DEPLOYMENT OF LLMs AT THE EDGE OF THE 6G NETWORK

Author: Alessandro Biagiotti *Università degli studi di Milano, Milano, Italia*

ABSTRACT:
LLMs are definitely the most interesting discovery of this century, they have been deployed in many different fields and have proven themselves worthy in many different situations. Currently, research for the next Mobile communication standard has started and the idea of pushing LLMs close to the end users has stimulated the interest of many researchers.

In this paper I will explore the opportunities and the difficulties of deploying LLMs at the edge of the 6G network as well as providing some brief insights on solutions that caught the attention of the scientific community.

KEYWORDS:
6G, AI, LLM, paper-reviewing

INTRODUCTION:

The impressive development of AI technologies that followed the publication of the notorious article "Attention is all you need"[17] has pushed research teams to try and incorporate LLMs and other such models into every aspect of human life ([10], [7], [18]). From consumer electronics to cars, aerospace and industrial applications AI is going to be a pervasive part of society from now on.

Every ten years the mobile communication standard changes and a new iteration is released, 2020 paved the way to 5G era, and now the research and development process has begun for the next mobile communication standard. As stated in [10], 6G is "poised to revolutionize the landscape by delivering unprecedented speed, reliability and security". This is a big promise, users will not be facing the next incremental improvement that they are so used to seeing nowadays with consumer electronics, a revolutionary way of interacting with the network is apparently waiting for us less than ten years from now.

6G is in fact meant to marry Wireless Networks and AI creating a chimera capable of speed, customization and pervasiveness. This new standard is meant to bring to life an infrastructure that will be able to support the immense requirements of the upcoming field of IoT.

All of this sounds incredibly promising on paper but, as it is known, big changes need to overcome even bigger challenges: How can one put AI in the hands of the users and make it work reliably? How can one make sure that LLMs behave correctly and reply in a timely fashion? How can one assure that the network is functioning correctly and the model is correctly trained? How can we handle the ever pressing problem of data security?

These are just some of the questions that one could be asking about 6G research. Due to the sheer size of the topic, for this paper I chose to delve into some paper reviewing alongside some considerations about the specific topic of Artificial Intelligence integration with the network, in particular in the next sections I will go through: (I) in which I will be giving a very brief overview of the 6G network, (II) in which I will be introducing the different opportunities that AI brings to the table, and I will give some pointers as to what components are essential in the 6G architecture, (III) in this section I will be explaining which are the biggest technical limitations for the implementation of LLMs at the edge of wireless networks, furthermore I will go through some solutions that have been provided to the above problems; in the last section (IV) I will dedicate myself to drawing some conclusions and pointing at some research topics that might be of active interest.

6G OVERVIEW AND AI AGENTS:

Mobile communication technologies have been a standard evolving since the very first generation, released in 1981. Every ten years a new standard is formulated and developed to propel the industry forward, updates have not been necessarily industry shifting, as an example, the move from 3G to 4G was more a performance bump [2], meanwhile the passage to 5G and 6G represented and will represent a big leap forward in terms of network capabilities, the mobile network and cloud infrastructure will be so tightly interconnected that will basically become one.

Currently, researchers from all fields and developers are studying and pushing the boundaries of network technology to move from theoretical study to actual practice, the first real world tests of the new network should be expected not earlier than 2030. According to Nokia[13] the six key aspects of the future generation of mobile communication technologies can be summarized in the following subsections.

NEW SPECTRUM TECHNOLOGIES:

5G, in essence, increased the datarates for all users in mobility both upstream and downstream, alongside a complete re-organization of the network putting the final nail in the coffin of an hardware network exclusively deicated to phone connectivity and calling, by using new spectral wavelengths and a new type of antenna that took advantage of millimeter waves (wavelength in the realm of the millimeters), the available spectrum bands were:

- 24 - 71 GHz which is high band 5G.
- 2.5 - 4.9 GHz which is the mid band 5G.
- 600 - 2600 MHz which is the low band 5G.

