo, dan 591 pesan umum. Total pesan *non-scam* sejumlah 598 ditambah 591 yaitu 1189 *records*. Dikarenakan penggabungan kelas SMS promo dengan SMS teks biasa, maka rasio kelas SMS *scam* dengan *non*-scam menjadi tidak seimbang. Jumlah pembagian data untuk setiap *virtual worker* sesuai dengan metode dapat dilihat pada Tabel 3.1 dan Tabel 3.2.

Tabel 3. 1 Common Federated Learning Data Partition

Data Holder	Train Data		Eval Data		Sum by Data Holder
	Scam	Non-scam	Scam	Non-scam	
Anne	173	323	0	0	496
Bob	141	355	0	0	496
Global Model	0	0	83	165	248
Total					1240

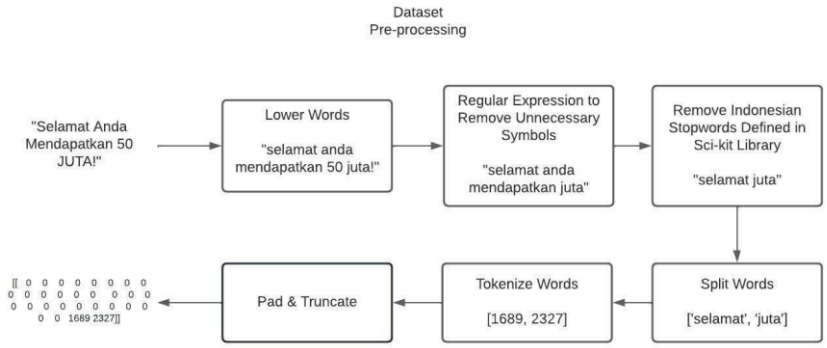
Tabel 3. 2 Daisy-chain Federated Learning Data Partition

Data Holder	Train Data		Eval Data		Sum by Data Holder
	Scam	Non-scam	Scam	Non-scam	
Anne	173	323	37	87	620
Bob	141	355	46	78	620
Global Model	0	0	0	0	0
Total					1240

3.3.4 Labelling

Secara manual memberikan label pada data menggunakan aplikasi *spreadsheet*. Terdapat dua kolom pada *dataset* yang disimpan dalam format Comma Seperated Value (CSV) yaitu “Teks” dan “Label”. Teks berisi pesan SMS yang terklasifikasi menjadi penipuan, promosi, dan pesan umum. Teks SMS yang merupakan *scam* diberi label “1”, sementara promosi dan pesan SMS umum diberi label “0”.

