Group 11

Title: Step Smarter, Not Harder : Queue-Aware Diffusion Sampling

Feedback: I like the title and the overall direction. (i)**The problem**: is to decide the early exit level based on the real time load. What is the reason of early exit, instead of caching? Do check out the diffusion scheduling work. Also, can you also make choosing DDIM or DDPM as part of decision? (ii) **The GAI model and the data set**: are yet to be decided. You may get some novel points by choosing an interesting data set. (iii) **The optimization and the system**: do remember to compare with non-optimized systems and choose the latency and quality metrics smartly. Do remember to separate the job time and queueing time, due to the load.

Group 08

Title: Adaptive AI Podcast Clip Generator - Wizard of POdz

Feedback: I like the title and the overall direction. (i)**The problem**: is to generate a story and voice clips based on the real time load. (ii) **The GAI model and the data set**: you are exploring 2X2 combinations of high/low LLM and TTS. This is sufficient for the project. What is not clear is the “request generation”? How many different kinds of requests will be emulated? Are you fine-tuning those models first?(iii) **The optimization and the system**: there are a lot of metrics considered. Can you simplify them a bit? The other part of the project requirement is to predict the latency. Can you build some simple regression or queueing model to predict the job latency and response times (including the queueing time)

Group 06

Title: Automating Medication List Data Capture: An On-Device AI Prototype

Feedback: The application is interesting but is not align well with the project requirements yet. (i)**The problem**: is to use LLM to reconcile heterogeneous data sources into EPR system. (ii) **The GAI model and the data set**: you will use vision-LLM and SLM models. How big is the data set? Is 20-25 list big enough? I am getting the point of using two-stage LLM? (iii) **The optimization and the system**: I am expecting you to present the performance analysis of this application. What is the

specification of your device? What is the latency of analyzing each list? How efficient is your system, compared to the Cloud-based solution?

Group 9

Title: GearShift LLM: Dynamic Model Switching for Efficient Inference

Feedback: Clear project and aligned well with the requirements. . (i)**The problem**: is to use two types of LLMs to improve the inference efficiencies. (ii) **The GAI model and the data set**: you have a clear idea on LLMs. But there is no description on the data sets. Which data sets are you going to use to emulate the user behaviors? One question per request? Or multiple questions per request? Also, what metrics to evaluate the quality of LLM outputs? (iii) **The optimization and the system**: Good that you have quite a clear picture on the performance aspect of the system. Additionally, can you think of the baseline systems where the “gear” can not be shifted? Also, how accurate of “predictive” model can predict the latency?

Group 3

Title: Automatic Room Furnishing

Feedback: This is a fairly baren proposal and behind the expectation. (i)**The problem**: is to build a furnishing application. What is the input to the furnishing model? Empty room only? Or additional textual description? (ii) **The GAI model and the data set**: you really need to start thinking about what generative models to use and where to get your data sets to train/fine-tune those models?(iii) **The optimization and the system**: say, when you have this model done, what is the latency and quality to furnish an empty room? Is there a parameter that you can adjust such that there is a tradeoff between quality and latency.

Group 4

Title: *FastRead: Optimized Large Language Model for Text Summarization*

Feedback: Good proposal and well aligned with the requirement. (i)**The problem**: apply different accelerating strategies to improve the inference performance for text summarization. Very focus. (ii) **The GAI model and the data set**: good that you know

what model and data set to use and define the suitable metrics (iii) **The optimization and the system**: you have defined three strategies, quantization, batch sizes and early exit. I think it's too ambitious to try all three strategies. Better to go into two or even one strategies, and go deeper into them. For instance. quantization strategies can be done in post-quantization manner. Different bandwidth may have quite different impact on the latency, depending on the implementation. There are also different choices of early exist. One aspect missing is how to emulate the request loads, i.e., how frequent and what type of requests shall be streamed into the system.

Group 2

Title: ???

