Project Proposal: On-Device Personalized Agent using Fine-tuned Qwen3-0.6B

Brandon Howell

1. Introduction

Scope: This project aims to create a personalized AI assistant capable of running directly on an Android smartphone. The core involves fine-tuning a small, large language model (LLM), specifically Qwen3-0.6B[1], with personal user data and other Android domain specific information. Subsequently, this personalized model will be integrated into an experimental Android agent framework (potentially a fork of Gosling[2]) to enable it to perform tasks by interacting with other applications on my device.

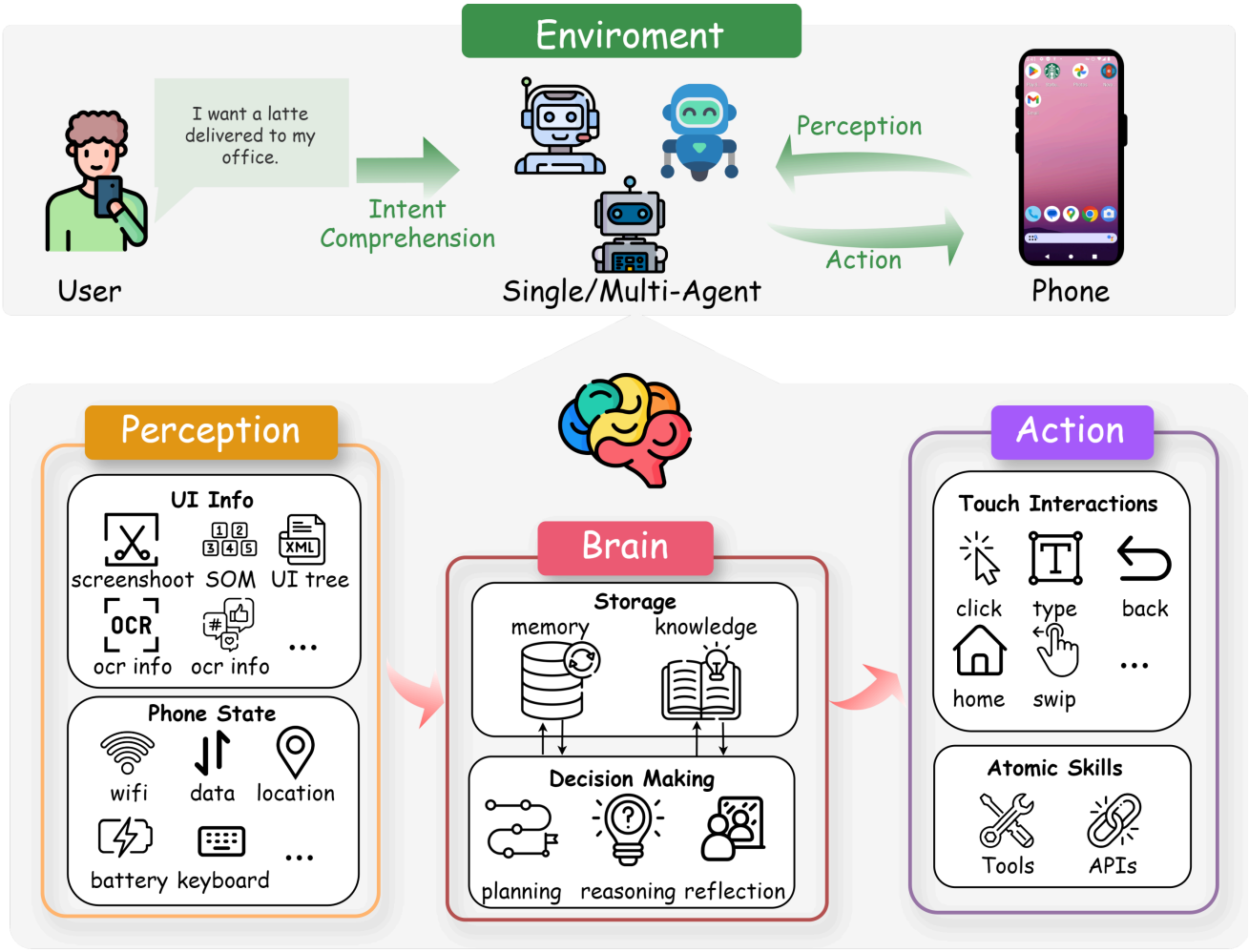
Problem: Current large language models most people are familiar with and use, while powerful, are typically generic and operate in the cloud, which raises privacy concerns and lacks more personalization. Running large models locally on resource-constrained devices like smartphones is computationally challenging. Enabling these models to interact with the device's applications and data in an automated fashion (i.e., act as agents) presents significant technical hurdles in the mobile environment. This project addresses the need for a private, personalized, and capable AI assistant that lives entirely on a user's device.

