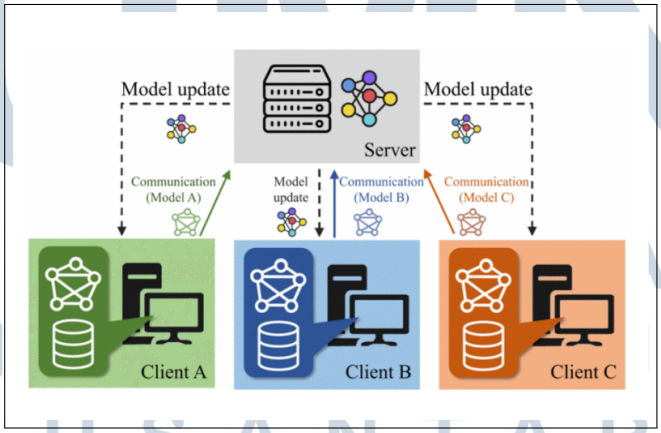
BAB 2
LANDASAN TEORI

2.1 Federated Learning

Federated Learning (FL) merupakan pendekatan pembelajaran mesin terdistribusi yang memungkinkan sejumlah klien melatih model secara kolaboratif tanpa harus mengirimkan data mentah ke server pusat [1]. Melalui FL, setiap klien melatih model menggunakan data lokal masing-masing, kemudian hanya parameter atau pembaruan model yang dikirim ke server untuk dilakukan agregasi. Server selanjutnya akan memperbarui model global berdasarkan kontribusi dari setiap klien, sehingga proses pembelajaran dapat dilakukan secara kolaboratif tanpa memindahkan data lokal [1, 3].

Struktur umum FL terdiri atas satu server pusat dan sejumlah klien yang berpatisipasi dalam proses pelatihan. Pada metode FL konvensional seperti *Federated Averaging* (FedAvg), server mengirimkan model global awal kepada klien untuk dilatih secara lokal. Setiap klien mengirimkan pembaruan model kembali ke server untuk diagregasi menjadi model global baru yang digunakan pada ronde pelatihan berikutnya setelah pelatihan selesai [2, 3]. Proses ini dilakukan secara iteratif hingga model mencapai performa atau konvergensi yang diharapkan. Gambar 2.1 memperlihatkan arsitektur umum *Federated Learning* dengan skema *client-server*.



Gambar 2.1. Arsitektur umum *Federated Learning* berbasis *client-server*.

Sumber: [14]

2.2 Knowledge Distillation

Knowledge Distillation (KD) merupakan metode dalam pembelajaran mesin yang bertujuan untuk mentransfer pengetahuan dari model besar dan kompleks (*teacher*) ke model yang lebih sederhana (*student*) [15, 16]. *Student model* tidak hanya belajar dari label asli, tetapi juga dari *output teacher model* yang mengandung informasi hubungan antar kelas atau *dark knowledge*. *Dark knowledge* dapat membantu *student model* memperoleh pemahaman yang lebih halus terhadap data dibandingkan pembelajaran yang hanya menggunakan *hard-label*. Dalam knowledge distillation, pengetahuan biasanya ditransfer melalui output model pada lapisan akhir, seperti *logits* atau *soft-label*.

2.2.1 Logits dan Soft-Label

Logits merupakan *output model* yang dihasilkan pada output layer sebelum diproses dengan fungsi *softmax*, sedangkan *soft-label* merupakan distribusi probabilitas yang diperoleh setelah *output model* tersebut diproses menggunakan fungsi *softmax* [10]. Berbeda dengan hard-label yang hanya menunjukkan satu kelas benar, soft-label menyimpan informasi probabilitas pada setiap kelas sehingga dapat merepresentasikan tingkat keyakinan model terhadap beberapa kemungkinan kelas.

