CMPE 492

Performance Evaluation Of LLM Services Over Multi-Tier Edge Networks

Ozan Oytun Karakaya

Advisor:

Cem Ersoy

1/10/2025

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Broad Impact

Rapid advancements in the field of Artificial Intelligence (AI) leads to newer technologies adopted in various industries and replacements of traditional service providing with a wide range of AI Applications powered by Large Language Models (LLM), Deep Neural Networks (DNN), etc. There are lots of studies being conducted about replacing or enhancing traditional services with the advanced LLMs in the year of this document, 2024. Motivation behind these advancements is the fact that capabilities of LLMs such as generating human-like texts, performing complex tasks and processing vast amounts of data are to be proven in the recent years with services most people use in their daily lives. On the other hand, this situation comes with increasing demand to these tools and Quality of Service (QoS) requirements respectively. LLM tools handle vast amount of data in order to meet certain criteria when providing services to its users [5]. During these processes, these services are also required to meet the latency criteria since LLM services are deployed on various industries concurrent with other technologies such as Augmented Reality (AR), business-specific use-case technologies where low-latency (ms) is an important criterion. As a result to this problem, researches considering the criterion of meeting low latency criterion leads contributors to find ways to deploy LLMs efficiently to the networks used. Moreover, LLMs that are utilized in IoT industries should also be compatible with the environments they are deployed on due to low resource availability, low network bandwidth, etc.

Following the problem mentioned in the above paragraph leads researchers, industries to solving the problem of various deployment schema of LLMs over the network. Edge Computing as one of the most prominent solutions to this problem can be utilized for LLM Deployment under various conditions. Since Edge Computing is an promising solution to the problem of optimizing computing resource utilization under latency constraints [1], it can resolve the problems that LLM services face due to high demand and QoS constraints. In this study, by displaying the advantages of using edge network

architectures during LLM deployment using simulations, it is targeted to pave the way for new researches to be done about Edge Network Deployments for LLMs so that higher utilization of LLM services for every both user and development budget can be achieved.

1.2. Ethical Considerations

In order to run the simulations as similar as to real world LLM applications, similar range of data sizes should have been taken into account with the similar LLM services. All of the data-sets will be utilized under the research are chosen publicly available and will be shared with the readers in order to avoid data right violations. Again LLM services studied for input/output network load generation are chosen among ones which have publicly available sources.

Since this study will display its results only using simulations for creating reference points for future advancements, it does not rely on any real-world system that is currently on use, thus, raising least amount of ethical concerns about the study. However, it is important to note that for future researches using real-world applications and devices based on this study, data usage throughout the study should be studied in detail with sufficient protection measures. Reason for taking such measures is the fact that using edge computing paradigms for LLM training and inferring requires data transfer over the deployed networks which may cause the used data to be vulnerable on the network.

2. PROJECT DEFINITION AND PLANNING

2.1. Project Definition

Under this study, main aim specified is displaying performance advantages gained such as lowering end-to-end latency caused by network load with the help of edge computing architecture for various LLM services and loads generated by those services. It is important to note that under this study, LLM training phase will not be considered since LLM usage generally covers the inference service where users make requests and get responses of the inference of LLM applications.

For current deployment schema, traditional cloud services are used for LLM deployment where a single, highly computational capable service endpoint is utilized for providing responses to user requests. There are few problems arise with this architecture explained below.

- For latency sensitive services, due to its architecture, traditional cloud deployment schema have a tendency to go through overloading problems for both computational and network wise since whole operation is run on a single endpoint, thus, high task failure rate eventually.
- This architecture generates the single point of failure problem that can cause poor service quality for services that interruptions are not tolerable such as healthcare, live-monitoring, etc.
- Aligned with the previous item, back-up services for such systems should be designed which can eventually lead to more cost for developers. Even though cost constraints are not studied under this study, this problem may arise during real world applications.

In order to overcome problems mentioned above, and other similar problems, edge computing architecture is suggested under this study where certain service types with the loads they generate in both user and service side is simulated and compared with

single-tier architecture. In order to have a solid comparison, most tolerable latency will be used for success/failure criteria and task failure rates will be compared with both architectures.

2.2. Project Planning

2.2.1. Project Time and Resource Estimation

It is estimated that this study requires 4 months of time approximately to be completed. The timeline estimation includes following phases.

(i) **Research (4 weeks):** As the beginning part to the study, various researches and studies is investigated to gain further knowledge about current industry implementations, problems arising along with them, etc. This phase also includes gaining insights about edge computing implementations' advantages for various tasks other than LLMs such as AR, IoT applications.

(ii) **Simulation Tool Study (2 weeks):** EdgeCloudSim as the simulation tool for edge computing will be utilized under this study (will be explained below in detail). During this 2 week, capabilities of EdgeCloudSim will be investigated for keeping the study as similar as to the real-world applications.

(iii) **System Design (4 weeks):** Under this phase of the study, various scenarios will be designed in order to be tested with edge architecture LLM deployment against single-tier cloud deployment with prepared estimations of request/response loads studied.

(iv) **Simulation Run (4 weeks):** Designed experiments will be run in this phase of the study. Since there may be various problems can arise due to simulation program, its compatibility, estimated run-times and so on, this phase is kept longer than it is expected to be.

