IMPLEMENTASI *DEEP LEARNING* MENGGUNAKAN *LIBRARY TENSORFLOW* DAN *APPLICATION PROGRAMMING INTERFACE* KERAS UNTUK KLASIFIKASI MULTI-OBJEK FOTO

Sahid Triambudhi[1)], Ihsan Doni Irawan[2)], dan Faisal Yusuf Fadhilah[3)]

[1]Teknik Informatika, Politeknik Negeri Indramayu
[2]Teknik Informatika, Politeknik Negeri Indramayu
[3]Teknik, Politeknik Negeri Indramayu
sahid@polindra.ac.id

Abstract

Perkembangan *deep learning* mendorong pemanfaatannya dalam klasifikasi citra karena mampu mengenali pola visual dengan akurasi tinggi. Penelitian ini bertujuan mengembangkan aplikasi berbasis web untuk klasifikasi multi-objek pada foto dua dimensi. Sistem dirancang menggunakan Python, TensorFlow, dan API Keras, dengan alur kerja pengiriman citra melalui AJAX dari *client* ke *server* TensorFlow, lalu menampilkan hasil klasifikasi secara *real-time*. Proses pengembangan mengikuti kerangka kerja scrum, melalui tahapan *sprint*, *backlog*, dan *review* agar adaptif terhadap perubahan kebutuhan. Pengembangan dibatasi pada jaringan privat menggunakan NAT atau *firewall*, serta terhubung ke internet melalui *Ngrok tunnel*. Hasil penelitian menunjukkan sistem mampu mengidentifikasi beberapa objek sekaligus, menampilkan nama serta tingkat akurasinya (%). Rekomendasi pengembangan selanjutnya meliputi integrasi perangkat keras (kamera/*webcam*), dukungan jaringan publik, klasifikasi lebih spesifik seperti "umur" atau "penyakit", serta perluasan format input termasuk dokumen *Word* dan *PDF*.

***Keywords**: Deep Learning, TensorFlow, Keras, API, Object Classification, Python.*

1. PENDAHULUAN

Perkembangan kecerdasan buatan menempatkan *deep learning* sebagai metode utama dalam visi komputer karena menawarkan akurasi dan efisiensi lebih tinggi dibandingkan teknik tradisional (Sarker, 2021). Implementasi *deep learning* dengan TensorFlow terbukti unggul dalam kecepatan pelatihan pada dataset besar seperti ImageNet (Jaiswal et al., 2022), sementara integrasi Keras API memudahkan perancangan dan pengujian arsitektur CNN (Chollet, 2023). TensorFlow Lite memperluas penerapan model ke perangkat mobile dan IoT (David et al., 2021), dan federated learning berbasis TensorFlow menjaga privasi data medis sekaligus menghasilkan diagnosis yang andal (Yang et al., 2021).

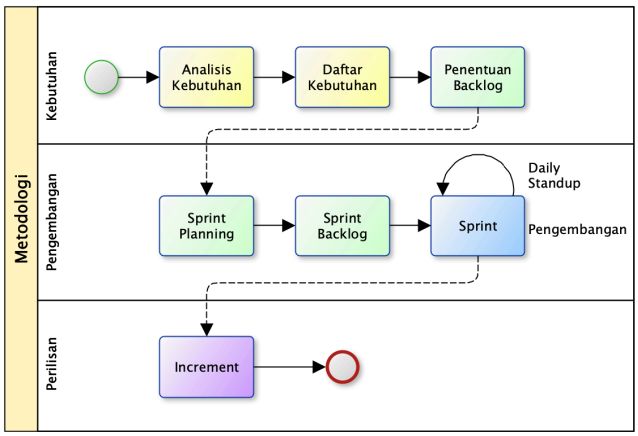
Sebagai bagian dari *machine learning*, *deep learning* meniru cara kerja otak manusia dan kini menjadi paradigma dominan untuk tugas kompleks seperti visi dan NLP (Alzubaidi et al., 2021). Perkembangannya ditandai dengan inovasi arsitektur (CNN, RNN, Transformer, GNN) dan teknik pelatihan (*self-supervised*, *federated*, *knowledge distillation*) yang memperluas aplikasi ke kesehatan dan ilmu material (Alzubaidi et al., 2021; Mienye & Swart, 2024). Misalnya, *library* TensorFlow yang saat ini sudah banyak diterapkan dalam deteksi kualitas material (Sinaga & Arnomo, 2024) dan analisis citra medis (Zhou et al., 2021).