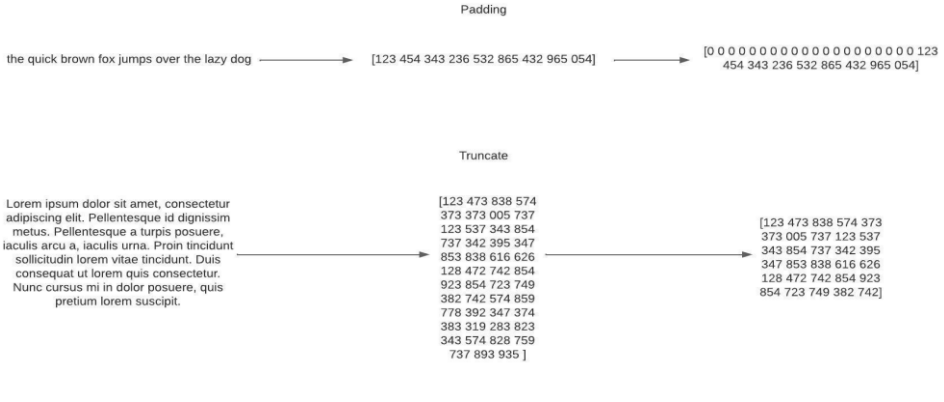
3.3.5 Text Pre-Processing



Gambar 3. 3 Dataset Pre-processing

Membuang karakter tidak berguna menggunakan *library* regex, *stopwords removal*, *truncation* atau pemotongan panjang data, *padding*, *tokenization*, dan mengonversikan ke bentuk berkas yang memiliki ekstensi *npy* atau *numpy array*. Pertama, seluruh teks dikonversikan ke huruf kecil dengan fungsi *lower*, lalu karakter simbol seperti “!@#$%^&*()” dibuang menggunakan Regex (Regular Expression). Beberapa kata yang sering muncul pada bahasa Indonesia dan tidak memiliki dampak yang relevan pada *training* disebut dengan *stop words* dan dibuang sesuai *stop words* pada bahasa Indonesia yang didefinisikan oleh *library* Sci-Kit Learn. Ketika teks telah bersih, setiap kata dipisah untuk membuat *word embedding* yang berbentuk *number index* sebagai representasi kata untuk model melakukan *training*. Kalimat yang memiliki panjang lebih dari *max length* dilakukan *truncation* sesuai maksimum panjang kalimat, sementara kalimat yang tidak mencapai *max length* diberi *padding* berupa token “0” di bagian kiri (*pre-padding*) sejumlah *max length* dikurangi

panjang kalimat tersebut. Proses *truncation* dan *padding* berguna agar semua teks yang akan menjadi input bersifat *uniform* atau seragam seperti yang diekspetasikan oleh *network* GRU. Input teks yang memiliki panjang seragam memiliki performa yang lebih baik karena *tensor* yang nanti dibentuk adalah *tensor* dengan *fixed shapes*. Setiap kata yang muncul pada *dataset* dikonversi menjadi angka dalam proses pembuatan *word embedding tokenization*, dan terakhir data hasil *pre-processing* disimpan dalam bentuk *numpy array* yang disimpan dalam bentuk *numpy binary file* berekstensi "npy".



Gambar 3. 4 Pad dan Truncate

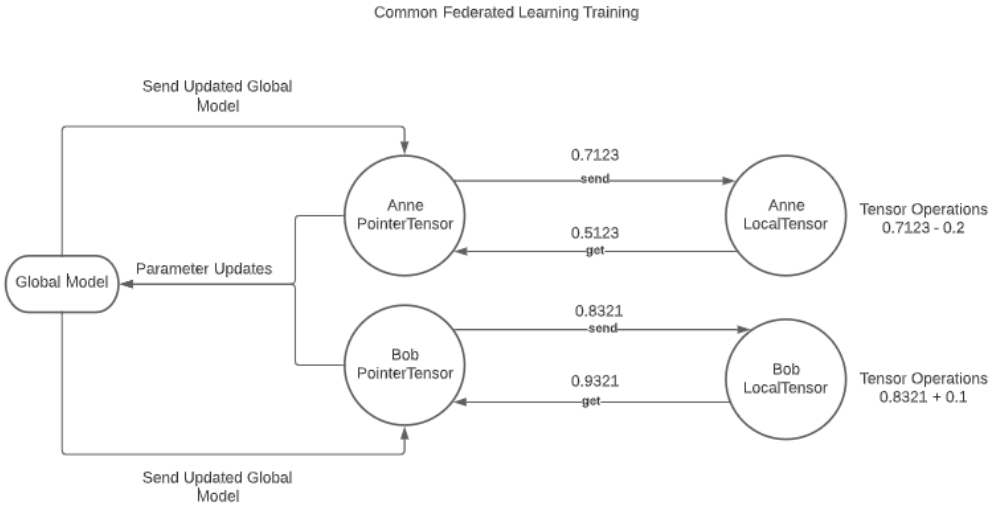
3.3.6 Load GRU Model

Memuat model GRU yang cocok dengan Federated Learning dan mengatur *parameter* yang dibutuhkan.