(v) **Result Gathering (2 weeks):** Under this phase of the study, all of the results gathered after all the work done will be collected and prepared in a format that is ready to present such as plots, graphics, etc. This phase also includes reporting phase of the study.

2.2.2. Success Criteria

Since this study mainly focused on gaining insights about edge computing architectures advantages for LLM deployment, it does not have a specific success criteria as it should have been in a project where a certain product will be implemented and tested against its competitors in the field. However, this study is conducted with a strong insight that emphasizing edge computing should be more suitable for meeting latency constraints compared to single-tier deployment due to fact that edge computing, as its own structure requires, has a distributed system paradigm which enables to have a more tended mechanism to respond to request with shorter network delays.

In order to prove the point mentioned above, task failure rates will be compared after the simulation runs for both system designs. Since all service requests will have a maximum amount of tolerable service time for successfully completion in end-to-end request-response schema, task failure here stands for the job completions that are failed to stay under this maximum tolerable service time. It is foreseen in the beginning of this study that multi-tier edge architecture will display a lower task-failure rate (or higher task-completion rate) compared to single-tier architecture which actually forms the motivation for this study.

2.2.3. Risk Analysis

All of the data will be used for the study are chosen among publicly available data-sets which prevents any risks about data usage. Moreover, since the study is based on simulation runs, it does not present any risks considering user scenarios and LLM applications. Although, the study does not project any risks considering outside of the study, it contains risks about deviated results as outputs of the simulations since the simulation tool used for this study are not created for simulating LLM loads, which leads to a important phase of the study that consists of transferring real-world alike loads to the simulation. All of possible deviation in the result will be tested and minimized during the system design and simulation runs.

3. RELATED WORK

During the research phase of this study, many studies considering both edge computing and LLM workloads are investigated. Some of these studies not only focuses on managing edge resources for AI workloads, but also accomplishing this task by using a model-driven approach as in Liang et al. suggested [4]. By using this model-driven approach, higher utilization on service utilization can be achieved. It is also studied as two different approaches for AI and Edge in the format of AI for Edge & AI on Edge as Deng et al. suggested in [2]. Moreover, edge computing paradigm's beneficial features are displayed widely [1]. It is also considered that benchmarking AI workload performance these studies is a crucial step for analyzing performance gains[3]. Lastly, with huge thanks, a base study for this study conducted in Bogazici University can conclude Edge Computing's benefits over AI workloads with simulation technique using EdgeCloudSim [8].

4. METHODOLOGY

Under this chapter, steps of our study will be covered in the sections below, explaining each of them in detail.

4.1. Research

As this study consists of two parts (Edge Computing and LLM Serving), both parts of the study should be covered under research phase of the project. Since these two technologies are separately researched for various applications and advancements, there are reasonable amount of research completed on these areas. Since LLMs are one of the most trending technologies in the recent industrial practices due to its capabilities, there are lots of communities that conduct researches and developments for LLMs and their serving. As for edge computing, there are several researches should be analyzed in order to proceed to other steps in order to understand its capabilities. Topics below are covered for research purposes.

- Edge Computing Multi-Tier Architecture - Task Offloading
- Real World Application LLM Data-Set Sizes
- System Requirements for LLM Inference on Various Devices
- System & Network Loads Generated By LLM Services
- User QoS Constraints of Real World LLM Applications

By completing researches about the items listed above, creating scenarios similar to real-world applications is aimed. This research phase will also enlighten the system design process for creating a robust, real-world feasible edge computing architecture for simulations run in the study.

4.2. Simulation Tool Study

Due to cost of the hardware devices (or service provider fees) that is required for building a real-world edge computing network for testing, it is decided to use simulation in this study. Although CloudSim is a simulation tool Cloud studies it is not sufficient to run multi-tier edge architecture simulations on it. Thus, EdgeCloudSim is developed and answers to this problem[6]. Since this simulation tool is the base of our simulations for our study, it is highly important to capture its capabilities for our study when simulating deployed LLMs. Creating Edge Computing simulations with sample applications, running & gathering their results is a crucial step of this study since the decision whether the simulation tool will be extended or not will be decided accordingly. Then, LLM applications will be covered with the tool's capabilities.

4.3. System Design

The comparison of single-tier deployment vs. multi-tier edge computing deployment for LLM serving lays down on the base of this study. In order to design such as comparison system, knowledge and capabilities covered in the above section will be utilized to create these two structures. There are important check-marks required to be covered during these phase.

- Applications simulated should be compatible for both single-tier and multi-tier architecture.
- Since latency constraints is the base of the study's success criterion, applications simulated should be latency-sensitive.
- Simulation data should cover real-world scenarios in the aspects of size, complexity, usability.
- Whilst meeting the criteria mentioned in the above item, simulation times should be in a feasible range for study to make progress continuously.

4.4. Simulation Run

To cover whole aspects of the comparison to be made, there would be reasonable amount of runs should be done with necessary configurations and warm-up periods. Moreover, this phase should be executed in an iterative manner since firstly created scenarios might not be feasible under the constrains or reflect the real-world scenarios appropriately. Thus, iteratively changing the configurations of simulations or extending the tool used according to necessities might be required to display appropriate results. Configurations should cover the terms below.