Ekosistem perangkat lunak juga berkembang melalui integrasi tf.keras, TFX, TensorFlow Lite, hingga TensorFlow.js (Moroney, 2020; Ramchandani et al., 2022). Keras sendiri telah berevolusi menjadi Keras 3 dengan API *multi-backend* yang lebih portabel dan *scalable* (Chollet, 2023), sementara dokumentasi desain sistem yang lengkap jadi lebih memudahkan penggunaan API Keras dalam pengembangan *deep learning* (Nordio, 2022).

Berdasarkan uraian tersebut, penelitian ini difokuskan pada penerapan *deep learning* untuk dapat melakukan klasifikasi multi-objek foto (bersamaan) secara *real-time* melalui web page. Sistem memungkinkan unggah foto, klasifikasi simultan dengan TensorFlow dan Keras API, serta menampilkan nama objek dan tingkat kecocokan. Selain itu, dilakukan pengujian fungsionalitas untuk mengevaluasi kesesuaian desain awal dengan implementasi akhir.

2. METODE PENELITIAN

Proses penelitian dilakukan dengan menerapkan kerangka kerja yang secara garis besar, dibagi menjadi 3 (tiga), yaitu proses penentuan kebutuhan, pengembangan sistem dan perilisan. Lebih detail, alur kerangka kerja dalam penelitian yang akan dilakukan dapat dilihat pada gambar 1.



Gambar 1. Kerangka Kerja Penelitian

Pada tahap identifikasi kebutuhan dilakukan analisis, penyusunan daftar, dan penentuan backlog sebagai dasar pengembangan. Backlog dalam kerangka kerja Scrum berisi daftar tugas yang harus diselesaikan. Scrum sendiri merupakan framework manajemen proyek berbasis Agile dengan tiga fase utama: *product backlog*, *sprint backlog*, dan *increment* (Scrum.org, 2021). Implementasi Scrum yang baik meningkatkan kualitas perangkat lunak melalui kolaborasi, adaptasi, serta iterasi cepat (Alami & Krancher, 2022), dan mempermudah pengelolaan backlog maupun sprint (Verwijs & Russo, 2021).

Setelah backlog ditetapkan, penelitian berlanjut pada *sprint planning* untuk menentukan *backlog items* yang masuk ke *sprint backlog* dan harus dirilis pada *sprint* berikutnya. Tahap akhir adalah *increment*, yaitu hasil rilisan berupa tambahan fitur yang memenuhi semua kriteria *Definition of Done* (DoD) dan sesuai kebutuhan awal. Definition of Done (DoD) membantu tim Scrum memastikan setiap Increment memenuhi standar kualitas dan siap digunakan, meskipun tidak semuanya langsung dirilis (Kopczyńska et al., 2022).

3. HASIL DAN PEMBAHASAN

3.1. *Product Backlog*

Product backlog adalah artefak utama Scrum yang berupa daftar pekerjaan terprioritaskan, yang dikelola oleh Product Owner untuk mencapai *product goal*." (Purnama et al., 2022). *Product backlog* berisi daftar seluruh kebutuhan yang diperlukan dalam proses pengembangan suatu sistem atau aplikasi. Pembahasan mengenai *product backlog* bertujuan untuk menguraikan dan mendetailkan daftar tersebut ke dalam komponen-komponen yang lebih kecil dan spesifik, yang dikenal sebagai *product backlog item*.

Tabel 1. *Product Backlog Item*

ID	Nama	*Story Point*	*Plan* (Hari)	Catatan
PB1	Daftar Kebutuhan	8	1	Penyusunan kebutuhan sistem
PB2	Desain Antarmuka	8	1	Pembuatan rancangan antarmuka
PB3	Desain Alur Sistem	8	1	Pembuatan desain alur sistem
PB4	Kebutuhan Pengkodean	8	1	Pembuatan alur permainan.
PB5	Pengkodean	8	2	Proses Pengkodean
PB6	Register Ngrok	5	1	Register akun *tunnel* Ngrok
PB7	Kebutuhan *Sample* Foto	2	1	Dari internet
PB8	Uji *Black Box*	5	1	Penyusunan tahap uji *black box*
PB9	Dokumentasi	5	10	Dokumentasi pekerjaan

