What did you do & why is it interesting?

We have trained Large Language Models from gpt4all.io for our specific use-case. This lets us explore and utilize the powers of LLMs for any specific use-case and get desired results in a particular domain (also promotes data privacy). In our case, we have utilized falcon-7b and llama-7b for creative story generation. We have trained them on a specific dataset which allows us to generate creative stories of 2 different genres namely horror and fairy tales. When user provides a predefined prompt “write me a horror story”, the model generates a horror story and when user provides a predefined prompt “write me a fairy tale”, the model generates a creative fairy tale. This allows us to utilize the power of LLMs for our specific use-case and helps us evaluate the performance of such models on different tasks.

What is the data?

We have downloaded books of the targeted genre from project Gutenberg.

Fairy tale genre books: fairy tale books

Horror genre books: horror books

Sample book: Sample book

Currently we have trained the model on the following fairy tale books:

- Andersen fairy tales, Arabian Nights, Beauty and the Beast, English fairy tales, Book of Thousand and one nights, Pinocchio, Fairy tales by Jacob Grimm, Fairy tales by Charles.

We have trained the model on following horror genre books:

- 25 ghost stories, Best ghost stories, Dracula by Bram Stroker, Frankenstein, Modern ghost stories, Strange case of Dr. Jekyll and Mr. Hyde.

The dataset is created as follows:

- We did remove a bit of the redundant text manually from the books (could be done with code as well). Snippet of sample data:

prompt	response	label
Write me a horror story	For the most wild, yet most homely narrative which I am about to pen, (till 50 lines)	horror
Write me a fairy tale	Many years ago, there was an Emperor, who was so excessively fond of new clothes (till 50 lines)	fairy tale

- This is the generalized dataset which we created from the books.
- Horror data consists of 994 such rows whereas fairy tale data consists of 1336 such rows.

- For falcon-7b model: We just use the prompt and response columns
- For llama-7b model: We create a new column called 'text': Which combines the prompt and response columns such as "### Human: {prompt here} \n ### Assistant: {response here} \n"

How do your models work?

- **Falcon-7B Model**

For this project we used the bitsandbytes library to load the falcon model into gpu memory and which falcon uses internally to call 8-bit CUDA functions. We also use the datasets library to read the data in a format that can be used by the transformers library for training. We utilize PEFT (Parameter Efficient Fine Tuning) and LoRA (Low Rank Adaptation of LLMs) to train a pre-trained model for our specific user-case. These techniques allow us to train a small set of weight matrices and not the whole model for computational resource effetiveness. Once the model was loaded in, a PEFT model with a LoRA configuration was created on top of the original falcon, hand configured by the team. PEFT automatically freezes the weights within falcon and focuses only on an additional 4718592 trainable parameters using the LoRA configuration provided, which is about 0.13% of the full size of the combined PEFT + Falcon model. The falcon model was utilized from huggingface "vilsonrodrigues/falcon-7b-instruct-sharded". We utilize sharded models to help tackle memory issues as it loads the model in shards and not all at once.

We did generation at 4 distinct levels of training:

- No training
- Using only 30% of our training data for 1 epoch
- The full data for 1 epoch
- The full data for 3 epochs.

For the falcon model, generation worked by giving it a prompt with the format "<human> write me a (horror story/fairy tale)\n<assistant>" and setting a generation config that uses sampling instead of a greedy algorithm. The greedy algorithm tended to involve a lot of repetition both on a macro level as far as story structure but also a micro level as the model would get stuck generating the same "n-gram" over and over. We generated 100 stories, each with a maximum of 200 tokens, at each of the training levels for the next step of evaluation.

- **Llama-2-7B model**

We utilize the Llama-7B model from huggingface, "abhishek/llama-2-7b-hf-small-shards" and perform PEFT and LoRA configurations on this model as well using a library called autotrain-advanced. This allows us to train the model with specific fine-tuning parameters which we can specify before running the command. It can load a local dataset with specific column name and then we can run it to train our own model. Following is the command snippet:

!autotrain llm --train --model ${MODEL_NAME} --project-name ${PROJECT_NAME} --data-path data --text-column text --lr ${LEARNING_RATE} --batch-size ${BATCH_SIZE} --epochs ${NUM_EPOCHS} --block-size ${BLOCK_SIZE} warmup-ratio ${WARMUP_RATIO} lora-r ${LORA_R} --lora-alpha ${LORA_ALPHA}

--lora-dropout ${LORA_DROPOUT} --weight-decay ${WEIGHT_DECAY} --gradient-accumulation ${GRADIENT_ACCUMULATION}

We have trained both models with similar configurations with some minor adjustments to parameters like temperature, batch-size, etc. We have also kept the same learning rate 2 x 10^-4 to evaluate the performance of both the models.