- Configured Cloud or Edge Network should reflect real-world devices' capabilities such as CPU, Memory, Storage resources and run-times.
- Both computation and network loads generated should reflect the wide range of use case scenarios including the congestion cases and the idle cases.

4.5. Result Gathering

Since this study's aim is to display the advantages estimated gained by using a multi-tier edge computing schema, results should be gathered and compiled to be ready to present at the end of the study. Not only the simulation variables, but also plots and graphics should be used to explain the study's findings during the reporting process.

5. REQUIREMENTS SPECIFICATION

Before getting into details of requirement specifications of this study, it is highly important to note that this study's aim is not to create an end product about the subject it covers. It's aim is to make a contribution to LLM serving methods with displaying edge computing's advantages, thus, this section might not cover a specific format of requirements in the aspect of functionality or system requirements. However, requirements expected from the systems to be tested under simulations can be specified.

5.1. Glossary

- **End Device**: Any kind of devices that make request to LLM services in the simulation.
- **System**: Deployed LLM Service, for both cases: single-tier and multi-tier.
- **Task**: Processing of the request in the system side.
- **Task Failure**: Tasks that are not completed in the maximum tolerable service time specified in requests.
- **Task Offloading**: Procedure of processing the request on a different host via request/response schema.
- **Host**: Each one of the Edge Servers in the system.

5.2. End Device Requirements

(i) End Devices shall be able to send their requests to the system and get their responses via network, WLAN or WAN.

(ii) End Devices shall be able to specify their requests features, application, computation load, data, maximum tolerable service time.

(iii) End Devices shall be able to get response to their request whether the task failed or not.

5.3. System Requirements

(i) System shall be able to get requests from end devices and send responses via network, WLAN or WAN.

(ii) System shall be able to process requests done by end devices in reasonable times specified with the sufficient computational resources allocated.

(iii) System shall be able to orchestrate its workflow via offloading tasks between hosts in the multi-tier architecture.

6. DESIGN

6.1. Information Structure

Information through this study has various forms as in the context of simulation data. All information can be separated into two parts: service information and simulation information.

6.1.1. Service Information

Services that will be simulated in this study can be summarized with their four features: Data Load, Task Load, Arrival, Maximum Tolerated Service Time.

(i) **Data Load:** Every service simulated in this study has average values for up-link load (requests to cloud) and down-link load (responses from cloud) over the network. Moreover, data processes within these requests should fit in the storage of the machine processing them.

(ii) **Task Load:** Since this study focuses on LLM services, it is suggested to use Floating Point Operations per Second (FLOPS) unit to describe the tasks and machine capabilities instead of using Million Instructions per Second (MIPS) unit. Task load on hosts are described with FLOPS required for each service along with the how many core required to run necessary inference process. It is important to note that, GPU usage is referenced rather than CPU usage on hosts since this study focuses on LLM tasks. By keeping the CPU utilization out of scope and making the assumption of sufficient CPU resource, simulations can be lighter (more effective) while still meeting the real-world scenarios' criteria which are investigated under this study.

(iii) **Arrival:** Arrival rates of incoming requests for clouds differ from service to service in the simulations. In addition, active/idle states of the LLM services will also be described in the simulations which enables flexibility between scenarios thanks to features of EdgeCloudSim. Lastly, usage percentage of different services will be

described on the hosts for more realistic scenarios, such as paid services.

(iv) **Maximum Tolerated Service Time T(max):** This feature will be implemented by extending the already implemented features of EdgeCloudSim in order to have better reflection of the real-world scenarios based on the assumption that tasks that take longer than this time-period are accepted failed since they will not be utilized in the real-world. For instance, a live health monitoring service that take up to 5 minutes can be assumed useless for emergency services.

6.1.2. Simulation Information

Simulation information is provided for each simulation run. There are three different configuration files for making changes on the simulation environment.

(i) **applications.xml:** This file is used for describing the services that will be run on the simulation. All of the configurations about the service (mentioned in the section above - Service Information) and more is provided to simulation with this file. Service Information part is described in a separate section above since the it contains additional features compared to this configuration file has in default.

(ii) **edge_devices.xml:** This file is used for describing the cloud environment that will be running the services defined in the previous configuration file. In this configuration file, two different types of settings will be tested under this study: single-tier data center with high computational resources and multi-tier edge computing architecture with lower but distributed computational resources.

(iii) **default_config.properties:** In this file, details of the simulation is defined for simulation program such as warm-period, run-time of the simulation, end device number, etc. In order to display the effects of different scenarios with varying numbers, this file settings will be iteratively changed between runs during simulation run phase of the study.

6.2. Information Flow

Information types described in the section above are predefined before each simulation run. After running the simulation, results (logs) of the simulation is recorded. Then, it is proceeded to iterative process for changing simulation environment for different scenarios. Describing the simulation data flow inside the EdgeCloudSim is out of the scope of this study, it is highly suggested to checking out related research of EdgeCloudSim [4]. Instead, flow of the inside simulation data will be inspected in this section for two different deployment schema.

6.2.1. Single-Tier Deployment

In this traditional deployment schema, service deployment obeys to the traditional Master-Servant schema where end devices make requests to cloud server where LLM service is hosted and cloud server responds back with the inference outputs.

