Aidan Farnen

CSCI 717

Assignment 0

Mini Report

* + 1. **Introduction**

For my ciphertext I chose to generate my own using toebes.com. I looked up a semi random quote that seemed to have a lot of characters in it that was the following: “Movements have narratives. They tell stories, because they are not just about rearranging economics and politics. They also rearrange meaning. And they’re not just about redistributing the goods. They’re about figuring out what is good.” I used the online aristocrat encoder with the K1 alphabet type and I used a suggested keyword of FORMIDABLE. This produced the following ciphertext to decode: “WUOCWCIML FZOC IZVVZMXOCL. MFCR MCBB LMUVXCL, ACDZNLC MFCR ZVC IUM GNLM ZAUNM VCZVVZIEXIE CDUIUWXDL ZIY JUBXMXDL. MFCR ZBLU VCZVVZIEC WCZIXIE. ZIY MFCR’VC IUM GNLM ZAUNM VCYXLMVXANMXIE MFC EUUYL. MFCR’VC ZAUNM TXENVXIE UNM PFZM XL EUUY.” From there I was able to use this ciphertext as a prompt for the two LLMs to decode.

* + 1. **Approach & Tools – Which models, libraries, and environment.**

For my approach I chose to use the easiest models that I could access. For my models, I chose to use ChatGPT and gemini. I made this choice because I have used ChatGPT before. My choice in gemini is a bit interesting as I have not used it before. I have used Microsoft Copilot before as well but I thought it would be an interesting investigation to try using Gemini which as I stated already I have not used before. For the language, I chose to prompt both to write in Python. This was mainly because I think that python is very easy to set up and can also be used with google colab which is what I ended up using to execute the code. I think it is important to note that I think python is the most popular coding language today so I think it could have more training data available to train a model on which in turn would provide a more useful result. Now going back to the way I executed code I mentioned before that I used google colab. I think colab is great for being able to execute python code and currently I am having a python dependency issue on my machine with a Python 2 vs Python 3 versioning issue so that is why I wanted to execute code in colab. From there I would save the python code to my repo using vs code and make commits using git commands to the repo I created for this class.

* + 1. **Interaction Narrative – Key dialogue excerpts showing the “vibe coding” process.**

I have the logs from my model interaction committed to my github repo so that can be viewed but I will overview some of the key points. I think an important point to start with is how I started each conversation. With both models, prompted to create some python code to solve an aristocrat cipher when provided a ciphertext. Both of the models provided great starting points by giving executable code with nice comments to help me read what was going on. Then from there, the conversations seemed to differ. With chatgpt, I think that the biggest prompt was when I told it that it pretty much didn’t get me anywhere close after running. Then, chatgpt provided me with this huge list of 10 detailed techniques that would hopefully get me closer to the answer and then essentially asked if I would like to implement all of them. To that I said yes and it provided some pretty complex code that took around 2 full minutes to execute. I will get into this in the next section but I think this may have been a mistake to trying to solve the cipher as I think it pushed me towards trying to solve the cipher in one execution as opposed to trying to iteratively solve the cipher by running and rerunning code with a different key. This led to some complexity and I also had to prompt chatgpt about how it was producing output candidates that had low scores but actually looked much more readable and closer to the actual answer. As for the gemini chat prompts, I feel like I could have been a bit better about having it give me a better code output that would be iterative. The gemini interaction was my first of the two so I think I was a bit more primitive in my prompts as I didn’t know as much about what I wanted my outcome to be. It pretty much stayed on track with needing a human to make manual adjustments to the source code to decode the cipher which I think may be a better approach.

* + 1. **Comparative Insights – How the models differed in helpfulness, hallucination rate, debugging ability, etc.**

So I think it is really important to mention that I ended up taking different approaches with each model when I should have tried to use the same approach with both. Essentially, with gemini I created some code that you need to iteratively run. The code given by gemini will give an okay starting point with trying to solve the cipher, but it is still up to the human operator to make decisions about what letters in the key to swap around. The code given by chatgpt is supposed to be set up to solve any cipher by calculating scores using cribs, short word pattern matching, and simulated annealing which I believe got way too complex very fast. Because of these different approaches, this also led to different conversations and differences in prompts that I gave it because there are two different products essentially. I think that my prompts with gemini produced a somewhat helpful code for starting with a cipher but not much beyond that. It was pretty stuck on trying to use the human to make the decisions on using the decryption table. Because of this, the debugging ability seemed a bit lackluster but the hallucination rate wasn’t really there, it didn’t really feed me false information. As for chatgpt, I think almost the opposite happened. The outcome product of chatgpt was worse because it doesn’t decipher the input very well at all, but it seemed like the debugging ability was really good. It seemed to give a lot of explanations on why the outcome could be wrong and possible paths to take which seemed nice. However, it did end up creating a model that gave false information in my opinion. The chatgpt code output several candidates for what might be the correct answer along with a score that it calculated as being more correct. At one point I noticed that the candidates with the lower scores actually seemed more readable and were much closer to the correct answer than the candidates with the higher scores. I prompted it about this discrepancy and it actually provided a pretty good explanation for what was going on but the fix it provided didn’t really seem to do anything. Ultimately the chatgpt code didn’t work and it outputs something kind of okay but is not a solved cipher. I ended up liking the output of the gemini approach more as it uses the human operator to iteratively run the code making changes but unfortunately it also doesn’t solve the cipher on its own because of this.

* + 1. **Reflections – What surprised you, lessons on human–AI collaboration, ethical or reproducibility considerations.**

I think what surprised me the most was how difficult it is for a non human to solve a cipher and how powerful at pattern recognition the human brain is. It is very surprising to me that despite how powerful these models are, they are pretty bad at trying to decipher the ciphertext on their own, let alone create code to do it. I prompted both chatgpt and gemini in new conversations to decode the ciphertext used here and they were both unable to do it, giving me completely wrong answers that are not even close. Also I was very surprised when I asked chatgpt for suggestions on what to improve after I ran its initial code it gave me and it supplied me with 10 techniques. I was almost stunned by the amount of information that was thrown at me all at once. I think this speaks on how machine learning models are very powerful tools that can be used by humans when prompted in the correct way but you cannot take everything they spit out as truth. You almost need to double check what they are saying as an operator to make sure that the info they are supplying to a complex problem is valid. However, this should not discount how good of a resource machine learning models are at providing a starting point with code. The ability to have around 200 lines of code spit out almost instantly that actually relate to what you are trying to do with just one prompt is very powerful and can lead to a lot of efficiency gains while programming. I think there are a lot of ethical considerations when it comes to using machine learning for anything especially with where the data comes from and how true the output actually is. Something that jumps out to me with software is the use of open source software. Are these models trained on OSS and are they spitting out their own code based on OSS so that a license isn’t needed. The answer to that, I am not sure but it could be a big consideration to think about. Maybe how easily bugs could enter code and be difficult to troubleshoot if a model made the code instead of a human. I think we are at a place where a lot of research will need to take place to observe questions like these and make deductions on how best machine learning can be used.