Classificação de Perfil de Usuário Mobile Usando Machine Learning em Dispositivos Edge: Uma Abordagem On-Device com TensorFlow Lite

Raul Cardoso Batalha*
*Especialização em Inteligência Artificial para a Engenharia de Testes de Software - IARTES
Universidade Federal do Amazonas (UFAM)
Manaus, Brasil
raulbatalh@gmail.com

Dr. José Reginaldo Hughes Carvalho†
†Instituto de Computação
Universidade Federal do Amazonas (UFAM)
Manaus, Brasil
reginaldo@icomp.ufam.edu.br

***Resumo*—A proliferação de smartphones e a crescente preocupação com privacidade de dados impulsionam a demanda por soluções de inteligência artificial que operem localmente nos dispositivos (Edge AI). Este trabalho apresenta um sistema completo de classificação de perfil de usuário mobile baseado em padrões de uso de aplicativos, implementado inteiramente on-device utilizando TensorFlow Lite. O sistema coleta dados através da API UsageStatsManager do Android e classifica usuários em cinco perfis distintos: Content Consumer, Social Butterfly, Gamer, Productivity Focused e Mixed User. Foram avaliados seis algoritmos de Machine Learning, sendo que a Regressão Logística obteve a melhor acurácia de 97,5%. O modelo neural convertido para TFLite alcançou 96% de acurácia com apenas 8,8 KB de tamanho, demonstrando viabilidade para deployment em dispositivos móveis com recursos limitados. Os resultados indicam que é possível criar sistemas de perfilamento de usuário eficientes, preservando completamente a privacidade dos dados.**

***Index Terms*—Edge AI, Machine Learning, Mobile Computing, User Profiling, TensorFlow Lite, Privacy-Preserving AI, Android, On-Device Inference**

I. Introdução

O uso de smartphones tornou-se ubíquo na sociedade moderna, com usuários passando em média 4 a 6 horas diárias interagindo com seus dispositivos móveis [1]. Este comportamento gera uma quantidade massiva de dados que pode revelar padrões significativos sobre preferências, hábitos e características dos usuários. A capacidade de identificar e classificar perfis de usuários com base em seus padrões de uso tem aplicações importantes em personalização de experiência, marketing direcionado, detecção de comportamentos problemáticos (como vício em redes sociais) e otimização de recursos do sistema.

Tradicionalmente, sistemas de análise comportamental dependem de processamento em nuvem, o que levanta sérias preocupações sobre privacidade e segurança dos dados pessoais. A transmissão de informações sensíveis sobre uso de aplicativos para servidores externos expõe os usuários a riscos de vazamento de dados e uso indevido de informações pessoais.

Neste contexto, a Edge AI (Inteligência Artificial na Borda) surge como uma solução promissora, permitindo que modelos de Machine Learning sejam executados diretamente nos dispositivos dos usuários, eliminando a necessidade de transmissão de dados para a nuvem [2]. O avanço de frameworks como TensorFlow Lite e a melhoria contínua do hardware móvel tornaram viável a execução de modelos complexos em smartphones com latência mínima e consumo energético aceitável.

Este trabalho apresenta as seguintes contribuições:

- Um sistema completo de classificação de perfil de usuário mobile que opera 100% on-device, garantindo privacidade total dos dados;
- Uma análise comparativa de seis algoritmos de Machine Learning para a tarefa de classificação de perfis;
- Um modelo TensorFlow Lite otimizado com apenas 8,8 KB que alcança 96% de acurácia;
- Uma implementação Android funcional utilizando a API UsageStatsManager para coleta de dados de uso.

II. Trabalhos Relacionados

A. Análise de Uso de Smartphones

A análise de padrões de uso de smartphones tem sido objeto de extensiva pesquisa. Li et al. [3] apresentaram uma revisão abrangente sobre métodos de coleta e análise de dados de uso de aplicativos, identificando quatro principais abordagens: surveys, aplicativos de monitoramento, dados de operadoras de rede e dados de lojas de aplicativos.

O projeto LSApp [4] disponibilizou um dataset público contendo 599.635 registros de uso de aplicativos coletados de 292 participantes, demonstrando a viabilidade de estudos em larga escala sobre comportamento mobile.

B. Classificação de Usuários

Trabalhos anteriores demonstraram que é possível diferenciar usuários com base em seus padrões de uso de aplicativos. Welke et al. [5] mostraram que, considerando apenas os 500 aplicativos mais frequentes, 99,67% dos usuários possuem

assinaturas únicas de uso, evidenciando o potencial discriminativo destes dados.