Pada tugas klasifikasi dengan C kelas, model menghasilkan keluaran sebelum normalisasi yang kemudian diproses menggunakan fungsi *softmax* untuk membentuk distribusi probabilitas. Dalam SCARLET, keluaran yang telah dinormalisasi dan digunakan dalam proses distilasi disebut sebagai *soft-label*. Soft-label yang dihasilkan oleh model $\mathbf{w}$ untuk input $\mathbf{x}$ dinotasikan sebagai $\sigma(\mathbf{x};\mathbf{w})$. Proses pembentukan distribusi probabilitas menggunakan fungsi *softmax* ditunjukkan pada Persamaan 2.1.

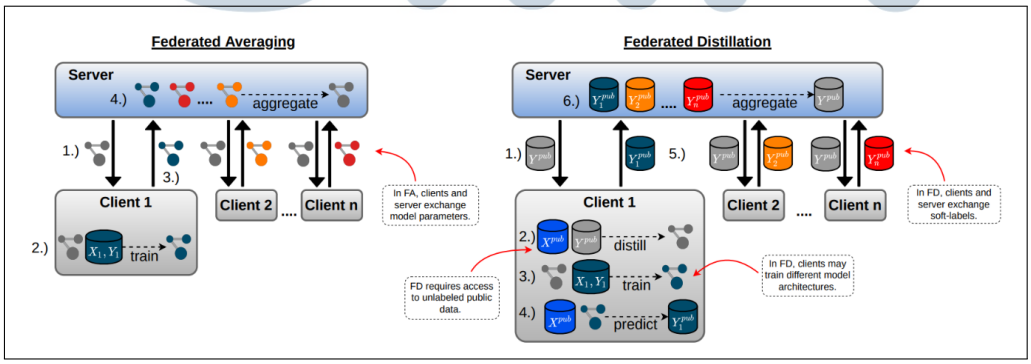
$$z_i = \frac{\exp(o_i)}{\sum_{j=1}^{C} \exp(o_j)} \tag{2.1}$$

dengan z_i merupakan probabilitas kelas ke-i, o_i merupakan nilai keluaran model sebelum normalisasi pada kelas ke-i, o_j merupakan nilai keluaran model sebelum normalisasi pada kelas ke-j, dan C merupakan jumlah kelas. Setelah melalui fungsi *softmax*, kumpulan probabilitas tersebut membentuk *soft-label* yang digunakan dalam proses distilasi.

2.3 Federated Distillation

Federated Distillation (FD) merupakan pengembangkan konsep *knowledge distillation* dalam lingkungan *Federated Learning*. FD memanfaatkan pertukaran *output model*, seperti *logits* atau *soft-label*, sebagai bentuk transfer pengetahuan antar klien dan server. Setiap klien tetap melakukan pelatihan lokal menggunakan data privat masing - masing, kemudian menghasilkan *soft-label* dari data publik yang tersedia. *Soft-label* tersebut dikirimkan ke server untuk diagregasi menjadi *soft-label* global yang digunakan sebagai terget distilasi pada proses pelatihan berikutnya.

Pada setiap communication round, server akan memilih sejumlah klien yang akan berpatisipasi. Klien yang terpilih kemudian akan melakukan sinkronisasi dengan server melalui pengunduhan soft-label yang sudah diagregasi pada data publik. Klien melakukan proses distilasi menggunakan soft-label tersebut sebagai target pembelajaran. Model hasil distilasi kemudian dilatih kembali menggunakan data privat lokal masing-masing klien. Setelah pelatihan lokal selesai, setiap klien menghasilkan *soft-label* baru pada data publik menggunakan model lokal yang telah diperbarui, lalu mengirimkannya kembali ke server. Server selanjutnya mengagregasi *soft-label* dari seluruh klien yang berpartisipasi untuk digunakan pada ronde komunikasi berikutnya. Perbedaan utama antara FL konvensional dan FD terletak pada bentuk informasi yang dipertukarkan antara klien dan server. Pada *Federated Averaging*, klien dan server bertukar parameter model sebagai representasi hasil pelatihan lokal. Sebaliknya, pada *Federated Distillation*, informasi pelatihan ditransfer melalui prediksi *soft-label* yang dihasilkan pada data publik bersama [17]. Perbedaan alur komunikasi dan komputasi antara *Federated Averaging* dan *Federated Distillation* ditunjukkan pada Gambar 2.2.