Interest & Non-Triviality: This project is interesting because it tackles the intersection of several cutting-edge AI domains that I've been following the news of closely for a while now:

- On-Device AI: Deploying capable LLMs on mobile phones pushes the boundaries of model optimization and efficient inference while preserving privacy.
- LLM Personalization: Fine-tuning with personal data explores methods to make AI truly tailored to an individual user's context, habits, and preferences.
- Agentic AI on Mobile: Integrating an LLM with device automation frameworks like Gosling explores the practical challenges and potential of creating autonomous agents in the complex and varied Android ecosystem.

- Parameter-Efficient Fine-Tuning (PEFT): Experimenting with techniques like adapters offers insights into efficient model adaptation for resource-constrained environments.

The combination of fine-tuning a very recent small LLM, deploying it locally, integrating personal context, and building agentic capabilities makes this project ambitious and non-trivial, offering ample room for experimentation and learning.



[3]

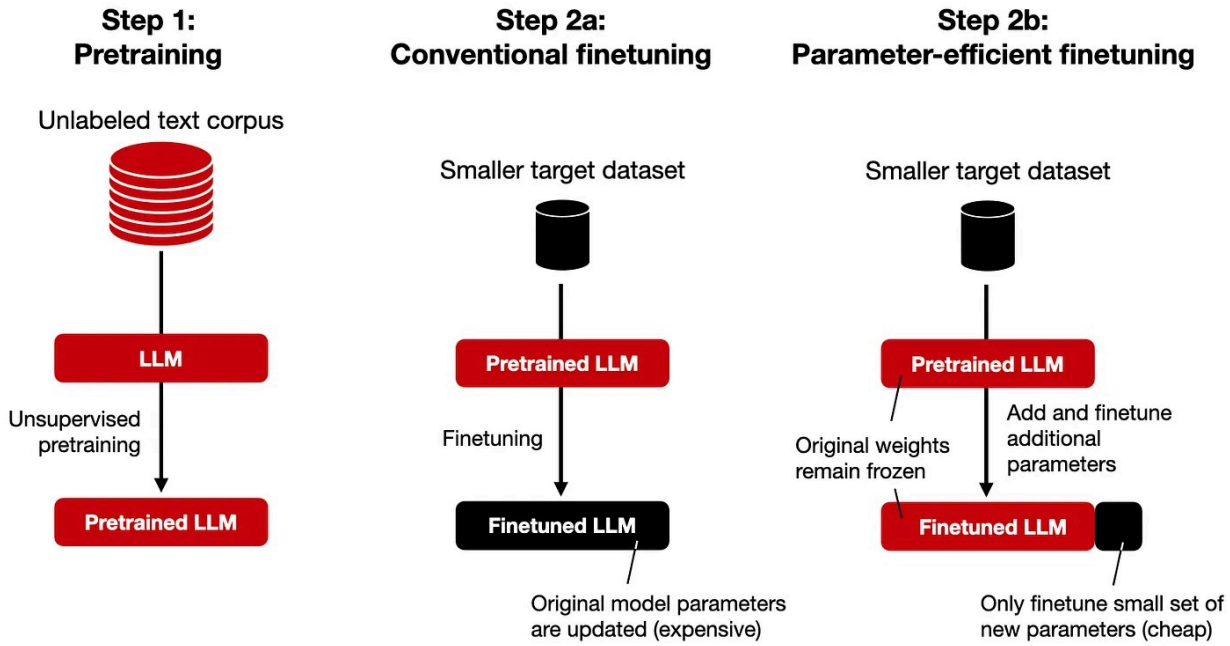
2. Related Work

The development of personalized, on-device AI agents draws upon research in several areas:

- Small Language Models (SLMs): The trend towards smaller, yet capable, language models is necessary for on-device deployment. Models like Phi-3 (Microsoft), Gemma (Google), and the chosen Qwen3-0.6B (Alibaba) represent efforts to achieve high

performance with significantly fewer parameters than models like GPT-3/4. Their smaller size makes them candidates for mobile inference. The Qwen3 series specifically notes optimization for resource-constrained environments[4].

- On-Device LLM Inference: Techniques for efficiently running LLMs on mobile devices are critical. This includes model quantization (reducing the precision of model weights), optimized inference engines (like `llama.cpp`, MediaPipe LLM Inference, ONNX Runtime), and leveraging mobile NPUs/GPUs[5].
- LLM Fine-Tuning for Personalization: Adapting pre-trained LLMs to specific domains or user data is a common practice. While full fine-tuning modifies all parameters, **Parameter-Efficient Fine-Tuning (PEFT)** methods have gained prominence. Techniques like Adapter Tuning (Houlsby et al., 2019[6]), LoRA (Hu et al., 2021[7]), and QLoRA (Dettmers et al., 2023[8]) allow adaptation by training only a small number of additional parameters. This is particularly relevant for on-device scenarios, reducing computational cost and storage requirements for personalized models.