C. Edge AI e On-Device Machine Learning

O deployment de modelos de ML em dispositivos móveis evoluiu significativamente nos últimos anos. O TensorFlow Lite [6] tornou-se o framework padrão para inferência on-device, suportando quantização e otimizações específicas para hardware mobile. Estudos recentes [7] demonstraram que mesmo modelos de linguagem de grande escala (LLMs) podem ser executados em smartphones modernos com desempenho aceitável.

D. Privacidade em Sistemas de Perfilamento

A preocupação com privacidade em sistemas de perfilamento de usuários motivou o desenvolvimento de técnicas de aprendizado federado e processamento local [8]. Nossa abordagem alinha-se com esta tendência ao processar todos os dados exclusivamente no dispositivo do usuário.

III. Metodologia

A. Arquitetura do Sistema

O sistema proposto segue uma arquitetura de três camadas, conforme ilustrado na Figura 1:

1) **Camada de Coleta**: Utiliza a API UsageStatsManager do Android para coletar estatísticas de uso de aplicativos;
2) **Camada de Processamento**: Realiza engenharia de features e normalização dos dados;
3) **Camada de Inferência**: Executa o modelo TFLite para classificação do perfil.

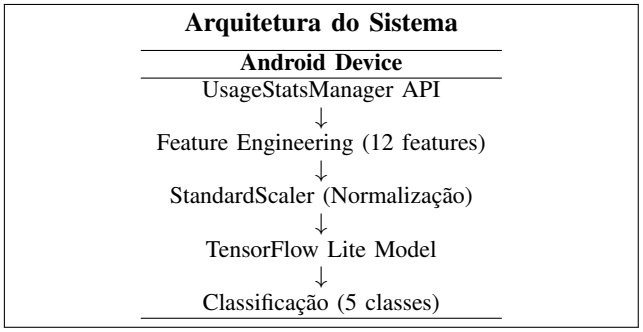

Figura 1: Arquitetura do sistema de classificação de perfil on-device.

A Figura 2 apresenta a distribuição dos perfis no dataset.

B. Dataset

Devido à escassez de datasets públicos com categorização detalhada de uso por tipo de aplicativo (YouTube, redes sociais, jogos, produtividade), foi gerado um dataset sintético baseado nas distribuições estatísticas reportadas em estudos anteriores [9], [10].

O dataset contém 1.000 registros com 21 atributos, incluindo métricas de uso diário por categoria de aplicativo, tempo de tela, uso noturno e características demográficas. A Tabela I apresenta as principais features utilizadas.

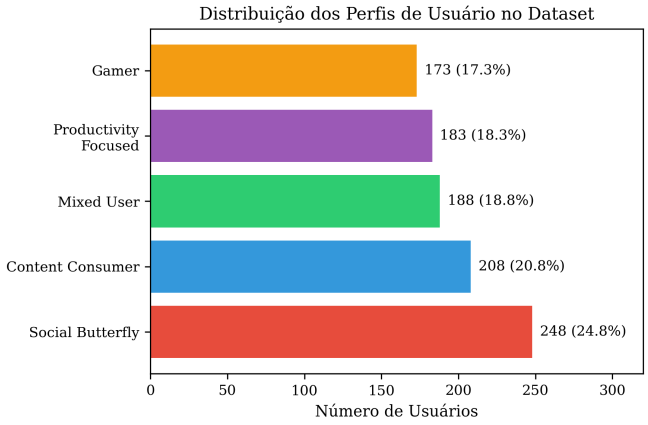

Figura 2: Distribuição dos perfis de usuário no dataset.

Tabela I: Features do Dataset

Feature	Tipo	Unidade
youtube_mins_daily	Numérico	minutos
social_media_mins_daily	Numérico	minutos
gaming_mins_daily	Numérico	minutos
productivity_mins_daily	Numérico	minutos
streaming_mins_daily	Numérico	minutos
total_app_usage_mins	Numérico	minutos
screen_on_hours	Numérico	horas
night_usage_pct	Numérico	percentual
app_switches_per_hour	Numérico	contagem
avg_session_duration_mins	Numérico	minutos
num_sessions_daily	Numérico	contagem
age	Numérico	anos

C. Perfis de Usuário

Foram definidos cinco perfis de usuário baseados em padrões de uso predominantes:

- **Content Consumer**: Alto consumo de YouTube e serviços de streaming;
- **Social Butterfly**: Uso intensivo de redes sociais (Instagram, TikTok, Twitter);
- **Gamer**: Foco predominante em jogos mobile;
- **Productivity Focused**: Uso concentrado em aplicativos de trabalho;
- **Mixed User**: Distribuição equilibrada entre categorias.