6G is thought to be implemented to use more wisely the space of millimeter waves and go beyond 71 GHz allowing for a higher spectral efficiency, the new standard will incorporate the 5G spectrum standard, widen it, and take better advantage of the pre-existing bands. The structure of the spectrum is still being studied but the foundations seem to be the following:

- 470 - 690 MHz extreme wide area coverage band.
- 600 - 2600 MHz wide area coverage band.
- 2.4 - 4.9 GHz urban band.
- 7 - 20 GHz urban subsection band.
- 24 - 71 GHz local hotspot band.
- > 92 GHz will be the extremely short range communication system and is currently being called the sub-TeraHertz band.

It's easy to see how the concept of dividing frequences into sub-bands based on geographical area coverage can be extended to the architectural model in order to achieve specific results, I will be using this division to go deeper into my architectural paradigm in (II).

AI AND MACHINE LEARNING:

On the particular aspects of AI and Machine Learning we have many diverse weapons that can be put to the service of the network or to the service of the users depending on the end goal that needs to be achieved.

AI for the end user comes especially in the form of the so called LLM or Large Language Model. This machine learning models work on text and are trained in a way that they are able to predict, inside a token sequence, the next most probable token, tokens are essentially sequences of words that can contain one or more words [1].

In the last couple of years models like BingAI, GPT-4, LLaMa, etc... Have seen a spike not only in their usage but also in the capabilities of the deployed models. The multimodal nature of some of the bigger models

can be leveraged to allow a more reliable and detailed resolution of the problems at hand and the use of techniques like fine-tuning, grounding or prompt engineering approaches like "Chain of Thought" and "Retrieval Augmented Generation" can turn the devices in the hand of end users into powerful assistants that can handle many tasks both on-demand and autonomously.

Private automation, robotics and healthcare are just a couple of the fields that will be most impacted by the addition of LLMs very close to the edge of the network. In the following I will be considering LLMs especially because of their importance in the current technological panorama.

LLM lifecycle can be summarized as follows:

1. Pre-training: During the pre-training procedure the model undergoes a general training procedure on very large unlabeled datasets, preparing it to handle user requests leveraging a general (foundational) knowledge [5].

2. Fine-tuning: Fine-tuning is an optional procedure that consists of aligning the pre-trained network with the user's necessities throught a partial or total retrain.

   Fine-tuning can be both complete, in which all parameters are updated, and parameter-efficient, in which only a subset of the parameters is updated; in this case the backbone is generally frozen and only a very small percentage of the total parameters is re-trained, to achieve this LoRA and other compression techniques should be employed, I will go through an introduction to LoRA in section (III). Effort is being put into discovering new and better ways to do parameter efficient fine-tuning, full network fine-tuning is not as interesting of a research branch because of the security implications linked to the topic [5].

3. Inference: Terminals input unlabeled data in the pre-trained and optionally fine-tuned model and the network generates an output.

The LLMs that we are in the habit of using at the moment actually do not undergo the fine-tuning procedure, services like ChatGPT and Bing-AI use different tricks that allow the models underneath to perform better in specific scenarios without the need of a true fine-tuning procedure. The OpenAI service ChatGPT uses context and prompt (Prompt fine-tuning or prompt-engineering), to enhance the model's capabilities [14]; Microsoft's BingAI has a different approach to the matter, undergoing a "grounding" process where, upon request, the model first fetches context information from the internet from a reputable souce (like Wikipedia) regarding the key points of the question and then uses the retrieved information alongside the user's prompt to generate a response[16].

AI for the network is going to be a collection of machine learning algorithms that handle all of the infrastructural necessities of the network: secuirity, resource management, model splitting, model migration to follow users' mobility.

SECURITY AND TRUST:

The changes in the spectrum distribution of the network and the use of AI in the network forces research to face a series of extremely important questions linked to private data security and individual security.

Giving AI more power and control means that it's necessary to be able to control its operation and make sure that is, not only, behaving according to specification, but also not being fed polluted data during the training or fine-tuning process by malicious third parties. Having extremely-high-frequency bands in the spectrum will expose communications to easy eavesdropping, it's therefore necessary to create security procedures that allow communications to remain private even when using these frequency bands.