Feedback: This is a vague proposal and not clear its alignment with the project requirements. (i)**The problem**: fast diffusion inference through quantization and caching (I think). What is going to beyond MixDQ is not clear yet. (ii) **The GAI model and the data set**: while the GAI models are clear, *SDXL-Turbo*, the data and evaluation pipelines are not clear. MixDQ has several data sets. Which one is your focus? iii) **The optimization and the system**: there are several directions mentioned, undistilled, and caching. I think it makes sense to focus on one and links it to the specific requirement of this course – load dependent control. Here are something to think further. What is the average latency to generate the image of different quantization levels? Can you find a parameter that can accelerate the latency per detection without dropping the detection accuracy? Can you think about how to tune such a parameter with respect to the load changes? Say, when there a lot of detection requestions, you tune this knob such that the latency is lower and the detection quality lower, and vice versa.

Group 1

Title: **Training-Free Detection of AI-Generated Images with Performance Modeling & Acceleration**

Feedback: This is a fairly baren proposal and behind the expectation. (i)**The problem**: the training free deep-fake detection. No clear on what aspect want to optimize. What to expect out of exploiting VFM embeddings ? What is the difference from the existing work? (ii) **The GAI model and the data set**: while the GAI models are clear, the data and evaluation pipelines are not clear. iii) **The optimization and the system**: fair that you have not learned much yet. Here is something to think further. You are now building

this training free detection system/application. What is the average latency to inspect a image? Can you find a parameter that can accelerate the latency per detection without dropping the detection accuracy? Can you think about how to tune such a parameter with respect to the load changes? Say, when there a lot of detection requestions, you tune this knob such that the latency is lower and the detection quality lower, and vice versa.

Group 13

Title: *Emotion-Aware Latency Modeling and Scheduling in Empathetic Chatbots*

Feedback: I like the proposal. It's clearly written and well thought through especially on the system optimization. This proposal aligns with the project requirements well. (i)**The problem**: priority based schedule for emotional urgent users. (ii) **The GAI model and the data set**: this is the vague part of this proposal and more thoughts are needed here. What chatbots do you have in mind? Not clear to me if the chatbot can clearly identify the emotion urgency? How to deal with the false positive and false negative of such identification? Or are you assuming this is a known fact? iii) **The optimization and the system**: this is the strong part of this proposal, exploring two schedule policies. There are few aspects you should think further: how many queues and how many LLM servers in the system? And, how do you want to emulate the requests of these two types of requests over time? How long is the evaluation windows? Maybe you can start out this part already to flush out all the details.

Group 17

Title: **??**

Feedback: This is a clear proposal and aligns with the project requirements, except the system optimization part. (i)**The problem**: realistic medical images of skin diseases based on the tabular data. At the first read, I thought cool and then started having some doubt if this idea will work. Typically, we conditional image generation, meaning that taking the tabular feature as conditions, and train a diffusion model that is able to take those inputs. What you are proposing is the opposite! I am curious if your approach will work better than the conditional diffusion? (ii) **The GAI model and the data set**: you plan to use stable diffusion and fine-tune it with LoRA, using ISIC Archive / HAM10000 dataset. iii) **The optimization and the system**: this is the part different from the

requirements. Here are some questions for you to think further. You are now building a image generation system. What is the average latency to generate such an image? Can you find a parameter that can accelerate the latency per generation without dropping quality? Can you think about how to tune such a parameter with respect to the load changes? Say, when there a lot of generations requests arriving at your application, you tune this knob such that the latency is lower and the detection quality lower, and vice versa.