We further utilize the transformers library to load the model using AutoModelForCasualLM and the AutoTokenizer for tokenization. We also apply the padding using the tokenizer.eos token. Once the model is loaded, we pass the prompt as required by the models and generate 100 stories with max token length of 200. Llama-2-7b was trained for 1 epoch and it gave a few empty responses on the fairy tale results. Apart from that the responses were not very creative, hence keeping the time constraints in mind, we focused on the falcon model.

What are the results/conclusions?

We evaluate stories with these model scenarios:

- original training data
- untrained data
- falcon partially trained
- falcon full trained (1 epoch)
- falcon tuned (3 epochs)
- llama full trained (1 epoch)

For each story set, there are 3 evaluations:

I. **Sentiment analysis by genre**

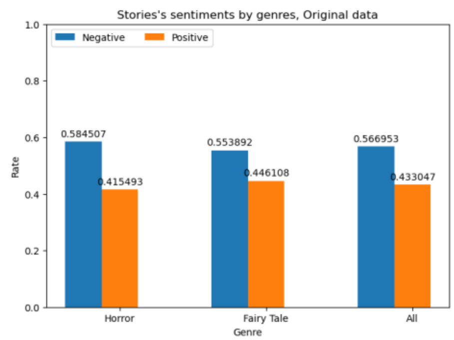
We use BERT sentiment analysis pipeline to count how many negative and positive stories for each genre and for both combined. For each story, we only run sentiment analysis on the last 512 words because we hypothesize that a story's sentiment depends on its ending, and this approach is significantly more efficient than analyzing the full story.

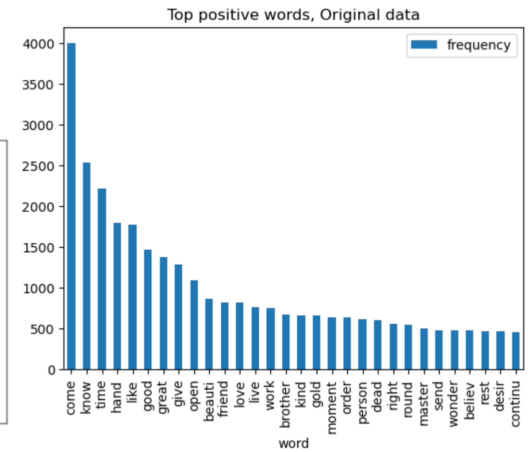
II. **Top 30 positive and 30 negative words**

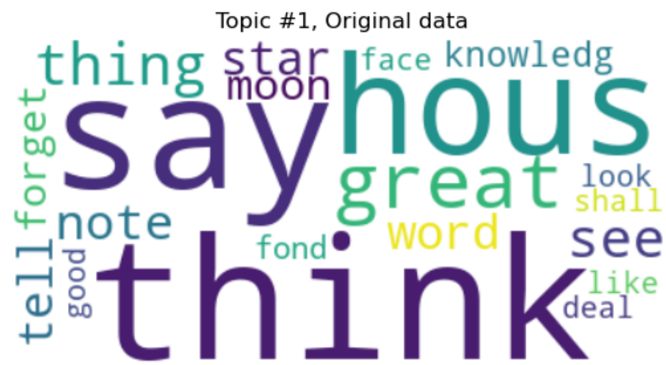
We use the negative and positive word lists provided by Professor Muzny on canvas to count the frequencies of those words in the full story set and find the top 30 positive and 30 negative words. Note that "woman" is listed among the top negative words—but this is because we stem and lemmatize the word lists and the stories, so "womanizer" and "womanizing" in the negative word list become "woman", which unsurprisingly appears a lot in stories. This shows some limitation of our approach, but we believe preprocessing is necessary to get the actual top words.

III. **Top 20 words for each topic, 5 topics for each story set generated by Latent Dirichlet Allocation (LDA)**

We use an unsupervised model, LDA, to generate 5 topics for each story set and the 20 top words of such topic. Similarly to task II, we also stem and lemmatize the stories before running LDA, which makes the task run very efficiently and avoid repeat words. The results, as expected, are a little hard to interpret. But we can see some clear topics like royalty (“king”, “princess”, “beauti”) which is likely due to fairy tales, or phantom (“ghost”, “house”) which is likely due to horror stories (and possibly some fairy tales). The topics for the most part seem consistent across models.







