Lecture 6 - Inference and Parting Thoughts

Gidon Rosalki
2026-06-30

Notice: If you find any mistakes, please open an issue in the github repository

1. Introduction

We began from networking, and performance, and how this may be used in terms of training neural networks. Today we will start asking about using these networks. We will note that training these models is incredibly expensive, training GPT 3.5 on AWS (for example), would cost around 45 million dollars. So for a number of years, the training was expensive, and the use was minor, so the training was not really worth it. These days, with agentic AI, it has become a much more useful (and theoretically profitable, though I remind you that AI companies are yet to turn a profit) market.

We do need to concern ourselves about inference, and how these networks run. Where initially, we ask a question, and it outputs an answer, with the agentic and reasoning models, the inference is becoming more an more expensive, since they create their own questions, and answer them internally, and begin using other tools, taking their output and internalising, increasing the context, and starts costing more and more.

2. Inference

When running our models, as we discussed before, we will pipeline our GPUs for pipeline parallelism, and may also include data parallelism, but how does this work in inference? How do we request things from the model? Well, we cannot request directly from the pipe stage GPUs, so we send a request to the frontend of the server, which sends requests to different pipe stages (which may be running further tensor parallelism, to reduce latency). Since we have many different independent requests, we may send these to different pipes of GPUs, without an issue.

At some point, we started specialising parts of the model into groups of experts, since we can split a model into better experts at different subjects, which means we only need to run part of the model for a given problem. For example, we do not need the parts of the model that are trained on cooking, when answering a coding question. In order to choose experts, we train another network, which given a question, can say with high probability to what expert the question should be sent. As mentioned above, by having different experts means that we are running less memory, less GPU, less everything. This makes running our model significantly cheaper, which is something that we want.

Training this together is a little more complicated. Initially, the router network will route somewhat randomly, but quickly learn that some experts are more trained than others, and just send the requests to those, which means we have achieved little. A method to resolve this is to force the router initially to distribute uniformly across the experts. We can also then run the experts simultaneously, and maybe even run numerous copies of the same expert, and answer lots of questions much more cheaply. Problem is, the communication between the router network, to the experts running on different computers becomes expensive. Overall, everyone agreed that it was not worth it. Until, DeepSeek came along, and had written their own networking library explicitly for this, and now every model uses parallelised experts, and this sort of library, to make it much more efficient.
There is a modern attitude that we may do this more cheaply. Using an FFN to find the expert is much more expensive than say a lookup table. There are all manner of companies looking at ways to do this, but this remains the current state of the art. We are essentially considering that making this network is essentially a way to compress information. There are many other theoretical methods of doing this, like the above example of a lookup table, and there are theoretically cheaper ways to compress this information than an FFN, but it is currently all theoretical, and not practical.

Making inference cheaper is also important, because whenever we run an entire prompt of many tokens, the LLM returns *a single token*. To get a full answer, we append this to the end, and repeat the request, and continue doing this until there is a complete answer. Just adding an extra token like “please” to the

request can cost millions in total, since this "superfluous" word is run through the LLM many many times, and running through the LLM begins to cost a lot of money. We will note that thanks to attention, we do not need to compute as much on subsequent runs with a single new token, since we have already computed the connections between most of the tokens, and this is "stored" in the LLM's attention.

Another problem is the amount we need in memory, and how much it is changed. Loading a gigabyte into a CPU, or a GPU, is slow and costly, no matter what. Since our model is huge, we run into this problem when doing inference. To try and resolve this, we split the model, and run different parts of the model on different parts of the datacentre.

Additionally, there has not been much research or work on computer architectures for around 40 years, since most the concepts of pipelining, branch prediction, and so on are old concepts, and there has not been much to do beyond that. It's only recently that we've started working on these deeply specialised architectures, like Nvidia's Groq, which has restarted research into new specialised architectures.

3. Gil'S Thoughts

3.1. Entry to the Workforce

We are about to enter a very interesting world of work. Gil's team are hiring all the time, he estimates that Nvidia are recruiting about 1000 people per year in Israel alone (so, the entire output of HUJI, the Technion, and maybe a couple of others). Unfortunately, beyond this, the world of the job market is not great. AI has had an impact, but he holds there's more to it than this. There are companies that will be severely impacted by AI, but there will be others that will be massively aided by AI.

Israel has a huge number of startups related to the medical world, and Gil is certain that we will also be able to massively use AI to create new medical techniques. The problem is the inbetween times, where we are not quite sure what we are doing. Universities will need to reconsider what they teach, it's important to know that things like Java exist, but like assembly, teaching it is no longer the meta. This is the world that we are now entering, "have fun, good luck". We will overall need fewer software engineers. The best, we may still need, but most we will not need. We will need those who are the cutting edge, that research, and create new concepts, but the "workers", that just write code, we will not.