With this particular aspect in mind, research efforts are being made to see if AI models can be used to make internet connections more secure, one of the propositions made was to allow AI models to change encryption schemes on the fly based on real-time security analysis results [10].

SENSING NETWORK:

The use of Machine Learning and AI techniques coupled with a very high number of sensors will allow the nodes of the network to sense the environment around them, this is an extremely useful technique to

generate a Digital Twin of the real world, storing precious information about the surroundings and generating information through LLM inference.

Regarding the above Minrui Xu et al. [18] wrote a paper discussing how the multimodality of todays LLMs alongside a wide array of sensors could be used to generate automatically car crash reports, in the field of automated driving, that can be further refined by sending them to a more powerful LLM deployed at the edge of the network.

EXTREME CONNECTIVITY:

The level of the services that are probably going to be brought to 6G require extremely fast and extremely low-latency communication, therefore the Ultra-Reliable and Low-Latency Communication service will be refined and improved to serve the new standard and to bring the latency below one millisecond.

NEW ARCHITECTURAL MODELS:

6G is thought to work alongside cloud and allow users to experience it differently, the paradigm of having big datacenters at the top of the cloud that handle all of the computation will be shifted towards an extremely distributed architecture centered around data management pipelines that involve only the devices in the hierarchy that are of use to the computation itself.

This means that the 6G network will be deploying, alongside the big datacenters at the top of the cloud, a series of intermediate stations that grow in computational power the more the data goes up in the hierarchy. Such intermediate nodes will be the ones that handle all of the computation associated to the infrastructure management, they will also manage all of the LLM-associated tasks linked to users (inference, fine-tuning, training, etc...).

In the next sections Ill be diving deeper in the opportunities, challenges and solutions associated with the deployment of LLMs at the edge of the network.

OPPORTUNITIES:

In this section I will present an architecture that can merge all of the more important traits seen in literature. Based on how the actual network will be shaped it will be more suited for certain types of situations rather than others but I am going to try and condense the structure of the various models proposed in the very recent literature into a series of shared features.

The idea of implementing AI as an agent in the field of 6G networking is rooted into the idea of creating the perfect personal assistant that is also, upon request, capable of leveraging professional knowledge in any given field. Attempts have been made in the past on the consumer level with clearly inferior models (e.g. "Siri" or "Cortana") but now the objective is to create a seamless and encompassing ecosystem that can adapt and provide the user with information on his own self that goes beyond anything we can currently imagine.

An example that often recurs in the literature to explain what the potential of AI, is to have a set of non-invasive sensors that can provide with discretely precise measurements of health data (bpm, glicemy, pressure, etc...) and having a model on a device, like a smartphone, that is able to understand whether the values are in the norm and eventually provide pieces of advice to make them better. This would have to be inserted in a framework that is also able to reliably understand the user's attitude and context to avoid annoying push notifications and incorrect (potentially harmful) advices.

The question that I will try to answer now is what is required to have a network that can handle users, provide them with very custom experiences and react to situations and stimuli automatically without the need of human intervention.

END:

Going from the ground-up we can build the network architecture starting with the sensors, the end devices, basically a Fog of computational devices of all matters and sizes able to do computation, generate data or process information. Less than the 1% of these nodes is going to be capable of running models and usually they will be extremely simple in nature, probably the only LLM available would be the ones in the order of <1B.

The Fog is supposedly a very heterogeneous mass of computing devices. The most important characteristic of the Fog is the variance of computational power of the devices. Two different classes of devices came to mind when thinking about the Fog structure:

Sensors - which are capable of handling only the measurement of certain parameters and some limited computations (usually battery powered).

Non-sensing devices - this category is quite broad but I would consider all of the personal devices to be part of it, anything that the user can interact with and is connected to the 6G network should be considered a bigger node in the Fog and should be capable of running AI models.

EDGE NODES (EN):

Edge nodes in the recent literature are basically devices with more compute power than the END nodes or other ENs that they manage. As with END nodes, ENs slide into tiers based on the compute power and the occupied position in the cloud hierarchy.