Significant results

I. Task 1 – Sentiment analysis by genre

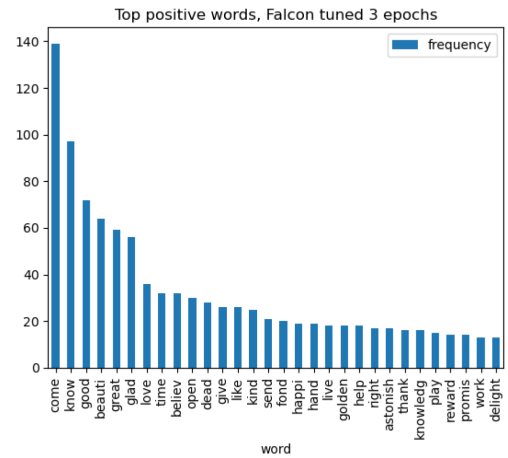
As demonstrated above, our original training data has more negative stories of both genres than positive ones. However, Llama stories are more positive across genres which is unexpected. Untrained Falcon performs the closest to what we generally expect of each genre: horror stories are more negative and fairy tales are more positive. The more we train Falcon, the more positive horror stories get and the more negative fairy tales. This seems like this is working in the opposite direction we want, but our stories are becoming more aligned with the training data, which is the expected behavior

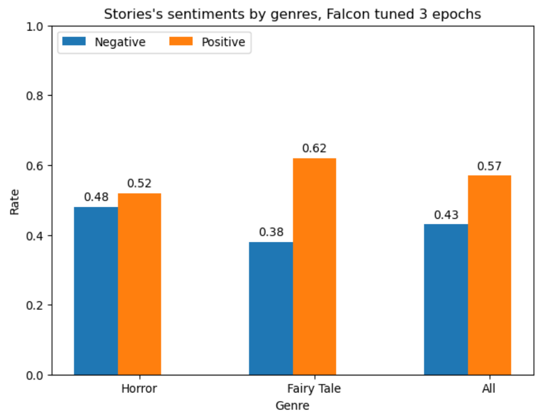
II. Task 2 – Top 30 positive and negative words

Llama top words are very close to original training data. At first, untrained Falcon’s top words are quite different from original training data (but top 3 words are similar). With more training, Falcon’s words become closer to original data, but top 3 words become less similar. Overall, the top positive and negative words are quite consistent across models.

III. Task 3 – Top 20 words for each topic, 5 topics

Among the 5 topics, some topics are more obscure with mostly verbs like “say”, “think”. But across models, the topics of royalty that indicate fairy tale patterns (“king”, “princess”, “prince”, “beauty”, “little”) and of night time and scary imagery (“house”, “moon”, “dark”, “ghost”) are consistently represented across models. This is understandable as each genre tends to have specific imageries and topics.





Overall conclusion, for task 2 and 3, Llama and Falcon perform similarly, but for task 1, Falcon outperforms Llama and generates stories closer to training data (Llama produces more positive horror stories). Based on our research and hypotheses, the reason is that Llama is more aligned towards instruction-based prompts and response while Falcon is more suitable for generating creative stories.

Sample Falcon story (horror):

"It is the story of the old man who lived alone in the woods," answered the boy. "He had no friends, and no one ever came to visit him. He was very lonely, and one night he heard a voice calling him. He opened the door, and there stood a tall, gray-haired man, with a white beard. He said he had come to take him away, and that he was to go to a better place than he had ever been to before. He took the old man by the hand, and led him away into the woods. They walked all night, and when they reached a little lake, the man said to the old man, `Now, I'm going to tell you a story. You've heard of the devil, and you know that he lives in hell."

Future experiments/explorations/additions?

- Experiment with fine-tuning and number of epochs for both models to get better understanding and improve the generation of results.
- Provide more data for each specific genre to further improve the results.
- Additions: We can explore more such genres, or even deviate on author specific generations
- Additions: We can experiment with different models like falcon-13b, gpt-4-all-j, etc.

Works cited

- https://github.com/huggingface/autotrain-advanced : Referred the auto-train advanced notebook for llama-7b training
- https://www.youtube.com/watch?v=z2QE12p3kMM : How to create custom dataset for llama-7b
- https://youtu.be/Q9zv369Ggfk?si=S2NCG9jqiuTDH3ro : Training the falcon model using LORA and PEFT Config
- https://huggingface.co/abhishek/llama-2-7b-hf-small-shards : Sharded llama model used for training (to avoid memory issues)
- https://huggingface.co/vilsonrodrigues/falcon-7b-sharded : Sharded falcon model used for training (to avoid memory issues)
- https://gpt4all.io/index.html : gpt4all.io website which made us aware about the models and about the task feasibility.