Gambar 2.2. Alur data dan komputasi pada *Federated Averaging* dan *Federated Distillation*.

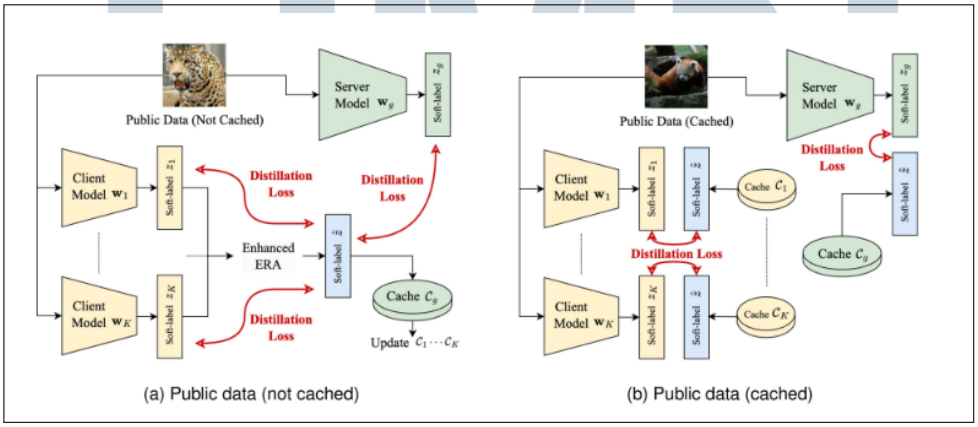
Sumber: [18]

2.4 Semi-supervised federated distillation with global Caching and Reduced soft-Label Entropy (SCARLET)

SCARLET merupakan *framework distillation-based Federated Learning* yang dikembangkan untuk meningkatkan efisiensi komunikasi dan kualitas agregasi *soft-label*. SCARLET menggunakan mekanisme *soft-label caching* agar *soft-label* yang masih tersedia di *cache* dapat digunakan kembali. Selain itu, SCARLET mengganti mekanisme ERA konvensional dengan *Enhanced Entropy Reduction Aggregation (Enhanced ERA)* untuk menghasilkan pengaturan ketajaman (*sharpness*) soft-label yang lebih stabil melalui parameter β [10].

2.4.1 Soft-Label Caching

Soft-label caching merupakan mekanisme utama dalam SCARLET yang digunakan untuk mengurangi komunikasi redundan dalam proses *Federated Distillation*. Pada mekanisme ini, server menyimpan *soft-label* hasil agregasi pada *global cache* C_g, sedangkan klien menyimpan *soft-label* terkait pada *local cache*. Pada awal setiap *communication round*, server memilih subset data publik P^t yang akan digunakan untuk distilasi. Server kemudian memeriksa indeks sampel pada subset tersebut dan membentuk daftar permintaan I^t_{req} yang berisi sampel publik yang belum tersedia pada *global cache*. Gambar 2.3 memperlihatkan mekanisme *soft-label caching* pada SCARLET.



Gambar 2.3. Mekanisme *soft-label caching* pada SCARLET.
Sumber: [10]

Pada kondisi data publik belum tersimpan di *cache*, klien mengirimkan *soft-label* ke server untuk diagregasi menggunakan *Enhanced ERA*. Hasil agregasi

tersebut kemudian disimpan pada *global cache* dan disinkronkan dengan *local cache* klien. Sebaliknya, ketika data publik telah tersedia pada *cache*, server dan klien dapat menggunakan kembali *soft-label* yang tersimpan untuk proses distilasi tanpa perlu melakukan pengiriman *soft-label* tambahan.