Their function is to be able to handle more complex inference and model training, such ENs should have enough power to do inference on classic models like GPT-3 and, depending on their position and power, with some tricks, should also be to undergo training and fine-tuning of such models.

Considering the literature ENs are considered to be larger downstream nodes inside the network capable of handling complex inference using the results coming from smaller models at the end of the network, the idea provided in [18], was to be able to do an automatic crash report for cars using LLMs running locally on the car or on the phone and aggregating the initial results at edge level, enriching the analysis with more context information that could only be generated utilizing more powerful LLMs.

In other cases the LLM controller is used to understand the user's necessities and modifies pre-existing models accordingly, like in [15]. Another solution seems to be more suited for a RLM approach, using a RLM trained to do efficient dispatching of resources [11].

If we consider a more industry-driven example, even in the field of heavy machinery or energy production, it's very useful to have a system able to take a decision extremely rapidly based on a certain situation. Modern

control systems are able to take action and make corrections rapidly and precisely, providing excellent results and avoiding costly mistakes, having an LLM that is instead able to make more "management-oriented" decisions in real time and based on data might provide useful to avoid energy or time wastes.

Going back to the division shown in (I) we can see how the various bands have been divided based on the amount of surface covered, therefore I think that tracing a parallelism and having a system based on small cells of varying sizes that are all contained into each other is quite a natural way of evolving the architecture. Each cell will be having one or more ENs that are responsible of helping downstream nodes with the harder tasks[1].

Keeping the original schema shown in (III) I envision the following cells:

- Femto-cell: Extremely small cells that contain a very restrictive amount of users, at most 100, and are supposed to take advantage of the sub-TeraHertz band for low-distance communications. A cell like this one could be an autonomous car or a bus and the EN on board should be able to do extremely-light inference but most importantly should be able to do data aggregation from the various sensors present in the cell.
- Neighbourhood-cell: A small cell that contains less than 10.000 users and is controlling an entire neighbourhood, this cell should contain at least a couple of ENs capable of handling inference at a discrete level (nothing more than GPT-3), and should be able to take part in the fine-tuning process, depending on the size of the cell they could also be involved in split-training procedures.
- sub-Urban-cell: To avoid making too steep of a jump and overworking upstream nodes, there should be some ENs handling sub-Urban-cells of at least 100.000 users, these nodes should be capable of inference, even in the harder cases, and should partake in the split-learning procedure with low-tier nodes.
- Urban-cell: The urban cell is the highest in the hierarchy before the CLOUD and can contain millions of users. Since the organization of the network is tight the requests floating this high up will only be the harder ones (that still have important latency limitations), therefore at urban level I expect there will be a series of high-power nodes capable of: inference, split-learning and more low-power LLM training.

CLOUD:

The last piece of the puzzle is the system of datacenters that take on the most complicated pre-training challenges that cannot be handled by any other nodes in the architecture or take on requests that do not have latency requisites and require a high level of precision.

If we consider the GPT-3 model, which is the one that is the name that appears the most in recent literature, it would be necessary to have more than 1000 GPUs running non-stop for more than 4 months to have the complete training of the 175B parameters that compose the model [5]. This task would be impossible for the any low-power device, regardless of the position in the edge layer or the END. It is therefore necessary to leverage the power of big datacenters positioned at the top of the network hierarchy.

Essentially the datacenters would be used to train models (using some tricks that I will introduce in the next section), store data and do important inference on extremely big datasets.

In the next section I will go through the various challenges that we will be facing when the moment comes to start implementing the fusion of the service architecture of 5G with the AI agents and why the upgrade is not as easy as it might look like in the above explanation.

[1] All of the characteristics of the cells explained in the following are purely based on speculation and take in account the utilization of all the more advanced distributed learning techniques, some of which have been shown in section (III).

TECHNICAL LIMITATIONS:

In the previous section I took some shortcuts to give a clearer view of the core components of the 6G architecture. It's now time to look under the rug and explore the problems and some potential solutions that have been proposed.