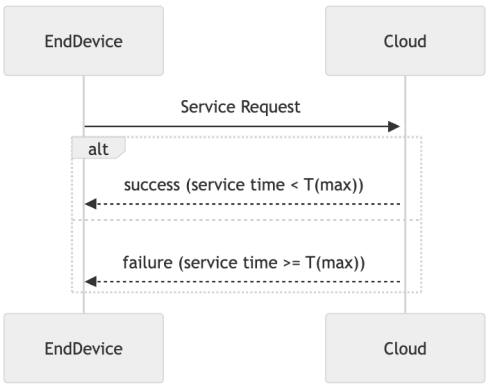

Figure 6.1. Single-Tier Deployment

6.2.2. Multi-Tier Deployment With Edge

In this deployment schema, there is one extra layer of network between end devices and the cloud server. In this schema, cloud server is not responsible of completing all tasks in the system. Edge servers are capable of completing inference tasks requested by end devices, however, it is important to note that edge servers are not as computationally capable as the cloud server. There may be more than one edge server (host),

however, there is always a single cloud server in this architecture. This structure can be enhanced with features such as logging on cloud server or alarming mechanism, etc. In this study's scope, it is assumed at edge servers (hosts) are orchestrated with a built-in default configuration by an edge orchestrator module responsible for load balancing. The proposed advantage of this architecture is the feature of edge computing paradigm which is task offloading. Task offloading is utilized during congestion phases of the system. Moreover, it can also be utilized with different configurations such as offloading heavy tasks (high FLOPS numbered tasks) to computationally rich cloud server compared to edge servers might easily increase QoS or decrease average service times along with task failure rates.

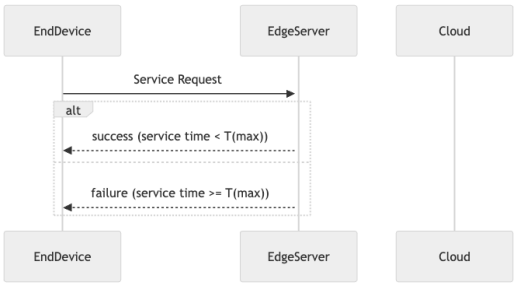

Figure 6.2. Multi-Tier Deployment Without Task Offloading

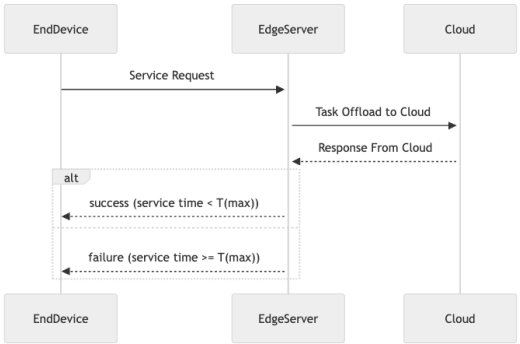

Figure 6.3. Multi-Tier Deployment With Task Offloading

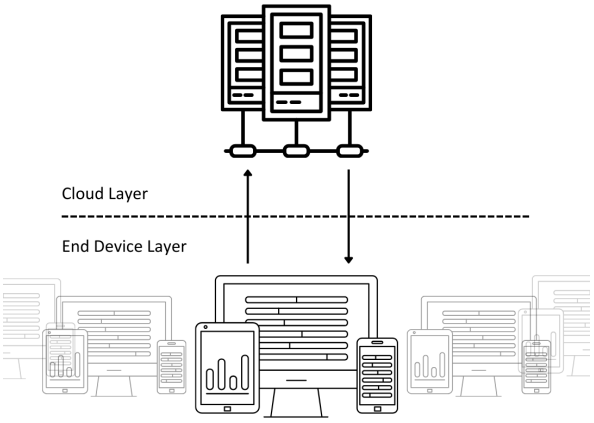

Figure 6.4. Single-Tier Architecture

6.3. System Design

Module diagrams and their responsibilities are widely explained for simulation tool used in the EdgeCloudSim repository[1] . In this section, system designs of the two different architectures will be explained briefly.

6.3.1. Single-Tier

Single-Tier architecture covers the traditional Master-Servant architecture in this study. This deployment schema is widely adopted throughout the years of development of software applications, embedded systems, etc. In the scope of this study, although its lower complexity, this architecture possess the problem of single point of failure and single service queue eventually. Thus, it is discussed under this study in the chapter above that this architecture may not be sufficient to meet low-latency criteria for highly demanded services.