Secara ringkas, alur *soft-label caching* pada SCARLET dapat dituliskan sebagai berikut.

Algorithm 1 *Soft-Label Caching* pada SCARLET

1: Server memilih subset data publik P^t pada ronde ke-t
2: Server memeriksa ketersediaan *soft-label* pada *global cache* C_g
3: Server membentuk daftar permintaan I^t_{req} untuk sampel yang belum tersedia pada C_g
4: Server mengirimkan I^t_{req} kepada klien
5: Klien menghasilkan *soft-label* hanya untuk sampel dalam I^t_{req}
6: Klien mengirimkan *soft-label* yang diminta kepada server
7: Server mengagregasi *soft-label* yang diterima menggunakan *Enhanced ERA*
8: Server menyimpan hasil agregasi ke dalam *global cache* C_g
9: Server dan klien menggunakan kembali *soft-label* yang tersimpan pada *cache* untuk ronde berikutnya

2.4.2 Parameter β pada *Enhanced ERA*

Parameter β pada *Enhanced ERA* berfungsi untuk mengatur tingkat ketajaman distribusi soft-label hasil agregasi. Setiap probabilitas kelas pada rata-rata soft-label dipangkatkan dengan nilai β. Ketika $\beta = 1$, proses *Enhanced ERA* menghasilkan distribusi yang sama dengan rata-rata *soft-label*, sehingga agregasi setara dengan *simple averaging*. Ketika $\beta > 1$, probabilitas kelas yang lebih besar akan semakin dikuat sehingga distribusi soft-label menjadi lebih tajam (condong ke 1 kelas) dan nilai entropinya menurun.

Pemilihan nilai β menjadi penting karena tingkat ketajaman dapat mempengaruhi kualitas sinyal distilasi. Nilai β yang terlalu kecil dapat membuat *soft-label* tetap ambigu, terutama ketika hasil agregasi memiliki entropi tinggi. Namun, nilai β yang terlalu besar dapat membuat distribusi menjadi terlalu tajam dan berpotensi menghilangkan informasi hubungan antar kelas (*dark knowledge*) yang terdapat pada *soft-label*. Oleh karena itu, nilai β perlu dipilih secara tepat agar hasil agregasi tidak kehilangan karakteristik probablistiknya dan tetap informatif.

Pada SCARLET, nilai β masih digunakan sebagai *hyperparameter* statis yang ditentukan sebelum proses pelatihan. Penelitian SCARLET menunjukkan bahwa nilai β statis tertentu mampu menghasilkan performa yang kompetitif, tetapi otomasi penyesuaian β secara adaptif masih menjadi arah pengembangan lebih lanjut. Salah satu sinyal yang dapat digunakan untuk proses penyesuaian tersebut adalah entropi *soft-label* hasil agregasi yang tersedia pada server [10]. Hal ini menjadi dasar bagi penelitian ini untuk mengembangkan mekanisme *Adaptive* β berbasis entropi *soft-label* pada *Enhanced ERA*.

2.4.3 Enhanced Entropy Reduction Aggregation (Enhanced ERA)

Enhanced ERA merupakan mekanisme agregasi *soft-label* yang digunakan untuk menggantikan ERA konvensional pada SCARLET. Pada *Federated Distillation*, server menerima *soft-label* dari beberapa klien yang kemudian akan digabungkan menjadi *soft-label* global. Server menghitung rata - rata *soft-label* dari seluruh klien seperti pada Persamaan 2.2.

$$\bar{\mathbf{z}}i^t = \frac{1}{K}\sum_{k=1}^{K}\mathbf{z}_{k,i}^t \tag{2.2}$$

dengan $\bar{\mathbf{z}}i^t$ merupakan rata-rata *soft-label* untuk sampel publik ke-i, K merupakan jumlah klien, dan $\mathbf{z}_{k,i}^t$ merupakan *soft-label* dari klien ke-k.