A distribuição dos perfis no dataset foi balanceada para evitar viés de classe, conforme mostrado na Tabela II.

Tabela II: Distribuição dos Perfis no Dataset

Perfil	Quantidade	Percentual
Social Butterfly	248	24,8%
Content Consumer	208	20,8%
Mixed User	188	18,8%
Productivity Focused	183	18,3%
Gamer	173	17,3%

D. Pré-processamento

Os dados foram pré-processados utilizando as seguintes etapas:

1) **Normalização**: Aplicação de StandardScaler para normalização z-score:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

onde μ é a média e σ é o desvio padrão de cada feature.
2) **Divisão**: Split estratificado em 80% treino e 20% teste.
3) **Codificação**: LabelEncoder para conversão das classes em valores numéricos.

E. Modelos de Machine Learning

Foram avaliados seis algoritmos de classificação:

- **Logistic Regression**: Modelo linear com regularização L2;
- **K-Nearest Neighbors (KNN)**: Com k=5 vizinhos;
- **Support Vector Machine (SVM)**: Kernel RBF;
- **Random Forest**: 100 árvores, profundidade máxima 10;
- **Gradient Boosting**: 100 estimadores, profundidade 5;
- **Multi-Layer Perceptron (MLP)**: Arquitetura (64, 32), ReLU.

Adicionalmente, foi realizada otimização de hiperparâmetros via Grid Search para XGBoost.

F. Conversão para TensorFlow Lite

Para deployment mobile, foi treinada uma rede neural em Keras com a seguinte arquitetura:

- Input Layer: 12 features
- Dense Layer: 64 neurônios, ReLU, Dropout 0.2
- Dense Layer: 32 neurônios, ReLU, Dropout 0.2
- Output Layer: 5 neurônios, Softmax

O modelo foi convertido para TFLite com otimização de quantização FP16:

Listing 1: Conversão para TFLite

```
converter = tf.lite.TFLiteConverter.
    from_keras_model(model)
converter.optimizations = [tf.lite.Optimize.
    DEFAULT]
converter.target_spec.supported_types = [tf.
    float16]
tflite_model = converter.convert()
```

IV. Implementação Android

A. Coleta de Dados via UsageStatsManager

A coleta de dados de uso é realizada através da API UsageStatsManager do Android, que fornece acesso a estatísticas detalhadas de uso de aplicativos. A implementação requer a permissão especial `PACKAGE_USAGE_STATS`, que deve ser concedida manualmente pelo usuário nas configurações do sistema.

O sistema coleta os seguintes dados:

- Tempo total de uso por aplicativo
- Eventos de foreground/background
- Timestamps de uso
- Número de sessões

B. Categorização de Aplicativos

Foi implementado um sistema de categorização que mapeia package names para categorias de uso. O mapeamento inclui mais de 50 aplicativos populares, com fallback para a categoria reportada pelo Google Play Store via `ApplicationInfo.category`.

Listing 2: Categorização de Apps

```
fun getAppCategory(packageName: String):
    AppCategory {
    appCategoryMap[packageName]?.let { return
        it }

    val appInfo = packageManager.
        getApplicationInfo(packageName, 0)
    return when (appInfo.category) {
        ApplicationInfo.CATEGORY_GAME ->
            AppCategory.GAMING
        ApplicationInfo.CATEGORY_SOCIAL ->
            AppCategory.SOCIAL_MEDIA
        else -> AppCategory.OTHER
    }
}
```

C. Inferência TFLite

A inferência é realizada utilizando o interpretador TFLite com as seguintes etapas:

1) Carregamento do modelo da pasta assets
2) Normalização das features usando parâmetros pré-computados
3) Preparação do buffer de entrada
4) Execução da inferência
5) Processamento das probabilidades de saída

V. Resultados

A. Comparação de Modelos

A Tabela III apresenta os resultados da avaliação dos seis modelos de Machine Learning testados.