3.3.7 Optimasi Model

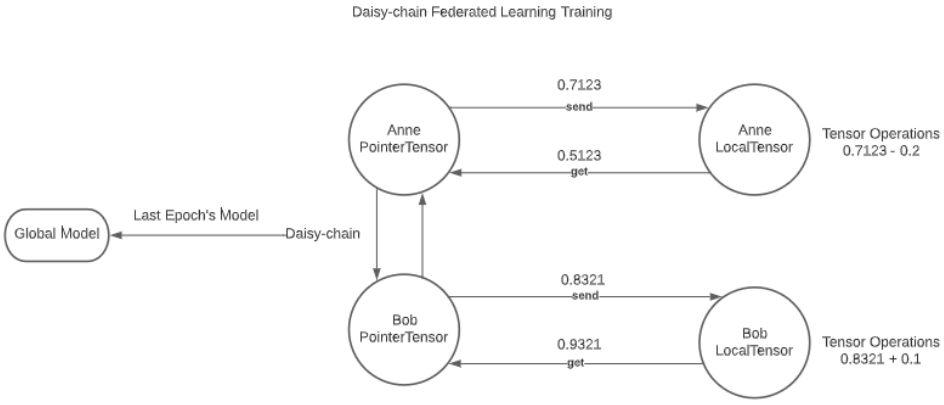
Memprogram alur algoritma agregasi Federated Averaging (FedAvg) dan algoritma pengiriman parameter *global model* ke setiap Virtual Worker. Beberapa *hyperparameter* umum pada proses *training* Machine Learning seperti *epoch, batch size, learning rate* didefinisikan sebagai konstan. Parameter yang mempengaruhi model GRU seperti jumlah *hidden dimension, embedding dimension*, nilai *dropout* juga didefinisikan sebagai konstan. Konstan tersebut diubah-ubah untuk memperoleh hasil *training* yang baik.

3.3.8 Training



Gambar 3. 5 Mekanisme Tensor dan Update Model Common Federated Learning

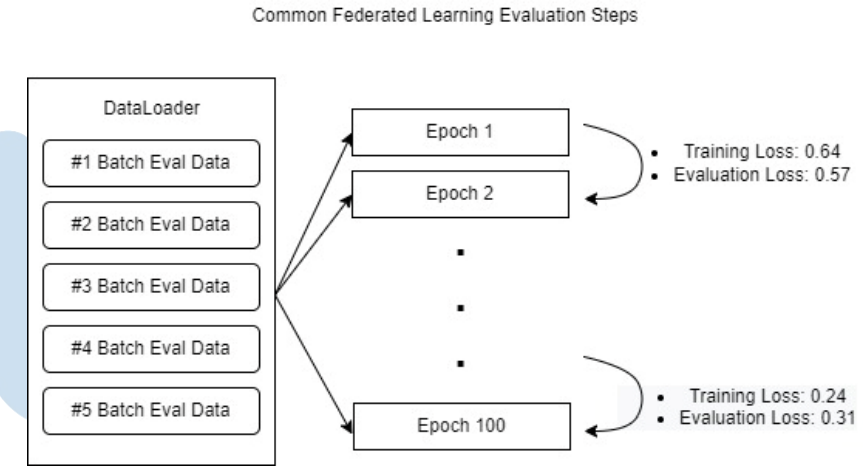
Melakukan pelatihan model menggunakan mesin lokal. Pelatihan model secara Common Federated Learning di mana dilakukan beberapa ronde pelatihan untuk setiap *client* atau *Virtual Worker* dan agregasi untuk *global model* menggunakan algoritma Federated Averaging. Pelatihan dilakukan dengan data yang ada pada mesin lokal, tapi telah dibagi untuk setiap *virtual worker*. Penyimulasian pembuatan pengiriman LocalTensor dan pembuatan PointerTensor juga dilakukan pada tahap ini. Tensor dibuat dan dikirim ke *virtual worker* dengan metode *send* yang disediakan oleh PySyft. *Training* dilakukan oleh *local model* pada *virtual worker* dan Tensor yang telah diubah oleh *virtual worker* dari hasil *training* dikembalikan untuk meng-*update global model*.