Group 10

Title: Teaching Generative AI to Laugh: An Adaptive Meme System Balancing Quality and Latency under Load

Feedback: I like the title. This is a clear proposal and aligns with the project requirements. (i)**The problem**: developing a system that can generate meme images using LLM captions and diffusion images. (ii) **The GAI model and the data set**: while your goal is clear, the exact GAI model and data set are not clear yet. Is there any model/application that can do what you want? Question: is this necessary to have LLM generating the text first? Can you simply say the users will have their own requests. iii) **The optimization and the system**: this is a strong part of the proposal. It's a bit too complicated and ambitious. Can you focus on a smaller set of parameters of exploration? And, you can skip the DOE and ANOVA part as this is no longer the requirement for this edition. To verify the queueing, keep in mind that you need to verify the distribution of generation time per emem. An interesting aspect is that, you can control the length of gif duration based on the load. High load, shorter gif, and low longer gif. This is a unique control knob for your project. The choices of DDPM v.s DDIM, and diffusion steps are general to all projects.

Group 15

Title: *??*

Feedback: Um. The proposed ideas are too vague to receive "proper feedback". (i)**The problem**: you have three ideas but not sure which one to take and their alignments to the project requirements. The project is evaluated on the "novelty" of the GAI application, optimizing the latency-quality performance of this application, and application of predictive. If you take pretrain models, you just simply get the novelty from the other two parts. (ii) **The GAI model and the data set**: so far, I know you want to

optimize some LLMs, in the direction of context windows, prompts and tokens. Try to first settle down this part first. iii) **The optimization and the system**: difficult for me to give further feedback. In addition to what you have, an idea came to me – load-aware context engineering. Depending on the user load to your LLM system, you allow them to have different context prompts. For instance, when there a huge queue in the system, your LLM allows short context prompts to save the memory requirements and shorter the generation time. You can also combine this trip with the availability of thinking process.

Group 14

Title: An Adaptive Comic Generation Service System Based on SDXL

Feedback: This is a complicated/confusing proposal and seem to be aligned with the project requirements (i)**The problem**: generating comics through high/low quality models. You need to connect the “loads” as a driving force to switch the high/low quality models. I don’t see comic generation being essential here. You can generate any kind of images, right? What so unique about comic here?(ii) **The GAI model and the data set**: you are using stable diffusions to generate the comic based on the prompts. Are you going to finetune the stable diffusion first? Or you already find existing models which are good at comic generation. iii) **The optimization and the system**: this part is way too complicated to be feasible for the given project time line. The DOE and ANOVA parts are no longer the requirements for this project. You can actually skip this part. I am confused about the quantization part. Think carefully if you can actually implement load-dependent quantization. Them, you also want to explore cashing policies? How exactly? My suggestion is to focus on one optimization strategy and do it well.

Group 7

Title: Making Diffusion Models Faster with Quantization

Feedback: This is a clear proposal and seem to be aligned with the project requirements (i) **The problem**: generating images using models with different quantization levels. Though this is a clear goal, it is not clear why/when/how to switch the quantization levels. Think about how to best use of existing works (ii) **The GAI model and the data set**: which GAI models and what kind of data sets? Do you need to fine-tune the model first ? (iii) **The optimization and the system**: good that you define

the base line performance and the performance optimization goal, i.e., 30% latency reduction. Think further what kind of requests/loads you want to generate and the average response times per request is constant independent of the loads. You shall come up with an algorithm to adjust the quantization level.

Group 5

Title: ??

Feedback: the proposal is clear and requires more thinking to align with the project requirements (i) **The problem**: using LLM for data cleansing tasks. I like the idea but find the The data sets you mention might be too easy for using LLM. The alternate idea is to use LLM to extract structure table features from textual descriptions, i.e., from text to tables. (ii) **The GAI model and the data set**: the LLAM2 will be an easier option to build this application. As for the data set, I suggest to use a more complicated data set because UCR and Kaggle table data sets are fairly structure. (iii) **The optimization and the system**: this is the missing part so far. Say, you build this LLM-based cleansing application and being used to serve users. How are you going to make sure the latency of such application is constant against any load conditions? Can you find a parameter that can be adjusted in the following way? When many cleansing jobs requests are there, you only ask LLM to do quick cleansing which may have lower quality and vice versa.