[1] `https://github.com/CagataySonmez/EdgeCloudSim`

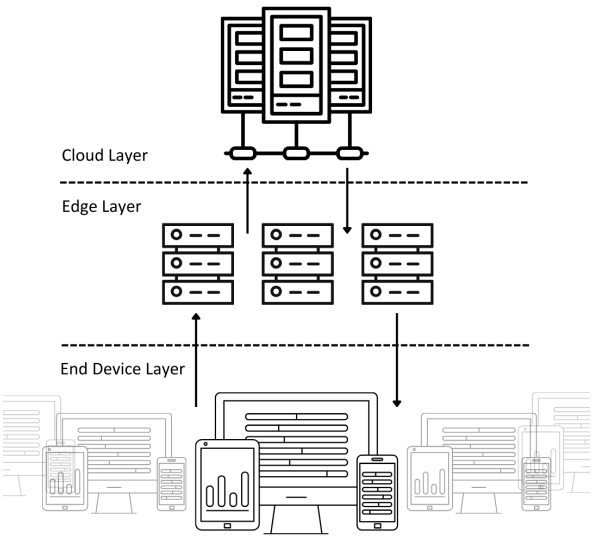

Figure 6.5. Multi-Tier Edge Architecture

6.3.2. Multi-Tier With Edge

Multi-Tier Architecture discussed in the scope of this study states that adding one more layer, might be more than one in other studies, between the single cloud server and various end devices. Aim for adding one more layer between these two layers is to move service deployments closer to the devices that make requests to them so that ensuring low-latency constraints. This architecture is also much more convenient for mobile deployment cases where stationary continuous serving is not possible due to physical conditions, etc. One example can be given as providing a V2V application regulating high-speed highways across mountains. In such a case, deploying the service in single-tier architecture might not help to end devices since WAN can fail to meet service's constraints.

This study's aim is maximizing the throughput under strict time related constraints for various LLM services, thus, it is suggested to move inference tasks closer to devices for achieving higher QoS on the user side of these services by using the edge computing paradigm. Although it yields complexity issues, especially due to fact that LLMs are complex models where keeping accuracy above reasonable thresholds is a complex task while downsizing the model so that it can fit onto edge servers, edge servers are getting stronger computational-wise thanks to advancements in hardware

technologies. Moreover, downsizing LLMs with various newly emerged techniques have fewer effects on accuracies of the models, such as low-bit quantization, KV Caching, etc.

7. IMPLEMENTATION AND TESTING

Before getting into the implementation phase, there are lots of parameters to be run in the simulation. Thus, heuristics & evaluation methods, processed data concerning the parameters of the simulation should be configured before the simulation phase. Moreover, since the study is conducted for experimental purposes to discover the possibilities of different deployment structures for LLM Deployment, implementations cover the parameters and configuration of simulation.

Under this chapter, steps of creating such simulation will be covered in detail in the Implementation section below.

7.1. Implementation

7.1.1. Scenario

In order to study different deployment structures for LLMs, a sample scenario may help readers to visualize and comprehend the context of the simulation. In order to satisfy this purpose, a scenario about academic usage is developed. An LLM service is assumed to be deployed for Bogazici University Computer Engineering students where students can utilize the application with various services it provides. These services are listed below.

- **Coding Helper**: Service that takes pseudo-code inputs and converts them into code pieces in Python language.
- **Summarization Service**: This services is utilized to summarize long texts which students can use to get insights from various resources of information about their courses, research topics, etc.
- **Course Module Service**: This service can be thought as a helper service for university course module where students can organize their course structures, add/drop courses, etc. by providing prompts to this service.

- **Logic Q&A Service**: This service is utilized to have an interactive chat with an assistant that includes a mathematical engine. This service is the heaviest service among all of the services since it includes heavy computations for logical questions.

Students are assumed to use this LLM applications in the campus with their devices, making requests to the application's endpoint. Under the scenario of the simulation, it is studied whether deploying this LLM service in single-tier or two-tier architecture where the cloud consists of more powerful resources compared to edge server. Edge server is assumed to be a single server deployed in the campus where the cloud is located far away from the location of Bogazici University (Istanbul) such as Germany, Europe.

7.1.2. Model Selection

Due to its transparency for research and accessibility, Llama[7], developed by Meta AI, is preferred in the context of this study. In addition, accessing to refined datasets with input/output token lengths is made easy through Hugging Face[2] for Llama models. As for the version of Llama, 7B, with 7 billion parameters, is chosen due its lightweight and effective structure which makes it suitable for the edge server deployment.

7.1.3. Server Configurations

As it is stated before, in the simulation, acceptance criteria is specified with the maximum tolerable service time for service requests. Thus, other parameters such as storage in the servers is excluded. Moreover, since LLM services specified for this simulation study consists of text-only inputs with low token lengths; therefore, keeping their input/output bytes at numbers that can be easily neglected. Under the scope of the study, there are two main parameters that can affect the service type: task length of services, delay for accessing cloud servers.

[2] `https://huggingface.co/`

As for the first one, data and the processing unit for tasks are crucial where NVIDIA A100 GPU[3] is selected to be simulated on the edge servers with its ability. Cloud's processing unit is again based on this model's power multiplied in the size. Reason behind this GPU to be selected in this study is simply this GPU's popularity and accessibility that makes it a solid reference point for other comparisons. In addition, A100's ability to compute 312 Tera Floating Point Operations Per Second (312 TFLOPS) makes it perfect candidate for the edge server deployment.

Latter one is explained in the simulation data section.

7.1.4. Heuristics

According to scenario developed for this simulation study, 4 different data sources are gathered in order to calculate average input/output token lengths of the related services. However, EdgeCloudSim defines the length of the services (named tasks in the application) in Mega Instruction (MI) units and servers in MI per second (MIPS) unit. In order to utilize the realistic data output of the gathered service data in the simulation, few heuristics are used to convert the units which are explained below.