3.2. *Sprint Backlog*

Sprint Backlog dipecah menjadi tugas-tugas lebih rinci dan diperbarui secara harian oleh pengembang untuk mencapai *sprint goal* dan menghasilkan increment yang valid (Ahmad Jalaluddin et al., 2024). Pada penelitian ini, aktivitas *sprint planning* dimulai proses analisis berdasarkan kebutuhan dan menentukan *backlog items* yang akan dikerjakan atau masuk ke dalam *sprint backlog.* Hasilnya, *sprint backlog* akan berisi daftar *product backlog item* yang sudah disepakati untuk dikerjakan dan harus dirilis pada *sprint* selanjutnya.

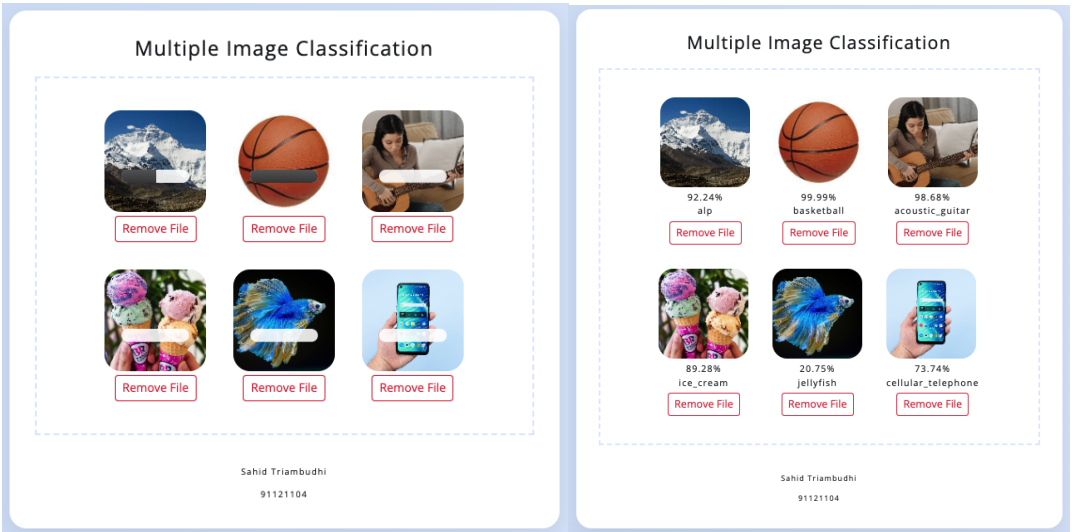
3.3. *Sprint*

3.3.1. Desain Antarmuka

Menampilkan segmentasi app.route hasil *render file* page-classification.html (/). Pada halaman awal, sudah terdapat fitur untuk melakukan *upload image* dengan 2 (dua) cara, yaitu *drag-and-drop* dan *manual upload* (via *explorer/finder*).

Gambar 2. Tampilan Antarmuka Awal

Saat mengirimkan gambar via API Keras, halaman *web* tidak akan melakukan *reload* sampai *response* diterima dari *server*, dan akan langsung ditampilkan hasilnya pada masing-masing gambar berupa persentase kecocokan dan nama objek klasifikasi. Tahap proses *train* - *prediction* bisa mengunggah gambar lebih dari 1, dengan ukuran gambar yang diperbolehkan adalah maksimal 5 MB. Jika melebihi ukuran tersebut, program akan memvalidasi dan menampilkan pesan *error* untuk pengguna. Proses *train* dan *prediction* dilakukan pada *servable TensorFlow*, jadi setiap proses dilakukan perlu koneksi internet.