The solutions that I will be providing are not mutually exclusive and all of them find a place inside the architecture I explored in the last section, before introducing them I want to quickly expose the problems that I'm trying to solve.

The first problem that needs solving is the model size.

If we consider the network architecture introduced in the last section the network nodes will be partitioned based on their computational power into the different tiers. Since the vision for 6G is to have the LLMs come closer to the user end there is two problems that need solving:

- The latency: we can put all of the big models (like GPT-4, Stable diffusion, LLaMa405B, etc...) in the CLOUD, while keeping lighter models like (LLaMa8B) closer to the user. Having such an organization would mean that the end users would need to wait for a time proportional to the quality of the response.

  Clearly, if the intervention of a model such as GPT-4v is required then the request will subsequently experience very big latency due to how big of a leap inside the architecture the message is going to need to do:

  - Sending the request from END to CLOUD, passing therefore through many different ENs that, at the moment, do not serve any function other than redirecting the request.
  - Waiting for the request to be processed and the inference to be made.
  - Sending the response from CLOUD to END.

  Since the operation of inference is already costly it's much better to be able to do inference (even on bigger models) as close to the edge of the network as possible, which in turn leads us to the next problem

- The space and compute: having to load models in memory or even saving them is extremely expensive, that is because of the amount of resources required. As can be seen in [9] the dimensions of required VRAM in GPU for LLMs at different ranges of performance (from GPT-3 to bigcoder/starcoder) require between 350 and 30 GB of VRAM, which is already extreme for any computing device that might inhabit the Fog, even at the lowest end of the model spectrum.

  Even the requirements in terms of basic storage are completely out of the capabilities of the majority of devices inside the Fog (175 GB for GPT-3 is a lot, especially if we consider that the user might not have stored locally just the one LLM but might need to deploy different ones for different pourposes).

  Last but not least, doing inference on the model or training the model will most likely require more compute power than the one available to most of the devices in the Fog.

It's clear enough that what is required here is to be able to employ smartly all of the resources at our disposal to allow efficient pre-training and fine-tuning operations, while, at the same time, being able to save more than one model on board in order to serve END devices as fast as possible when necessary. To make matters worse, when fine-tuning a model with users' data the procedure should be done as securely as possible, meaning that user's data should never leave their device. The safety hazard should be taken with the utmost caution since we are talking about an unprecedented amount of extremely sensitive data coming from different spheres of a users' life.

FEDERATED LEARNING:

Federated Learning is the first distributed learning technique for machine learning models with a very strong focus on users' data privacy. It was invented in 2017 by Google engineers trying to update the Android word-predictor without taking the users' data from them.

At a very high level FL works by storing a local copy of the current version of the model, then the model will undergo training based on local data for a certain amount of time, in our case we will be talking about updates to the dictionary and new inference rules; lastly, once the epochs counter reaches 0, the resulting

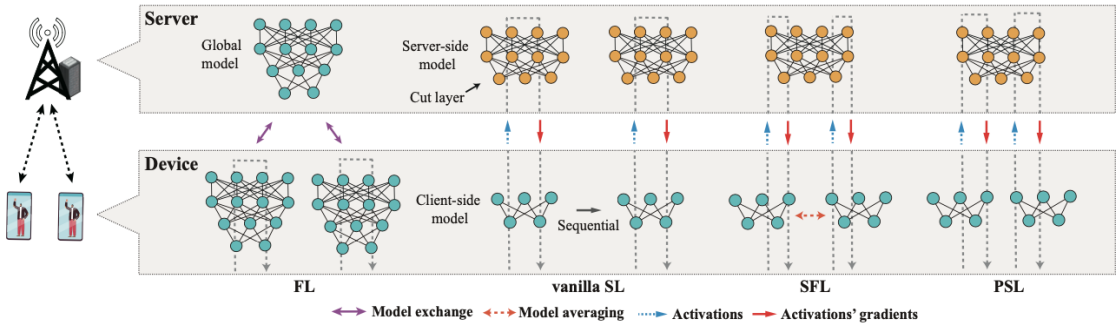

Figure 2: An example of the structure of a network implementing SL for model trainig [12]

model undergoes gradient backpropagation, the gradient is encrypted and sent to the head node responsible for the cumulative update of the model.