Setelah rata-rata *soft-label* diperoleh, Enhanced ERA melakukan proses *sharpening* menggunakan operasi pangkat terhadap setiap elemen probabilitas. Proses Enhanced ERA dapat dirumuskan seperti pada Persamaan 2.3.

$$\hat{z}_i^t = \frac{(\bar{z}_i^t)^\beta}{\sum_{j=1}^{C}(\bar{z}_j^t)^\beta} \tag{2.3}$$

dengan $\hat{z}_i^t$ merupakan nilai *soft-label* global setelah proses Enhanced ERA pada indeks ke-i dan round ke-t, $\bar{z}_i^t$ merupakan rata-rata *soft-label* pada indeks ke-i sebelum proses *sharpening*, C merupakan jumlah kelas probabilitas dalam distribusi *soft-label*, dan β merupakan parameter yang mengatur tingkat ketajaman distribusi probabilitas.

Berdasarkan persamaan tersebut, server terlebih dahulu menghitung rata-rata *soft-label* yang diterima dari seluruh klien untuk setiap sampel data publik. Setiap probabilitas kelas pada hasil agregasi kemudian dipangkatkan menggunakan

parameter β. Hasil pemangkatan tersebut dinormalisasi kembali agar jumlah probabilitas seluruh kelas tetap bernilai satu. Nilai β mengatur tingkat ketajaman distribusi, yaitu $\beta = 1$ mempertahankan distribusi rata-rata, sedangkan nilai $\beta > 1$ menghasilkan distribusi yang lebih tajam.

2.4.4 Distilasi pada Client dan Server

Pada SCARLET, proses distilasi dilakukan pada dua sisi, yaitu client dan server. Distilasi digunakan untuk melatih model agar prediksinya pada public dataset mendekati *soft-label* global hasil agregasi. Berbeda dengan pelatihan lokal yang menggunakan label asli dari private dataset, proses distilasi tidak menggunakan label asli, melainkan menggunakan distribusi probabilitas kelas sebagai target pembelajaran. Public dataset yang digunakan pada ronde ke-t dinotasikan sebagai P_t, sedangkan global *soft-label* hasil agregasi pada ronde tersebut dinotasikan sebagai $\hat{z}^t$.

Pada awal ronde komunikasi berikutnya, client terlebih dahulu memperbarui model lokal menggunakan *soft-label* global dari ronde sebelumnya, yaitu $\hat{z}^{t-1}$, dan public dataset yang digunakan pada ronde sebelumnya, yaitu P^{t-1}. Proses pembaruan model lokal melalui distilasi dituliskan sebagai berikut:

$$w_k^t \leftarrow w_k^t - \eta_{dist} \nabla \phi_{dist} \left(P^{t-1}, w_k^t, \hat{z}^{t-1} \right) \tag{2.4}$$

Pada persamaan tersebut, w_k^t menunjukkan parameter model lokal milik client ke-k pada ronde ke-t, η_{dist} menunjukkan *learning rate* pada proses distilasi, dan ϕ_{dist} menunjukkan fungsi *loss* yang digunakan dalam proses distilasi. Fungsi *loss* tersebut digunakan untuk mengukur perbedaan antara prediksi model client terhadap public dataset dan global *soft-label* yang diterima dari server. Pada SCARLET, fungsi ϕ_{dist} dapat direpresentasikan menggunakan Kullback–Leibler divergence.