Gambar 3. Tampilan Antarmuka *Train* (Kiri) dan *Result* (Kanan)

Tabel 2. Hasil Klasifikasi Multi-Objek

No	Foto	Klasifikasi	Sesuai?	Akurasi
1	Pegunungan Alpen	"alp"	✓	92.24%
2	Bola Basket	"basketball"	✓	99.99%
3	Perempuan dan Gitar	"accoustic_gitar"	✓	98.68%
4	Es Krim	"ice_cream"	✓	89.28%
5	Ikan Cupang	"jellyfish"	✘	20.75%
6	Smartphone	"cellular_telephone"	✓	73.74%

Hasil prediksi dari masing-masing gambar dipengaruhi oleh banyaknya model data yang ada pada *servable TensorFlow*. Semakin sering suatu citra dilatih dan diuji (*train - test*), maka nilai akurasi kecocokannya semakin tinggi, seperti "*basketball*" dengan nilai 99.99%. Contoh lain, klasifikasi objek "*jellyfish*" memiliki akurasi yang sangat rendah. Hal ini bisa dipengaruhi beberapa faktor, seperti minimnya *machine* dilatih untuk mengenali objek gambar dengan citra "ikan", atau model klasifikasi "*fish*" terlalu sedikit, sehingga nilai akurasinya rendah.

3.3.2. Implementasi TensorFlow dan API Keras

Potongan kode program pada gambar 10 menunjukan beberapa library yang diimport dan digunakan pada pembuatan *web-page* klasifikasi multi-objek foto. Terlihat ada flask, numpy dan tentu saja *TensorFlow* - Keras. Pemanggilan *library* menggunakan "alias" sehingga mempermudah cara pemakaian pada kode program.

```
from tensorflow.keras.models import load_model
from io import BytesIO

from PIL import Image, ImageFile
from tensorflow.keras.applications import MobileNet
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.mobilenet import preprocess_input
from tensorflow.keras.applications.mobilenet import decode_predictions
```

Gambar 4. Penggunaan TensorFlow pada Kode Program *Python*

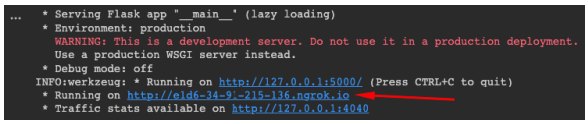
Setiap gambar yang berhasil diupload akan dikirimkan pada *servable TensorFlow* lalu dilakukan proses *train* dan *prediction* (decode_prediction) menggunakan fungsi tensorflow.keras.applications (MobileNet).

```
if file :
    ImageFile.LOAD_TRUNCATED_IMAGES = False
    img = Image.open(BytesIO(file.read()))
    img.load()
    img = img.resize((224, 224), Image.ANTIALIAS)

    x = []
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x_scale = preprocess_input(x)

    modelMobileNet = MobileNet(weights='imagenet')
    pred = modelMobileNet.predict(x_scale)
    result = decode_predictions(pred, top=1)

    for item in result[0]:
        # 'Nama Objek yang terdeteksi : ' +
        nama_objek = item[1]
        # 'Confidence : ' +
        persen_confidence = f"{float(item[2]):.2%}"

return jsonify(
    prediction=nama_objek,
    confidence=persen_confidence
)
```

Gambar 5. Potongan Kode Iterasi Proses Klasifikasi Multi-Objek

3.3.3. Ngrok *Tunneling*

Ngrok *tunneling* adalah proses yang memungkinkan untuk mengakses server lokal melalui URL publik yang dibuat oleh Ngrok, bahkan jika server tersebut berada di balik *firewall* atau NAT. Dengan kata lain, Ngrok memungkinkan untuk berbagi *server* lokal dengan orang lain di internet tanpa perlu konfigurasi jaringan yang rumit (Kimani, 2022). Setiap kali program berhasil di-*compile*, maka NgRok akan membuat jalur akses aman dengan network address translation, sehingga menghasilkan random ip yang berbeda-beda seperti gambar 6.



Gambar 6. Hasil *Running* Ngrok *Tunneling*

3.3.4. Pengujian

Metode pengujian yang diterapkan untuk menguji fungsionalitas antarmuka dalam penelitian ini adalah *black box testing*. Pendekatan ini menilai kinerja sistem berdasarkan keluaran yang dihasilkan dari suatu masukan, tanpa memperhatikan struktur internal program, logika kode, atau detail implementasi teknis. Fokus utama pengujian adalah pada kesesuaian antara input dan output sistem dengan spesifikasi serta kebutuhan perangkat lunak yang telah ditentukan. *Systematic review* menegaskan kebutuhan pengujian *black box* untuk membuat model lebih transparan dan dapat diandalkan (Muhammad et al., 2024).