- A thumb-rule concerning word-token relation is utilized. **1 token stands for 0.75 of an average English Word length.** This heuristic is a popular calculation inside LLM community where more information can be accessed from the interview made with a Meta AI engineer[4] .
- Instructions are assumed in 32-bit format. The deployed model is adopted with Low-Bit Quantization to keep it light-weighted so that it works with FP16 units. Thus, **2 FP16 instructions consist 1 32-bit instruction.**
  - $MI\_task\_length = avg\_input\_token \times [7 \times 10^9(parameter\_size)] \times \frac{1}{2 \times 10^6}$
- For Summarization Service, it is assumed that LLM model takes 10 times shorter path due to adopted attention mechanism making the model faster.
- For Course Module Service, it is assumed that LLM model takes 5 times shorter

[3] https://www.nvidia.com/tr-tr/data-center/a100/
[4] https://www.youtube.com/watch?v=Tmdk_H2WDj4&t=90s

Calculated Simulation Parameters	Coding Helper	Summarization Service	Course Module Service	Logic Q&A Service
Average Input Token Length	25	447	10	88
Average Output Token Length	122	27	245	20
Average Input Bytes	133	1970	49	396
Average Output Bytes	663	120	1178	92
Task Length (GI)	**87.5**	**156.45**	**7**	**308**
T-max (seconds)	2	3.5	0.2	1.5
Poisson Inter-Arrivals	20	10	10	20
Cloud Offloading Probability	40%	30%	0	50%

Figure 7.1. Application Parameters for Services

path due to adopted attention mechanism making the model faster.

It is important to note that this heuristics are used only for conversion to simulation parameters from the realistic datasets. Since the study focuses on performance evaluation between different deployment structures, this conversions make this parameters constant variables for both cases, not affecting the results since they are same for both deployment structures.

7.1.5. Service Data Preparation

All of the datasets used in this study are chosen publicly available due to reasons explained in the beginning chapters. Datasets chosen for each service is stated below.

- Coding Helper: CodeSearchNet[5]
- Summarization Service: EdinburghNLP/xsum[6]
- Course Module Service: Bitext - Customer Service Training Dataset[7]
- Logic Q&A Service: LogiQA[8]

Using the average values of tokenized datasets for input and output lengths, service parameters inside the simulation is calculated. The calculated simulation parameters for applications is stated in table in Figure 7.1.

[5] `https://huggingface.co/datasets/code-search-net/code_search_net`
[6] `https://huggingface.co/datasets/EdinburghNLP/xsum`
[7] `https://huggingface.co/datasets/bitext/Bitext-customer-support-llm-chatbot-training-dataset`
[8] `https://huggingface.co/datasets/lucasmccabe/logiqa`

It is important to note that the parameter "Cloud Offloading Probability" is only valid for the two-tier simulation runs. In single-tier simulation runs, all of the tasks are executed in the cloud server, thus, this parameter is set to %100 for all services in the single-tier scenario.

7.1.6. Simulation Parameters

EdgeCloudSim parameters are not suitable for the configuration running both the single-tier and the two-tier deployment structure in parallel for the same application parameters. Single-tier structure offloads all of the incoming tasks to the cloud whilst two-tier structure may or not offload the incoming tasks. Thus, two simulations are run with same parameters except the offloading parameter to make a comparison. Simulation parameters are stated in the table in Figure 7.2.

It is important to note that both simulations are run in "TWO_TIER" parameter. However, in the single-tier structure, all of the tasks are offloaded to the cloud, making it single-tier although the parameter is set to "TWO_TIER".

Another important note is the "WAN Propagation Delay" parameter. Since, this study seeks to find the advantageous deployment solution (single-tier, two-tier with edge); WAN delay reflects all kinds of delay that requests face while being directed to long distanced cloud server.

Lastly, mobile device numbers provided are in the unit of batches containing 10 separate mobile devices each. Reason behind the use of batching is to keep simulation times in the range of 1 minute for repeatability.

7.2. Testing

This study aims to get reasonable results for deployment experiments without conducting them with real-world conditions due to its expenses. Thus, it is not possible to test the results with real data. However, tasks fail in the simulations if their service

Simulation Parameters	Value
Minimum Number of Mobile Devices	10
Maximum Number of Mobile Devices	100
Mobile Device Counter Step Size	10
WAN Propagation Delay (seconds)	0.5
LAN Internal Delay (seconds)	5
WLAN Bandwith (Mbps)	200
WAN Bandwith (Mbps)	15
Simulation Scenarios	TWO_TIER

Figure 7.2. Simulation Parameters

times exceed their maximum tolerable times which are in range of few seconds leading the parameters configured for the simulation is acceptable in this manner.

7.3. Deployment

There is no product deployed for the scope of this study. However, the configuration .xml files for the simulation settings for this study can be shared with parties. Please contact with the author of this document to access to these files.

8. RESULTS

Simulations are run for 5 iterations for both deployment structures. As it is mentioned above, all of the simulation settings are kept same both in single-tier and two-tier scenarios except the "Cloud Offloading Probability" parameter. After simulation runs, log files from two simulations are gathered.

The results of the simulation runs display the opposite of the expectations related to motivation of this study. In the two-tier structure, it was expected to be lower on the ratio of the failed tasks compared to single-tier ratio since there are more computing units involved and the distance to cloud servers are handled. However, results display that at each mobile device number starting from 10 batches, ending in 100 batches(of 10 devices); single-tier structure have lower failed task ratio, Figure 8.1.