Tabela III: Comparação de Modelos de Machine Learning

Modelo	Accuracy	CV Score	Tempo
Logistic Regression	**97,50%**	97,25%	0,04s
SVM (RBF)	97,00%	97,62%	0,05s
K-Nearest Neighbors	96,50%	96,00%	0,02s
XGBoost (Otimizado)	96,00%	96,50%	2,10s
Random Forest	96,00%	96,25%	1,12s
Neural Network (MLP)	95,50%	97,25%	2,82s

A Regressão Logística obteve o melhor desempenho com 97,50% de acurácia, seguida pelo SVM com kernel RBF (97,00%). É notável que modelos mais simples superaram os mais complexos nesta tarefa, possivelmente devido à boa separabilidade linear das classes no espaço de features. A Figura 3 ilustra visualmente esta comparação.

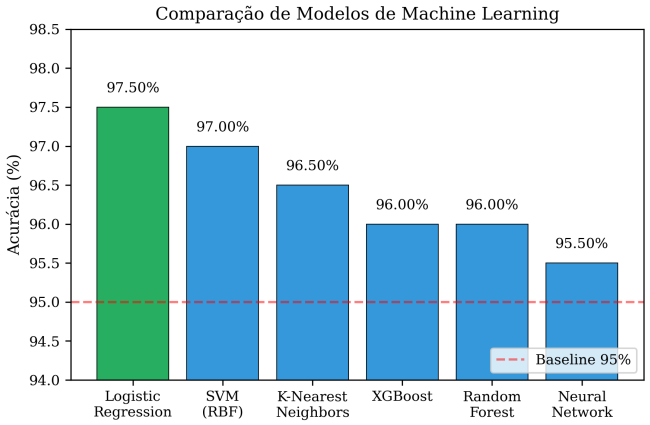

Figura 3: Comparação de acurácia entre os modelos de ML testados.

Tabela IV: Relatório de Classificação - Regressão Logística

Perfil	Precision	Recall	F1-Score	Support
Content Consumer	0,98	0,98	0,98	42
Gamer	1,00	1,00	1,00	35
Mixed User	0,97	0,92	0,94	37
Productivity Focused	0,97	1,00	0,99	37
Social Butterfly	0,96	0,98	0,97	49
Accuracy			**0,97**	200
Macro Avg	0,98	0,97	0,98	200

B. Análise Detalhada do Melhor Modelo

A Tabela IV apresenta o relatório de classificação detalhado para a Regressão Logística.

O perfil "Gamer" apresentou classificação perfeita (100%), provavelmente devido aos padrões de uso muito distintos (alto tempo em jogos, uso noturno elevado). O perfil "Mixed User" teve o menor recall (92%), o que é esperado dado sua natureza de uso equilibrado que pode se sobrepor a outras classes. A Figura 4 apresenta a matriz de confusão completa.

C. Importância das Features

A análise de importância das features (Figura 5) revelou que as features mais discriminativas são:

1) social_media_mins_daily (18,0%)
2) productivity_mins_daily (16,8%)
3) gaming_mins_daily (14,7%)
4) youtube_mins_daily (12,4%)
5) app_switches_per_hour (10,2%)

A idade do usuário apresentou baixa importância (0,9%), indicando que os padrões de uso são mais determinantes que fatores demográficos.

D. Modelo TensorFlow Lite

O modelo neural convertido para TFLite apresentou as seguintes características:

O tamanho extremamente compacto (8,8 KB) permite inclusão do modelo mesmo em aplicativos com restrições severas

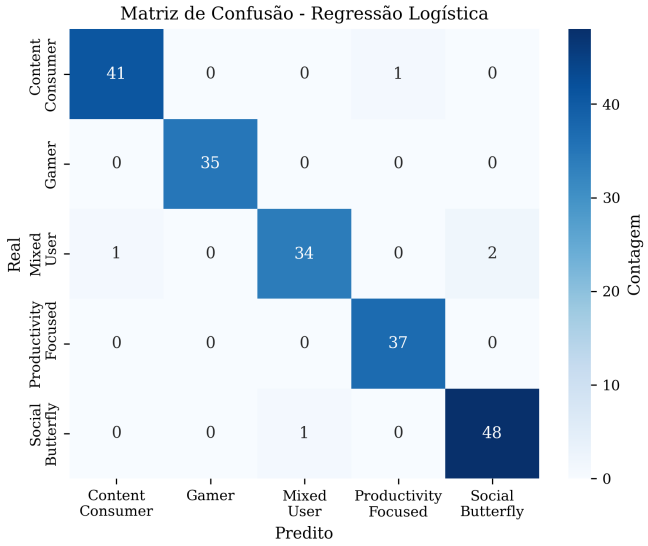

Figura 4: Matriz de confusão do modelo de Regressão Logística.