Tabel 3. Hasil Uji Fungsi

No	Antarmuka	Hasil
1	Menampilkan halaman awal	Berhasil
2	Menampilkan area *upload*	Berhasil
3	*Browse files - upload manual*	Berhasil
4	*Drag-and-drop files*	Berhasil
5	Menampilkan pesan jika gagal *upload*	Berhasil
6	Menampilkan *loading bar*	Berhasil
7	Menghapus masing-masing gambar	Berhasil
8	Mengirimkan gambar ke *servable TensorFlow* via API Keras	Berhasil
9	Memproses masing-masing gambar	Berhasil
10	Menampilkan hasil klasifikasi	Berhasil

Hasil uji fungsi menggunakan *black box* pada 10 proses menunjukan bahwa semua komponen yang telah selesai dikembangkan berjalan sesuai dengan persentase 100% menunjukan hasil positif (*positive result*). Dengan demikian, dapat disimpulkan bahwa *sprint* telah selesai dengan semua *product backlog items* terverifikasi uji dan sesuai dengan kebutuhan.

3.4. *Increment*

Increment adalah tahapan akhir dari *scrum* menuju tercapainya tujuan pengembangan sistem. Setiap *backlog* yang selesai merupakan tambahan fungsional terverifikasi secara menyeluruh, memenuhi semua kebutuhan awal *backlog* tersebut. *Increment* disebut juga tambahan fitur dari setiap rilis *sprint*. Jadi dengan berakhirnya waktu pelaksanaan dan penyelesaian semua *task* dalam *sprint backlog*, maka tujuan pengembangan web klasifikasi multi-objek untuk penelitian ini selesai (dirilis).

Tabel 4. *Release Item's*

ID	Nama	Story Point	DoD?	Status
PB1	Daftar Kebutuhan	8	✓	Release
PB2	Desain Antarmuka	8	✓	Release
PB3	Desain Arsitektur Sistem	8	✓	Release
PB4	Kebutuhan Pengkodean	8	✓	Release
PB5	Pengkodean	8	✓	Release
PB6	Register NgRok	5	✓	Release
PB7	Kebutuhan *Sample* Foto	2	✓	Release
PB8	Uji *Black Box*	5	✓	Release
PB9	Dokumentasi	5	✓	Release
Total *Story Point*		**57**		

4. KESIMPULAN DAN SARAN

4.1. Kesimpulan

Penelitian ini menghasilkan beberapa kesimpulan yang menunjukkan keberhasilan penerapan teknologi *deep learning* menggunakan library TensorFlow. Pertama, berhasil mengimplementasikan metode *deep learning* dalam klasifikasi multi-objek foto secara *real-time*, sehingga sistem mampu mengenali lebih dari satu objek pada gambar dengan cepat dan akurat. Penerapan ini membuktikan bahwa teknologi *deep learning* dapat dioptimalkan untuk kebutuhan klasifikasi visual yang kompleks.

Kedua, penelitian juga berhasil membangun sebuah *web page* sederhana yang dilengkapi dengan fitur *drag-and-drop* untuk memudahkan pengguna dalam mengunggah gambar. *Web page* ini tidak hanya menampilkan hasil klasifikasi, tetapi juga memberikan persentase tingkat kecocokan untuk setiap objek yang terdeteksi, dengan memanfaatkan TensorFlow dan API Keras sebagai kerangka utama.

Ketiga, hasil pengujian terhadap fungsi *web page* klasifikasi multi-objek foto menunjukkan performa yang baik dari segi kecepatan pemrosesan, ketepatan klasifikasi,

maupun pengalaman pengguna. Temuan ini menegaskan bahwa penelitian memiliki potensi untuk dikembangkan lebih lanjut ke dalam aplikasi dengan cakupan yang lebih luas.

4.2. Saran

Beberapa saran untuk pengembangan lanjutan *web page* klasifikasi multi-objek foto adalah sebagai berikut.

1. Dapat membaca objek dari perangkat keras (*camera,* webcam, dll).
2. Berjalan tidak hanya lewat *tunneling*, tapi juga pada jaringan publik, sehingga dapat digunakan secara luas.
3. Dapat dikembangkan lebih spesifik, misalkan hanya untuk mengenali objek klasifikasi "umur", "penyakit", "jenis kelamin", dll.
4. Dapat melakukan klasifikasi objek dengan ekstensi selain gambar, seperti word dan pdf.