There is another point can be inferred from the Figure 8.1. which is the fact that by the increasing mobile device number, failed tasks ratio is getting lower. This can be caused from implemented delay mechanism for this study or the simulation system itself. Since this occurrence in the results do not affect the scope of this study, which is the comparison between the two deployment scenarios, this problem is not investigated in the study.

Figure 8.2. displays the ratio of delay-reasoned failed tasks among the all failed tasks. For both cases, it can be seen that this ratio is almost higher than 98% for every device count which proves that simulation settings are configured as intended so in the study. Other small ratio reasons cover the capacity and bandwith problems in servers.

Figure 8.3. displays the distribution of the server where tasks are failed, edge servers or cloud server in Two Tier deployment. There are not any notable changes in the failure rates with the increasing device number. The thing here to note that, edge server in the two-tier structure fails to meet the T(max) condition of application requests much more severely than the cloud server. Thus, it can be deduced that the

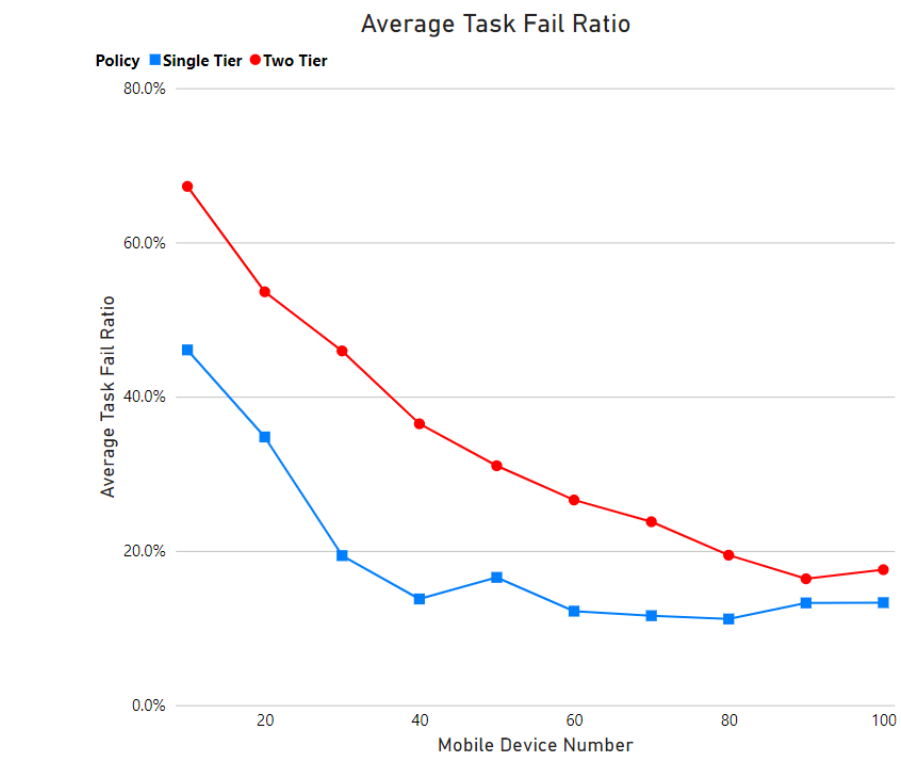

Figure 8.1. Average Task Fail Ratio: Single-Tier vs. Two-Tier

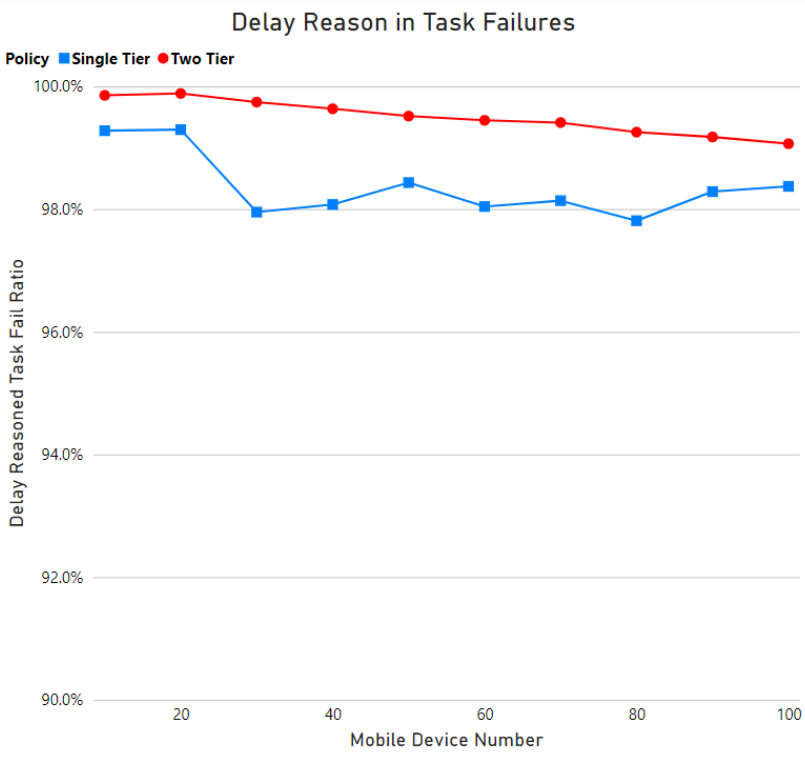

Figure 8.2. Delay Reasoned Failed Task Ratio

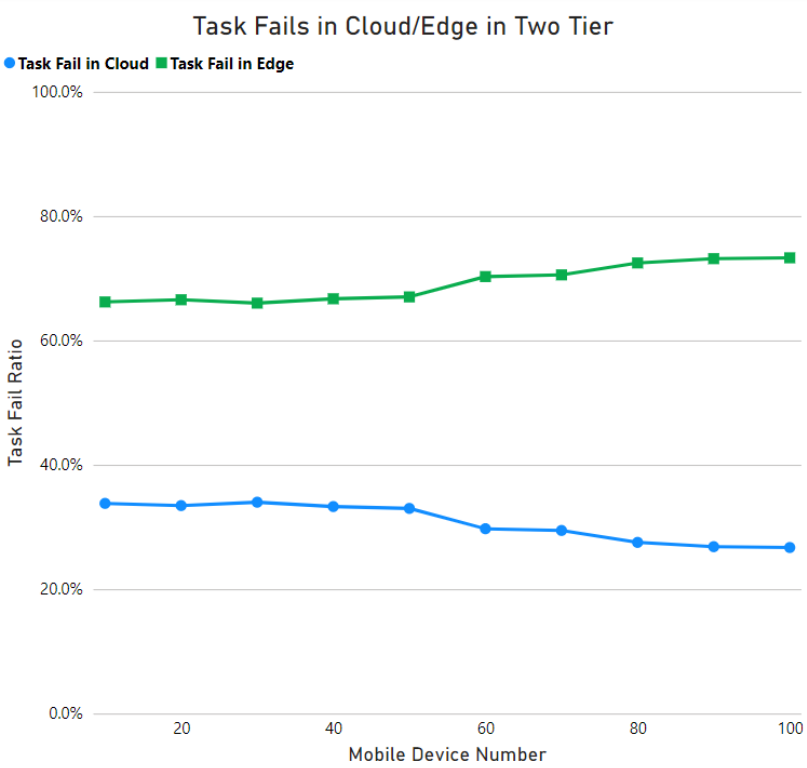

Figure 8.3. Ratio Comparison for Failed Tasks (Two-Tier): Edge vs. Cloud

congested edge server leads to the result that two-tier structure performs worse than the single-tier.

9. CONCLUSION

In this study, a different approach considering LLM deployment structures, using the edge servers. Through this motivation, different structures and their advantageous and disadvantageous specifications about the deployment is discussed in detail. Afterwards, a scenario to be simulated is studied with different types of LLM services with varying task complexities. In addition, server devices, network conditions, etc. are researched and finalized for the simulation purposes.