Even without a deeper dive in the inner workings of FL it's really easy to see how difficult it is to extend the above technique to the world of LLMs, having small extremal terminals handling their own copy of the base model is not possible for two reasons:

- **size**: The models are simply too large to be saved entirely on modern day mobile hardware, the only possible solutions to this issue would be to do model compression (changing the precision of the float values in the weights matrices of the model) or doing a hybrid of federated and split learning.

  Furthermore the model would need to be moved repeatedly between the center node and the various END nodes participating in the training procedure, that would put extreme amounts of strain on the network, even in the case of compression.

- **updating the models**: The fundamental concept of FL is to have a decentralized training process thanks to the participation of different nodes in the network, but small nodes such as the END nodes cannot possibly handle LLM inference.

  Let us be reminded that most of the LLMs currently in use are running on cloud hardware, even newer artificial intelligence features coming to consumer technology, especially in the case of smartphones software, are cloud based, only the very smallest models can actually run on local hardware (like Gemini mini [4]).

That said I think that FL is an amazing instrument to be used to pre-train bigger models like GPT-3 175B using the cloud datacenters alongside Urban ENs, a distributed procedure is surely much more efficient than locking an entire datacenter to solve the problem of pre-training one single model for weeks or months on end.

SPLIT LEARNING:

Split learning (SL), in a way, is an upgrade of the FL technique. SL allows model the division of the model in various sub-models that can be trained separately, the backpropagated gradients can then be sent through the network to the "sibling nodes" to allow them to procede with their own training as is shown in Figure 1. The way the model will be divided is dependent on many factors e.g. size of the model, privacy concerns of the user, level of customization required by the user, network topology and level of mobility of the user, energy efficiency, etc... Since the variables are many it might be a very good idea to allow the closest EN To handle the decision on how to split the model. It could be possible to deploy an RLM on ENs that is able to chose how to slice the model in order to strike a balance between having too many different slices, energy efficiency, latency and other parameters.

SL will be a very powerful tool and a central part of the software architecture deployed alongside 6G since it allows users to have powerful models at their disposal without sending their data to the cloud (the fine-tuning is done on-device and the end node will be the starting point for the inference process). SL is just a piece of the intricate puzzle that we need to build, it allows us to have fast training procedures, supposing the model has been divided effectively, but it forces us to have the model fragmented in different parts of the network

and this complicates things because we want the model to be "local" to the user. To grant this locality property we will need to have load balancing algorithms or RLMs that can handle the choice of model migration when the user is in a mobility phase, it should work in such a way that it can strike a balance between latency and having the rest of the model as close as possible to the user.

Although SL makes the training process possible while keeping user's private data safe, it introduces complications on the inference side, depending on the number of sections that the model has been partitioned into the inference process is going to require the participation of all the computing elements that store a part of the whole model. This implies the necessity of a discrete number of hops inside the network, which results in added latency.

LOW-RANK ADAPTATION

Low-Rank Adaptation (LoRA) is a compression technique that freezes the pretrained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of training parameters for downstream tasks.

This technique really strikes the point because it's of the utmost importance to be able to reduce the number of trainable parameters for the constrained devices that will need to face some sort of fine-tuning procedure in the END layer. Instead of doing a fine-tuning of the complete model we do the fine tuning of a reduced version of the original weight matrices, the procedure is well explained by Figure 3. Per the studies made in article [8], given a weight matrix $W_0 \in \mathbb{R}^{d\times k}$ we can represent the weight update matrix via a low-rank decomposition like:

$$W_0 + \Delta W = W_0 + B \cdot A \tag{1}$$

where $B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k}$ and the rank r, which is an hyperparameter of the network, is much smaller than either d (which is the full rank of the dense layers' decomposition matrices, in the case of GPT-3 the count can get up to 12.288) or k. The result is that instead of working with the full $d \times d$ weight update matrix we are actually working with two extremely underdimensioned matrices.