Those that do not love this field, will not survive in it for much longer. Sorry for those that hated their degree just for the end salary, because they are not going to succeed here for much longer. Even those that are middle / bottom of this world are also not going to succeed. What is good here? Difficult to define, and we do not really know well enough to define, but it is definitely not dependent on grades.

Interpersonal communication is important here. Gil drinks more coffee with his team, than with his wife, because he sees them so much, and being able to communicate with them is important. If you are offered a drink at an interview, then always say yes, because this is already the test. Take the couple of minutes by the coffee machine to already start building a relationship, and being interesting to them. Before going to interview, go above and beyond, do your homework. Do not just prepare with leetcode, but also read up on what they do. If your interviewer has spent decades on networking and AI, be able to ask a question / know a few sentences on what they did, and what it's about.

Referrals are important, but not the end of the world. If Gil opens a new junior job, then he closes it in 2 hours, because he has 300 CVs. If someone else there knows you, has good things to say about you, then this can make your CV jump to the top of the pile.

3.2. Questions

3.2.1. Quantum computers

This is a world that has grown over the years, but is still not there. He thinks that in the next 10 years there will not be quantum computation. He is not aware of research or work in the field in Israel, though there is some in Europe, and the UK. The big money in this field is with companies like Google, that can afford to spend lots of money on the research, without making money from it yet. Most of the work is in

Physics, rather than CompSci at the moment (maybe aside from the algorithms relating to error correction, and so on, but it is hard to believe that a bachelor's student will do that).

In short, right now, quantum computers are not most useful. If Gil gave you a quantum computer right now, you probably would not even bother turning it on, since the only application we have for it is breaking cryptographic algorithms. Beyond that, we have not really found another algorithm that it can do better than traditional computers.

3.2.2. Tools Most Missing to Students Entering the Workforce

Firstly, play with AI. It is a very powerful, and useful tool, that is very relevant to the workforce, which models are most powerful, most effective, and so on.
He does not really know how to get there, but at the end of the day, the most useful skill is thinking flexibility. It happens a lot that we learn something, and it is in our comfort zone, and from there build a lot of our beliefs, and comforts. We need to get used to these base concepts becoming obsolete thanks to AI. Learn to think broadly, and not deeply, to be ready to change our thoughts when the world changes, and be adaptable to the way things change suddenly. It happens a lot that Gil builds plans for the next year, and considers what actually happened the previous year, and sees that there is no connection whatsoever, due to how much the world has changed.

Be able to show that you know how to use AI beyond writing a prompt. For example, ask for an architecture to solve the problem, then ask for comparison between this architecture, and others, and why it chose this, what the trade-offs are. Demonstrate knowledge of the many steps before you even ask it to generate code.

In general this returns to communication, learn the language. If there is a company that you really want, then work more for that, in order to try and get it. For example, in Italy, if you show the extra mile of knowing a couple of sentences in Italian, then they love it. The same applies in interviews, show you went the extra mile, know a bit about what they do, and how they do it, and it goes down well.

3.2.3. Research, master's degree, and AI

Gil thinks that in Israel there is a severe lack of researchers in Israel. Gil has a requirement to recruit researchers, he would much rather recruit Israelis, but mostly recruits foreigners, because we do not have that many researchers. It does not really matter what you research, just that you research, and know how to research. Not everyone can be a good researcher, and be good at it. There is something lonely about researching, and it's a lot of work. Every researcher handles his own area, and it is very lonely as a result. The chances that you have a researcher adjacent to you, that works in a close enough field to discuss, is very small. Turning engineers into researchers over 3 years is not worthwhile, since we may get there and discover they do not enjoy it.

So, research, learn *how* to research, though admittedly in a relevant field. Researching flower arranging is not helpful here.

3.2.4. Higher degrees

For Gil (not everyone) he prefers a master, than someone with 2 years in the field. He does also prefer PhDs, since they have learnt to research properly, but a master's is more useful to him than an engineer from the field.

The ideal combination is to do an internship during the PhD. Most of the world does this, where every summer you go to your internship, and at the end hopefully have a related paper, and have gained experience, with your supervisor helping choose internships that are relevant and useful.

Do not take a few years to work, and then go back to do a PhD. It does not happen, people do not manage to do this. Do the PhD, and find an internship to combine with it. Gil commits to try and take an intern a year from Israel. He currently has 3 in Zurich, since he cannot find interns / researchers here in Israel.

We seem to have created a unique model here of a student job. It's not the worst, but it's not ideal. Lots of supervisors do not like it, and it's a bad model. We seem to be unable to stop it, since all companies would need to stop at once. Overall, Gil would say it is more important to research, and focus on that,

and once you have done your research, then go work. Remember, it is unlikely that a professor will allow you to go do a student job externally, since they want you in their lab, researching.

Gil has an architect in his team, that is a biologist. He hired her because she is an incredible researcher, with a remarkable ability to learn, and he trusted her to make up the differences in technical ability herself. For him, the important part is knowing how to learn, and how to research.