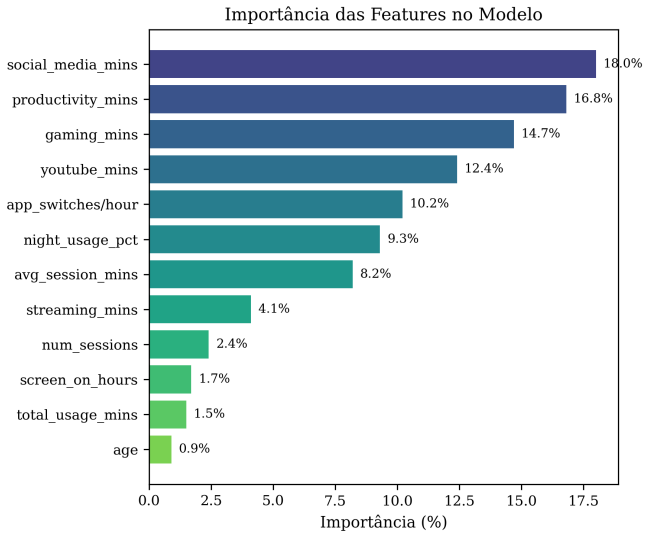

Figura 5: Importância relativa das features para a classificação.

Tabela V: Especificações do Modelo TFLite

Especificação	Valor
Tamanho do modelo	8,8 KB
Acurácia	96,0%
Quantização	FP16
Latência estimada	< 10 ms
Formato de entrada	Float32[1, 12]
Formato de saída	Float32[1, 5]

de tamanho, enquanto a latência sub-10ms garante experiência responsiva para o usuário.

E. Análise de Correlações

A análise exploratória revelou correlações significativas entre as features:

- Total de uso ↔ Tempo de tela: +0,985
- Gaming ↔ Uso noturno: +0,652
- Productivity ↔ Uso noturno: -0,570
- YouTube ↔ Consumo de dados: +0,886

Estas correlações validam a coerência dos dados sintéticos e sugerem que gamers tendem a jogar mais à noite, enquanto usuários focados em produtividade evitam uso noturno. A Figura 6 apresenta o tempo médio de uso por categoria para cada perfil, enquanto a Figura 7 mostra a matriz de correlação entre as principais features.

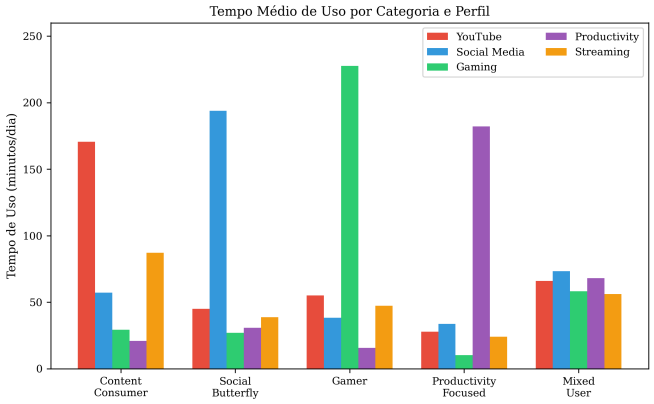

Figura 6: Tempo médio de uso por categoria de aplicativo e perfil de usuário.

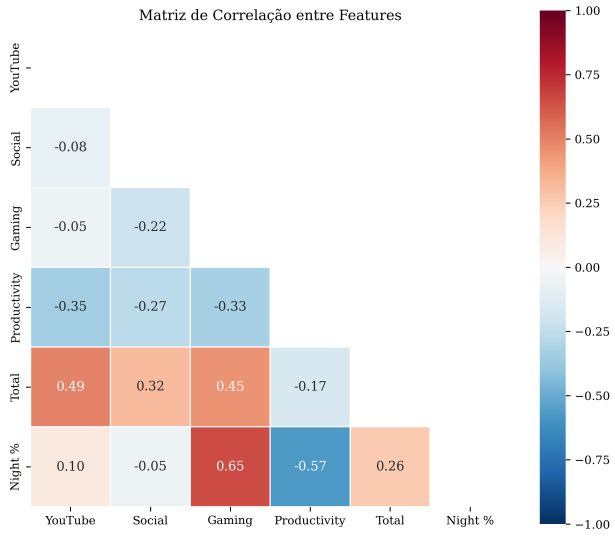

Figura 7: Matriz de correlação entre as principais features do dataset.