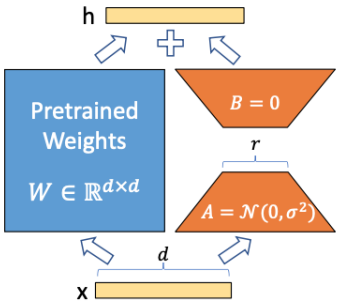

Figure 4: The LoRA approach proposed in [8]

The gradient updates for the matrix W_0 are frozen while A and B contain the trainable parameters.

The empirical results when using GPT-3 175B were the following:

- slicing VRAM of up to 2/3 since there is no need to store Adam optimizer states for the frozen parameters (they went from 1.2 TB to 350GB of VRAM usage).
- If the rank of the decomposition was 4 and, of all the transformer matrices, only the value projection and query matrices were adapted the checkpoint size is reduced of circa 10000 times which in turns means very easy model switching due to the forgettable size of LoRA weights.

To provide another reason for LoRA, it's really easy to implement next to other techniques. I would go as far as to say that it's essential to allow the correct implementation of distributed learning techniques like SL, because even the partial retrain of a network for fine-tuning might be too expensive for the underpowered nodes at the END of the 6G network. What is even more impressive is that, by using LoRA, it's possible to change the alignement of the LLM by swapping the A and B matrices shown in Figure 3; this concepts fit perfectly in the frame of having a split model, if we consider that the part related to the user's specific data (the A and B matrices) will be living on the END device, we can imagine having a series of different trained low-rank adapters that can be swapped to change the behaviour of the LLM and make it more adapt to the situation. If we factor in the fact that having a snapshot of the LoRAs takes (in the case of GPT-3) 35MB because it's only necessary to save the matrices themselves and not all of the weight matrices, it's very easy to see how doable a development like this would be.

As an additional point of interest, that will not be covered, there have been developments in the field of LLM optimization and QLoRA was born out of the mix between quantization techniques and LoRA. The

necessity of expanding the LoRA approach was born from the fact that the fine-tuning operation for a model like LLaMA 65B (full 16 bit fine-tuning) requires over 780GB of GPU memory and quantization only works at inference time. The approach shown in [6] explains how it is possible to quantize a pre-trained model to 4 bit precision and then backpropagate the gradients after doing the adapter injection to train them; this revelation renders the memory requirements for fine-tuning worth considering (less than 48GB of GPU memory).

FUTURE CHALLENGES:
Everything that has been introduced to this point has that *je ne sais quoi* of feasible, we have all of the pieces needed to build the puzzle and the direction seems to be the right one, all that is required now is just research to see how the various components interact and if there is anything that can be done to increase the performance, or the efficiency.

As with any other progress in the field of computer science, but more so with this one than any other, having the guarantee of dealing with an extremely efficient system that can do inference and train complex models with by making compromises with the lowest possible overhead might reduce dramatically the amount of ENs that would need to be deployed at the edge of the various cells introduced in (II) (ENs are very costly to produce because necessitate the help of GPUs to perform training and inference efficiently).

Many important steps forward have been made when it comes to LLM optimization and we achieved very good results even when deployed in resource constrained conditions, this does not mean, though, that the research is over. While with a model like QLoRA it's possible to fine-tune a 65B model in just under 24 hours with a single GPU there is going to be the need of fine-tuning and training more than just one model at the same time, in the meanwhile the ENs and END devices will have to cooperate to do inference and solve more classical requests that do not necessarily require AI intervention.

That is why it's still necessary to put a lot of work in the field of inference and training optimization at low precisions using instruments like LoRA, QLoRA and quantization; furthermore, as was shown in (III) SL is a very powerful technique that makes training possible even in constrained environments (especially if paired with quantization techniques), more work should be put in trying to solve the problem of finding optimal splitting points for models using RLMs or other heuristic techniques based on different network parameters (as was shown in [3]).