[9]

- LLM-Powered Agents: Frameworks enabling LLMs to use tools, plan, and execute tasks have emerged (e.g., LangChain, AutoGPT). Applying these concepts to mobile environments involves interacting with apps and system services. Projects like Gosling from Block[2:1] specifically explore agentic capabilities on Android, using platform features like Accessibility Services or custom inter-app communication protocols (like their proposed "mobile MCP") to automate tasks. This relates to broader research on

autonomous agents and tool augmentation for LLMs.

3. Method

This project will be developed in two main phases:

Phase 1: Personalized On-Device LLM Fine-tuning

1. Model Selection: The core model will be Qwen3-0.6B, chosen for its small size and state-of-the-art performance within its parameter class[1:1].
2. Data Collection & Preparation: A personal dataset will be curated, which may include (but is not limited to):
   - Calendar entries (structure, common events, timings).
   - Anonymized or synthesized communication logs (SMS/email snippets reflecting personal style, common contacts).
   - Personal notes or documents (FAQs about oneself, preferences, routines).
   - App usage patterns (e.g., frequently used apps for specific tasks).
   - Location habits (e.g., common places like "home", "work", "gym").
   - The format will likely be structured prompts and desired responses (instruction fine-tuning format).
3. Fine-tuning Implementation: Two fine-tuning approaches will be explored using libraries like Hugging Face `transformers` and `peft`:
   - Full Fine-tuning: Update all weights of the Qwen3-0.6B model on the curated personal dataset.
   - Adapter-based Fine-tuning (PEFT): Freeze the base Qwen3-0.6B model and train lightweight adapter modules[6:1][7:1][8:1].
4. On-Device Deployment: The fine-tuned models (both versions) will be prepared for on-device execution. This will likely involve:
   - Quantization (e.g., to 4-bit using GGUF format).
   - Using an on-device inference engine compatible with Qwen models and Android, such as `llama.cpp`'s Android bindings or potentially adapting other frameworks if necessary.
   - Building a simple Android application wrapper to load the model and allow text-based interaction for initial testing.