By running two simulation at each iteration, one for single-tier approach whereas other for two-tier approach, for gradually increasing device numbers for each iteration, simulation displayed that the single-tier deployment structure outperforms the two-tier deployment in terms of T(max) as came out the opposite of the expected in the study.

There is a possibility that this result might not reflect the real-world case due simulation settings and simulator program. The settings might not reflect the real-world scenarios or the simulator can include some modules that are not working as expected. Thus, in the next steps, reasoning behind this result should be investigated further since testing with real-world scenario is not possible as it is stated under this study.

Bibliography

[1] Keyan Cao et al. “An Overview on Edge Computing Research”. In: *IEEE Access* 8 (2020), pp. 85714–85728. DOI: 10.1109/ACCESS.2020.2991734.

[2] Shuiguang Deng et al. “Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence”. In: *IEEE Internet of Things Journal* 7.8 (2020), pp. 7457–7469. DOI: 10.1109/JIOT.2020.2984887.

[3] Tianshu Hao et al. “AI-oriented Workload Allocation for Cloud-Edge Computing”. In: (2021), pp. 555–564. DOI: 10.1109/CCGrid51090.2021.00065.

[4] Qianlin Liang et al. “Model-driven Cluster Resource Management for AI Workloads in Edge Clouds”. In: *ACM Trans. Auton. Adapt. Syst.* 18.1 (Mar. 2023). ISSN: 1556-4665. DOI: 10.1145/3582080. URL: https://doi.org/10.1145/3582080.

[5] Zhiqiang Shen et al. “SlimPajama-DC: Understanding Data Combinations for LLM Training”. In: (2024). arXiv: 2309.10818 [cs.CL]. URL: https://arxiv.org/abs/2309.10818.

[6] Cagatay Sonmez, Atay Ozgovde, and Cem Ersoy. “EdgeCloudSim: An environment for performance evaluation of edge computing systems”. In: *Transactions on Emerging Telecommunications Technologies* 29.11 (2018). e3493 ett.3493, e3493. DOI: https://doi.org/10.1002/ett.3493. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.3493. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3493.

[7] Hugo Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”. In: (2023). arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[8] Atay Özgövde Yağmur Göktaş Cem Ersoy. “Large Language Model Deployment on Edge Devices: A Simulation Study”. In: (2024).