Last but not least, there is no framework that puts all of the ingredients together, therefore it's of the maximum importance to start working in that direction to solve the orchestration problems that arise from working with an architecture as complex as 6G's. Some questions that could be interesting for future research could be the following:

- How to handle the mobility of users and how to guarantee the principle of locality of the model shards generated via SL?
- How to efficiently split the network in order to maintain good inference performance
- How to handle model caching

References

[1] Cosmia nebula et al. *Large language model*. URL: https://en.wikipedia.org/wiki/Large_language_model.

[2] Baeldung. *Mobile Networking: 1G vs. 2G vs. 3G vs. 4G vs. 5G*. URL: https://www.baeldung.com/cs/mobile-networking-generations.

[3] Francesco Binucci et al. "Enabling Edge Artificial Intelligence via Goal-oriented Deep Neural Network Splitting". In: (2024). arXiv: 2312.03555 [eess.SP]. URL: https://arxiv.org/abs/2312.03555.

[4] Dave Burke. *A New Foundation for AI on Android*. URL: https://android-developers.googleblog.com/2023/12/a-new-foundation-for-ai-on-android.html.

[5] Ning Chen et al. "Towards Integrated Fine-tuning and Inference when Generative AI meets Edge Intelligence". In: (2024). arXiv: 2401.02668 [cs.DC]. URL: https://arxiv.org/abs/2401.02668.

[6] Tim Dettmers et al. "QLoRA: Efficient Finetuning of Quantized LLMs". In: (2023). arXiv: 2305.14314 [cs.LG]. URL: https://arxiv.org/abs/2305.14314.

[7] Tiago M. Fernández-Caramés and Paula Fraga-Lamas. "Forging the Industrial Metaverse for Industry 5.0: Where Extended Reality, IIoT, Opportunistic Edge Computing, and Digital Twins Meet". In: *IEEE Access* 12 (2024), pp. 95778–95819. ISSN: 2169-3536. DOI: 10.1109/access.2024.3422109. URL: http://dx.doi.org/10.1109/ACCESS.2024.3422109.

[8] Edward J. Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: (2021). arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

[9] Hugging-face. *Optimizing LLMs for Speed and Memory*. URL: https://huggingface.co/docs/transformers/v4.35.0/llm_tutorial_optimization.

[10] Navneet Kaur, Naresh Kshetri, and Purnendu Shekhar Pandey. "6AInets: Harnessing artificial intelligence for the 6G network security: Impacts and Challenges". In: (2024). arXiv: 2404.08643 [cs.NI]. URL: https://arxiv.org/abs/2404.08643.

[11] Zheng Lin et al. "Pushing Large Language Models to the 6G Edge: Vision, Challenges, and Opportunities". In: (2024). arXiv: 2309.16739 [cs.LG]. URL: https://arxiv.org/abs/2309.16739.

[12] Zheng Lin et al. "Split Learning in 6G Edge Networks". In: (2024). arXiv: 2306.12194 [cs.LG]. URL: https://arxiv.org/abs/2306.12194.

[13] Nokia. *6G explained*. URL: https://www.nokia.com/about-us/newsroom/articles/6g-explained/.

[14] OpenAI. *Fine Tuning*. URL: https://platform.openai.com/docs/guides/fine-tuning/when-to-use-fine-tuning.

[15] Yifei Shen et al. "Large Language Models Empowered Autonomous Edge AI for Connected Intelligence". In: (2023). arXiv: 2307.02779 [cs.IT]. URL: https://arxiv.org/abs/2307.02779.

[16] Andreas Stöffelbauer. *How Large Language Models work, From zero to ChatGPT*. URL: https://medium.com/data-science-at-microsoft/how-large-language-models-work-91c362f5b78f.

[17] Ashish Vaswani et al. "Attention Is All You Need". In: (2023). arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[18] Minrui Xu et al. "When Large Language Model Agents Meet 6G Networks: Perception, Grounding, and Alignment". In: (2024). arXiv: 2401.07764 [cs.AI]. URL: https://arxiv.org/abs/2401.07764.