Phase 2: Agentic Integration with Gosling

1. Framework Setup: Fork the Gosling Android agent repository. Familiarize with its architecture, particularly how it invokes LLMs and interacts with device capabilities (Accessibility Services, Intents, potential "mobile MCP").
2. Model Integration: Modify the forked Gosling code to use the personalized, locally deployed Qwen3-0.6B model (likely the PEFT version due to efficiency) as its reasoning engine. This will involve adapting the LLM API calls within Gosling to interface with the chosen on-device inference engine.
3. Tool Definition & Use: Explore Gosling's mechanisms for defining and using tools. This might involve:
    - Leveraging existing Gosling capabilities triggered via natural language prompts.
    - Potentially implementing a simple custom "mobile MCP" provider app (as shown in the Gosling README example) to expose a new capability (e.g., retrieving a piece of personal info directly).
4. Task Automation: Define and test simple, multi-step tasks that require the agent to use the personalized LLM and interact with other apps via Gosling's capabilities.

4. Evaluation

Success will be evaluated based on experimentation depth and insights gained, rather than solely on flawless execution.

Phase 1 Evaluation (Personalized LLM):

- Dataset: A held-out set of prompts based on the personal data domain (e.g., questions about schedule, preferences, communication style).
- Metrics:
    - Qualitative Assessment: Subjective evaluation of the model's responses for personalization (does it know my context?), accuracy (correct information), coherence, and tone compared to the base Qwen3 model.
    - Quantitative Assessment (Exploratory):
        - Perplexity on the held-out personal dataset.
        - Resource Usage: Measure model storage size (base vs. full fine-tune vs. base + adapter), inference latency, and peak memory usage on the Android device for both fine-tuning approaches.
    - Comparison: Directly compare the effectiveness and resource trade-offs between the fully fine-tuned model and the adapter-based model.

Phase 2 Evaluation (Agentic Capabilities):

- Dataset: A set of defined, multi-step tasks requiring interaction with device features or apps.
- Metrics:
  - Task Completion Rate: Percentage of tasks successfully completed by the agent.
  - Qualitative Assessment: Evaluate the agent's planning ability, tool usage effectiveness, error handling, and overall usefulness for the defined tasks. Does the personalization from Phase 1 demonstrably improve task execution (e.g., using correct contact names, understanding implicit context)?
  - Efficiency (Observational): Note the time taken and number of steps/interactions required for task completion.

5. Milestones

- Week 6: Download Qwen model and test its current capabilities, begin making fine tuning dataset
- Week 7: Fine tune and test new model, evaulate success, where new model succeeds that old model didn't
- Week 8: Assuming above has gone well, fork Gosling and begin working on Agent integration
- Week 9: Combine fine-tuned Qwen3 with Gosling to automate tasks, test and get quantitative results of success
- Week 10: Wrap up any last features, test, and evaluations and finish report

6. Task Assignments

I'm working solo for this project since there was some miscommunication with groups, so don't have to worry about splitting up tasks.

---

1. https://huggingface.co/Qwen/Qwen3-0.6B ↩ ↩
2. https://github.com/block/gosling ↩ ↩
3. https://www.preprints.org/frontend/picture/ms_xml/manuscript/

3da31779a6c7c196419c841cee5292cf/preprints-145259-g005.png ↩
4. https://qwenlm.github.io/blog/qwen3/ ↩
5. https://arxiv.org/abs/2312.03863 ↩
6. https://arxiv.org/abs/1902.00751 ↩ ↩
7. https://arxiv.org/abs/2106.09685 ↩ ↩
8. https://arxiv.org/abs/2305.14314 ↩ ↩
9. https://magazine.sebastianraschka.com/p/finetuning-llms-with-adapters ↩