Gambar 3. 6 Mekanisme Tensor dan Update Model Daisy-chain Federated Learning

Sama halnya dengan cara Common Federated Learning, komputasi pada Daisy-chain Federated Learning dilakukan dengan menggunakan mekanisme PointerTensor dan LocalTensor. Seperti yang telah dijelaskan pada subbab sebelumnya, yang membedakan Daisy-chain adalah tidak adanya agregasi.

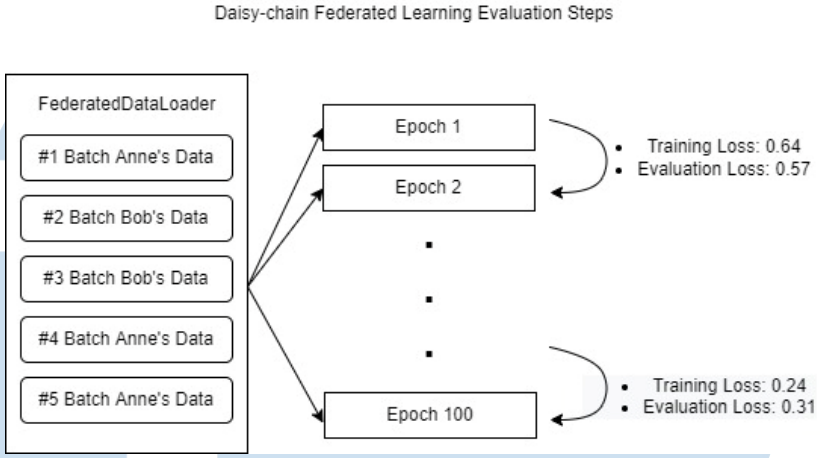
3.3.9 Evaluation



Gambar 3. 7 Langkah Evaluasi Common Federated Learning Setiap Epoch

Pada setiap *epoch*, model dievaluasi untuk melihat kemajuan *training* model. Metrik yang ditampilkan untuk melihat kemajuan adalah *evaluation loss*. Jika semakin rendah nilai *evaluation loss*, maka semakin baik model belajar. Dalam Common Federated Learning, *global model* hasil agregasi setiap

epoch dievaluasi menggunakan *evaluation data* yang telah disebutkan sebanyak 20%. Evaluasi dilakukan pada *local machine* dan tidak dilakukan pada Virtual Worker manapun.



Gambar 3. 8 Langkah Evaluasi Daisy-chain Federated Learning Setiap Epoch

Beda halnya dengan Daisy-chain Federated Learning, evaluasi tetap memakai *evaluation data* sebanyak 20%, namun setiap *evaluation data* memiliki Virtual Worker yang menjadi pemilik suatu *instance data*. Jika pada satu *epoch* Anne telah selesai tahap *training*, maka *evaluasi* dilakukan pada Virtual Worker Anne juga menggunakan *evaluation data* miliknya.

3.3.10 Testing

Melakukan pengujian atau *inference testing* terhadap 540 pesan SMS. Kalimat SMS diberikan ke *saved model* dan inferensi dilakukan untuk menentukan apakah teks SMS tersebut merupakan *scam* atau *non-scam*. Model yang tersimpan dan siap dilakukan *testing* adalah *global model* dari masing-masing metode *training*. Data untuk melakukan *testing* adalah sebanyak 540 SMS yang dikumpulkan dan diberi label secara manual, serta disediakan dalam bentuk CSV. Sebanyak 540 SMS yang digunakan untuk *testing* adalah data yang tidak pernah digunakan untuk *training*. Besaran 540 didapat dari kurang lebih 30% dari data *training* [15]. Akurasi dari hasil *testing* akan diberikan dalam bentuk ROC AUC, Classification Accuracy, F1 Score yang terdiri dari Precision dan Recall, serta visualisasi dengan Confusion Matrix yang terdiri dari True Positive, False Positive, True Negative, dan False Negative.

3.4 Dokumentasi

Dokumentasi dilakukan dengan menyimpan *source code* di repositori daring Github. *Source code* juga disediakan dalam bentuk Python Notebook untuk memudahkan *developer* lain menganalisis dan mencoba proyek.