Selain pada client, proses distilasi juga dilakukan pada model global di server. Setelah server menerima local *soft-label* dari client dan membentuk global *soft-label* $\hat{z}^t$, model global w_g^t diperbarui menggunakan public dataset P^t dan global *soft-label* tersebut. Secara umum, proses pembaruan model global melalui distilasi dapat dituliskan sebagai berikut:

$$w_g^t \leftarrow w_g^t - \eta_{dist} \nabla \phi_{dist} \left(P^t, w_g^t, \hat{z}^t \right) \tag{2.5}$$

Pada tahap ini, server tidak menggunakan label asli dari public dataset, melainkan menggunakan global *soft-label* $\hat{z}^t$ sebagai target pembelajaran. Dengan demikian, model global dilatih untuk meniru pengetahuan kolektif client yang telah diringkas dalam bentuk distribusi probabilitas kelas.

Proses distilasi dilakukan menggunakan data publik dan *soft-label* global sebagai target pembelajaran. Pada setiap *mini-batch*, model menghasilkan distribusi probabilitas prediksi dalam bentuk log-probabilitas. Selanjutnya, *Kullback–Leibler Divergence* dihitung antara *soft-label* global dan distribusi prediksi model. Nilai *loss* tersebut digunakan untuk memperbarui parameter model melalui proses *backpropagation*. Logika proses distilasi ditunjukkan pada Algoritma 2.

Algorithm 2 Proses Distilasi Menggunakan KL Divergence

Require: Model M, indeks data publik I, *soft-label* global $\hat{z}^t$, dan jumlah *epoch* distilasi E

Ensure: Nilai rata-rata *distillation loss*

Ambil data publik berdasarkan indeks I
Pasangkan data publik dengan *soft-label* global $\hat{z}^t$
Atur model ke mode pelatihan
$total\_loss \leftarrow 0$
$total\_sample \leftarrow 0$
for $epoch \leftarrow 1$ **to** E **do**
    for setiap pasangan *mini-batch* data publik X dan *soft-label* z^t **do**
        Hitung keluaran model $M(X)$
        Hitung log-probabilitas keluaran model menggunakan *log-softmax*
        Hitung *loss* KL Divergence
        Kosongkan gradien model
        Lakukan *backpropagation*
        Perbarui parameter model
        if $epoch = E$ **then**
            Tambahkan nilai *loss* ke $total\_loss$
            Tambahkan jumlah sampel ke $total\_sample$
        end if
    end for
end for
Hitung rata-rata *loss* dari $total\_loss$ dan $total\_sample$
return rata-rata *distillation loss*

Pada Algoritma 2, $\mathbf{X}$ merupakan *mini-batch* data publik, sedangkan $\hat{\mathbf{Z}}$ merupakan *soft-label* global yang digunakan sebagai target distilasi. Distribusi prediksi model diperoleh melalui fungsi *softmax*. Selanjutnya, KL Divergence mengukur perbedaan antara distribusi target dan distribusi prediksi model. Semakin kecil nilai KL Divergence, semakin dekat distribusi prediksi model terhadap *soft-label* global.

2.5 Shannon Entropy pada Soft-Label

Tingkat ketidakpastian dari distribusi probabilitas tersebut dapat diukur menggunakan *Shannon entropy*. Dengan menyesuaikan formulasi tersebut terhadap *soft-label* hasil agregasi, nilai *Shannon entropy* untuk sampel ke-i pada *round* ke-t dihitung menggunakan Persamaan 2.6.

$$H(\bar{\mathbf{p}}_{t,i}) = -\sum_{c=1}^{C} \bar{p}_{t,i,c} \log_2 (\bar{p}_{t,i,c}), \tag{2.6}$$

dengan $\bar{p}_{t,i,c}$ merupakan probabilitas kelas ke-c pada *soft-label* hasil agregasi, sedangkan C merupakan jumlah kelas pada proses klasifikasi. Nilai entropi yang lebih tinggi menunjukkan distribusi probabilitas yang lebih merata dan tingkat ketidakpastian yang lebih tinggi, sedangkan nilai entropi yang lebih rendah menunjukkan distribusi yang lebih terkonsentrasi pada kelas tertentu. Normalisasi menghasilkan nilai entropi pada rentang 0 hingga 1, dengan nilai yang mendekati 1 menunjukkan ketidakpastian maksimum dan nilai yang mendekati 0 menunjukkan distribusi prediksi yang lebih terkonsentrasi.