DAFTAR PUSTAKA

Ahmad Jalaluddin, E. K. B., & Mahatma, K. (2024). Recommendation for Scrum-Based Software Development Process with Scrum at Scale: A Case Study of Software House XYZ. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 8*(3), 406. https://doi.org/10.1000/resti.v8i3.406.

Alami, A., & Krancher, O. (2022). *How Scrum adds value to achieving software quality? Empirical Software Engineering, 27*(8), 165. https://doi.org/10.1007/s10664-022-10208-4.

Alzubaidi, L. (2021). *Review of Deep Learning: Concepts, CNN Architectures, Challenges*. Journal of Big Data, 8(53). https://doi.org/10.1186/s40537-021-00444-8.

Chollet, F. (2023). Introducing Keras 3.0. Keras. Retrieved from https://keras.io/keras_3.

David, R., Duke, J., Jain, A., Janapa Reddi, V., Jeffries, N., Li, J., Warden, P. (2021). TensorFlow Lite Micro: Embedded machine learning for TinyML systems. *Proceedings of Machine Learning and Systems, 3*, 1–15. https://doi.org/10.48550/arXiv.2010.08678.

Jaiswal, A., Gianchandani, N., Singh, D., Kumar, V., & Sharma, R. (2022). Comparative analysis of deep learning frameworks for image classification. *Journal of King Saud University – Computer and Information Sciences, 34*(9), 6112–6122. https://doi.org/10.1016/j.jksuci.2020.10.018.

Kimani, K. (2022). *What is Ngrok? How does Ngrok Work?* Retrieved from https://sendbird.com/developer/tutorials/what-is-ngrok.

Kopczyńska, S., Piechowiak, J., Ochodek, M., & Nawrocki, J. (2022). On the benefits and problems related to using Definition of Done—a survey study. *arXiv preprint*. https://arxiv.org/abs/2208.04003.

Mienye, I. D., & Swart, T. G. (2024). *A comprehensive review of deep learning: Architectures, recent advances, and applications*. Information, 15(12), 755. https://doi.org/10.3390/info15120755.

Muhammad, L. J., Algehyne, E. A., Usman, S. S., Ahmad, A., Chakraborty, C., & Mohammed, I. A. (2024). Explainable artificial intelligence (XAI) for COVID-19 and other medical applications: A systematic review. *Computational and Structural Biotechnology Journal, 22,* 3968–3987. https://doi.org/10.1016/j.csbj.2024.08.005.

Moroney, L. (2020). Recap *TensorFlow Dev Summit*. Retrieved from https://blog.tensorflow.org/2020/03/recap-of-2020-tensorflow-dev-summit.html.

Nordio, M., Meyer, B., & Oriol, M. (2022). Design by contract for deep learning APIs. *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering (ICSE '22 Companion)*, 229–230. https://doi.org/10.1145/3611643.3616247.

Purnama, M. I. W., Fauziah, F., & Sholihati, I. D. (2022). Scrum framework and greedy algorithm in product backlog wedding application planner (Wepplan) activities. *CESS (Journal of Computer Engineering, System and Science), 7*(1), 209–224. https://doi.org/10.24114/cess.v7i1.30625.

Ramchandani, A., Bhatnagar, A., & Raturi, A. (2022). A survey of TensorFlow in machine learning. *Journal of Physics: Conference Series, 2273*(1), 012008. https://doi.org/10.1088/1742-6596/2273/1/012008.

Sarker, I. H. (2021). Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science, 2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1.

Scrum.org. (2021). *The Scrum Framework Poster*. Retrieved from https://www.scrum.org/resources/scrum-framework-poster.

Verwijs, C., & Russo, D. (2021). A theory of Scrum team effectiveness. *arXiv preprint arXiv:2105.12439.* https://arxiv.org/abs/2105.12439.

Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2021). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology, 12*(2), 1–19. https://doi.org/10.1145/3298981.

Zhou, S. K., Greenspan, H., Shen, D. (2021). *A review of deep learning in medical imaging: Imaging traits, technology trends, case studies with progress highlights, and future promises*. IEEE, 109(5), 820–838. https://doi.org/10.1109/JPROC.2021.3054390.