VI. Discussão

A. Vantagens da Abordagem On-Device

A implementação on-device apresenta vantagens significativas:

- **Privacidade**: Dados de uso nunca deixam o dispositivo;
- **Latência**: Inferência em milissegundos sem dependência de rede;
- **Disponibilidade**: Funciona completamente offline;
- **Custo**: Elimina custos de infraestrutura de servidor.

B. Limitações

O estudo apresenta algumas limitações:

- **Dataset sintético**: Embora baseado em distribuições reais, pode não capturar toda a variabilidade de usuários reais;
- **Categorização manual**: O mapeamento de apps para categorias requer manutenção contínua;
- **Generalização**: O modelo foi treinado principalmente com apps ocidentais, podendo ter menor acurácia em outros mercados.

C. Aplicações Práticas

O sistema proposto pode ser aplicado em diversos cenários:

- **Digital Wellbeing**: Identificação de padrões potencialmente problemáticos de uso;
- **Personalização**: Adaptação de interface e recomendações baseadas no perfil;
- **Parental Control**: Monitoramento do perfil de uso de menores;
- **Pesquisa**: Coleta anônima de dados para estudos comportamentais.

VII. Conclusão

Este trabalho apresentou um sistema completo de classificação de perfil de usuário mobile implementado inteiramente on-device, demonstrando a viabilidade de aplicações de Machine Learning que preservam a privacidade do usuário.

Os principais resultados incluem:

- Acurácia de 97,5% na classificação de cinco perfis de uso;
- Modelo TFLite de apenas 8,8 KB com 96% de acurácia;
- Implementação Android funcional usando UsageStatsManager;
- Análise comparativa de seis algoritmos de ML.

Como trabalhos futuros, pretendemos:

- Validar o modelo com dados reais de usuários;
- Implementar aprendizado federado para melhorias colaborativas;
- Expandir o conjunto de perfis para capturar nuances comportamentais;
- Investigar técnicas de concept drift para adaptação temporal.

O código-fonte e os datasets estão disponíveis em: https://github.com/raulbatalha/user-profile-classifier

AGRADECIMENTOS

Agradecer à Universidade Federal do Amazonas (UFAM) e o Coordenador Dr. José Reginaldo Hughes Carvalho do curso de Especialização em Inteligência Artificial para a Engenharia de Testes de Software - IARTES.

REFERÊNCIAS

[1] Backlinko, "Screen Time Statistics 2025," Available: https://backlinko.com/screen-time-statistics, 2025.

[2] SiliconFlow, "Best Small LLMs for Edge Devices in 2025," Available: https://www.siliconflow.com/articles/en/best-small-llms-for-edge-devices, 2025.

[3] Y. Li, et al., "Smartphone App Usage Analysis: Datasets, Methods, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1109-1138, 2022.

[4] M. Aliannejadi, et al., "LSApp: Large Dataset of Sequential Mobile App Usage," in *Proc. ACM SIGIR*, 2018.

[5] P. Welke, et al., "Differentiating Smartphone Users by App Usage," in *Proc. UbiComp*, 2016.

[6] Google, "TensorFlow Lite," Available: https://www.tensorflow.org/lite, 2024.

[7] Medium, "Edge LLM Deployment in 2025: Running AI on Mobile & IoT Devices," 2025.

[8] Q. Yang, et al., "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, 2019.

[9] Kaggle, "Mobile Device Usage and User Behavior Dataset," Available: https://www.kaggle.com/datasets/valakhorasani/mobile-device-usage-and-user-behavior-dataset, 2024.

[10] Kaggle, "Smartphone Usage and Behavioral Dataset," Available: https://www.kaggle.com/datasets/bhadramohit/smartphone-usage-and-behavioral-dataset, 2024.

[11] Y. Li, et al., "Smartphone Apps Usage Understanding, Modelling and Prediction," *UbiComp/ISWC Tutorial*, 2019.

[12] D. Yu, et al., "Smartphone App Usage Prediction Using Points of Interest," *Proc. ACM IMWUT*, vol. 1, no. 4, 2018.