2.6 Exponential Moving Average (EMA)

Exponential Moving Average (EMA) merupakan teknik pemulusan data berurutan yang memberikan bobot lebih besar pada observasi terbaru, sedangkan bobot observasi sebelumnya berkurang secara eksponensial seiring bertambahnya jarak waktu. EMA dapat mengurangi pengaruh perubahan jangka pendek dengan tetap mempertahankan kecenderungan dari observasi sebelumnya [13]. Penerapan mekanisme pembobotan eksponensial pada deret nilai entropi juga dapat mengurangi variansi nilai entropi yang diamati secara berurutan [11].

EMA dihitung secara rekursif sebagai kombinasi antara nilai EMA sebelumnya dan observasi terbaru menggunakan suatu koefisien pemulusan. Formulasi EMA ditunjukkan pada Persamaan 2.7.

$$E_t = \rho E_{t-1} + (1-\rho) x_t, \tag{2.7}$$

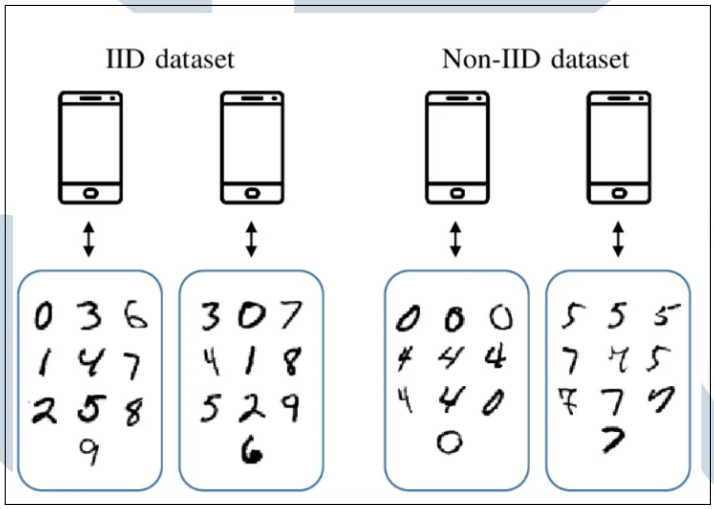
dengan E_t merupakan nilai EMA pada waktu ke-t, E_{t-1} merupakan nilai EMA sebelumnya, x_t merupakan observasi terbaru, dan ρ merupakan koefisien pemulusan dengan rentang nilai $0 \leq \rho < 1$.

2.7 Dataset dan Model

Penelitian ini menggunakan dataset MNIST dan EMNIST sebagai data eksperimen pada skema *Federated Distillation*. MNIST digunakan sebagai *private dataset* pada sisi klien untuk proses distilasi.

2.7.1 MNIST dan EMNIST

MNIST merupakan dataset citra tulisan tangan yang berisi digit angka dari 0 hingga 9. Setiap citra pada MNIST umumnya berupa gambar grayscale berukuran 28×28 piksel [19]. Gambar 2.4 memperlihatkan contoh distribusi data MNIST pada kondisi *non-IID*. Pada kondisi ini, setiap klien memiliki proporsi kelas yang berbeda sehingga distribusi data antar klien tidak seragam. Sebagian klien dapat memiliki data yang didominasi oleh kelas tertentu, sedangkan klien lain memiliki distribusi kelas yang berbeda. Kondisi tersebut menunjukkan heterogenitas data antar klien yang umum dijumpai pada skema *Federated Learning*.

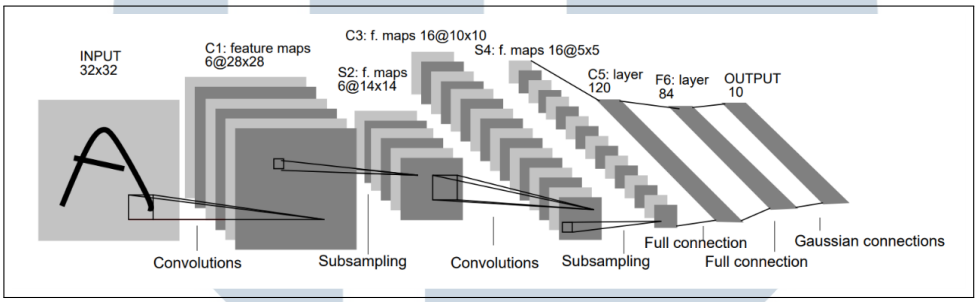


Gambar 2.4. Contoh distribusi data MNIST pada kondisi *non-IID*.
Sumber: [20]

EMNIST merupakan pengembangan dari MNIST yang berisi citra karakter tulisan tangan dengan cakupan kelas yang lebih luas, seperti huruf dan digit [21]. Pada penelitian ini, EMNIST digunakan sebagai *public dataset* yang tersedia bagi klien dan server dalam proses *Federated Distillation*. Data publik tersebut digunakan untuk menghasilkan *soft-label* pada sisi klien dan sebagai media transfer pengetahuan antar model tanpa perlu membagikan data privat klien secara langsung.

2.7.2 LeNet-5

LeNet-5 merupakan arsitektur *Convolutional Neural Network* (CNN) yang dikembangkan untuk tugas pengenalan citra tulisan tangan. CNN bekerja dengan memanfaatkan lapisan konvolusi untuk mengekstraksi fitur spasial dari citra, lapisan pooling untuk mengurangi ukuran representasi fitur, serta lapisan *fully connected*. Karena memiliki struktur yang relatif sederhana dan jumlah parameter yang tidak terlalu besar, LeNet-5 sesuai digunakan pada dataset citra sederhana seperti MNIST dan EMNIST [22]. Gambar 2.5 memperlihatkan arsitektur umum LeNet-5.



Gambar 2.5. Arsitektur umum LeNet-5.

Sumber: [22]

2.8 Metrik Evaluasi

Metrik evaluasi digunakan untuk menilai kinerja metode yang diterapkan dalam penelitian ini. Evaluasi dilakukan terhadap performa model. Penggunaan beberapa metrik tersebut bertujuan untuk mengetahui pengaruh penerapan *Adaptive* β berbasis entropi pada *Enhanced ERA* terhadap efektivitas proses distilasi dalam SCARLET.

Accuracy digunakan untuk mengukur kemampuan model global dalam melakukan klasifikasi dengan benar terhadap data uji. Pada penelitian ini, nilai *accuracy* dihitung pada sisi server melalui proses evaluasi model global terhadap data uji MNIST. Prediksi model diperoleh dari keluaran model pada setiap data uji, kemudian kelas dengan nilai prediksi tertinggi dibandingkan dengan label asli. Karena dataset yang digunakan adalah MNIST, nilai *accuracy* yang dilaporkan merupakan *top-1 accuracy*, yaitu jumlah prediksi benar dibagi dengan jumlah seluruh data uji.

Loss digunakan untuk mengukur tingkat kesalahan model global pada data uji. Pada penelitian ini, nilai *loss* dihitung menggunakan fungsi *cross-entropy*

antara keluaran model global dan label asli pada data uji MNIST. Nilai *loss* tersebut dihitung pada setiap batch data uji, kemudian dirata-ratakan untuk memperoleh nilai *loss* pada satu ronde komunikasi. Dengan demikian, *loss* yang dilaporkan pada hasil eksperimen merupakan *evaluation loss* model global, bukan *loss* pelatihan client maupun *loss* terhadap *soft-label*.