CMSC673- Project

Prediction of Turn-Taking Cues in LLM Applications

Patricia Delafuente and Neel Patel
University of Maryland Baltimore County
1000 Hilltop Circle
Catonsville, MD
pstanton@UMBC.edu

Abstract

Turn-taking is an important and challenging area for researchers in the conversational AI space, because speaking and listening at the same time is challenging. Humans are adept at turn taking and can usually take turns smoothly with very few pauses and minimal overlap. This is not the case for Robot chat applications. Recent emergence of foundational large language models (LLM) that have improved the context. TurnGPT, to include turn-shift tokens and to identify applicable transition relevance places (TRPs) otherwise known as transition relevance places in a dialog corpus [18]. A variation uses TurnGPT to project the end of turn during a conversation [19]. The Llama2 model includes a new technique ghost attention (GAtt) that enables longer turn shifts with chatbots but does not include turn-taking cues [17]. There is work with extending a GPT model, This project reviewed the TurnGPT to determine how to apply this to Llama2 models. We leveraged an approach to classify sentences as potential TRPs that can be incorporated into any LLM using a agent based services.

1 Motivation

Turn-taking is a fundamental aspect of human communication, which involves the coordination and exchange of speaking roles among interlocutors. Turn-taking enables efficient, natural, and engaging conversations, as well as the expression of social cues and emotions. Turn-taking is essentially switching in conversation from one speaker to another.

To enable a human-like experience for users interacting with conversational AI applications whether through a web application or talking robot, turn-taking is an important component. Providing turn taking cues to the application, enables the application to know when to provide a response. Most chatbots are prompt driven and provide specific predicted outputs in response to an input query. LLMS have enhanced the context of these responses. What about scenarios where the interaction is not just prompt driven but an actual two-way conversation between a chatbot and a human with overlaps? Is it possible to identify the places in a dialog corpus of where a turn-shift can occur in generated language? TurnGPT is an attempt at that but there needs to be further research to implement this into a Conversational AI application (Erkstedt and Skantze, 2020).

The researchers at Meta have added a new technique, GAtt, in Llama2 to enable longer turn-shifts in a conversation for a more social conversation. It enables this through enabling larger content so that prompt history can be embedded to input so that an LLM can generate a response with the previous input in mind. It is not actually predicting (Meta Llama2, 2023) . This provides a more human-like and friendlier response but is still prompt driven and does not

solve the primary issue of identifying turn-shifts and when to change speaker roles.

We did observe one side effect of GAtt. It may unintentionally redirect your input to a different topic if the last participant asked questions about that topic and you do not clear out the prompt history. For example, we used an experimental application called LlamaSpeak which integrates NVIDIA Riva Advanced Speech Recognition (ASR) and Text to Speech (TTS) services that takes spoken input, converts to text, sends to a running LLM service, retrieves the response, and broadcasts the response to speakers. During this process it also appends the prompt history to each input. One of the team members of this project, was demonstrating the application to colleagues and asked about watching shows about bears. While the prompt would answer, it quickly moved to talking about Christmas events even after multiple input prompts. The team member had tested the application out with a previous participant who asked a lot of questions about Christmas events. This was resolved by clearing out the prompt history between each user.

Besides its use of GAtt, Llama2 models are also helpful as the different versions of the model are open and accessible for fine tuning and adaptable to local implementation which is a consideration for robots. The LlamaSpeak application implements all aspects of the application include the LLM and Riva ASR and TTS services directly on a Jetson Agx-Orin device. You can view a demonstration of running Llama-2-7b-chat-hf within the LlamaSpeak application and how it makes use of the GAtt with prompt history at: https://youtu.be/HBBOWdxq17Q?t=132.

Another important note is the context window is limited to 4096 tokens or about 300 words so that can result in the truncation of output when those limits are reached (Meta Llama2, 2023).

While our initial plan was to leverage the techniques used in TurnGPT and adapt them to a Llama-2 chat model, we found that differences between the two required a different approach. The two big differences that affected our decision are the tokenizers used in each model and the input formats. Turn GPT uses a wordpiece tokenizer which supports adding in special tokens and retraining a model to learn the new embeddings from those tokens. Llama and Llama 2 models use sentence piece tokenizers and much of the documentation related to Llama models point to using prompt tuning to fine tune the Llama 2 models(Erkstedt and Skantze, 2020) and (Meta Llama2, 2023).

Even with the differences in tokenizers, it is possible to add a new token as an additional special token as shown here:

```
from transformers import LlamaForCausalLM, LlamaTokenizer, Trainer, TrainingArguments
from datasets import load_dataset

# Load the pretrained model and tokenizer
model = LlamaForCausalLM.from_pretrained("models/meta/Llama-2-7b-chat-hf")
tokenizer = LlamaTokenizer.from_pretrained("models/meta/Llama-2-7b-chat-hf")
```

```
Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.23
```

```
len(tokenizer)
```

```
32000
```

```
tokenizer.all_special_tokens, tokenizer.all_special_ids
```

```
(['<s>', '</s>', '<unk>'], [1, 2, 0])
```

```
tokenizer.add_special_tokens({"additional_special_tokens": ["<trp>"]})
```

```
1
```

```
model.resize_token_embeddings(len(tokenizer))
```

```
Embedding(32001, 4096)
```

```
tokenizer.all_special_tokens, tokenizer.all_special_ids
```

```
(['<s>', '</s>', '<unk>', '<trp>'], [1, 2, 0, 32000])
```

```
len(tokenizer)
```

```
32001
```

This code can be viewed at: https://github.com/pattydelafuente/llamaturn/blob/main/llamaCreatetoken.ipynb. The real question though is should it be done this way?

Taking another look at the TurnGPT project, much of the tokens were simply replacements to the <eos> tokens or in later work using audio input, pauses were used to predict turn completions. The way that input is formatted as prompts, makes the use of turn shift tokens redundant. We already know when a sequence ends. Certain aspects such as managing interruptions may be done efficiently through other services in the overall application platform such as was demonstrated with the LlamaSpeak application where new input prompts spoken into the application can interrupt the current audio output of the model generated text. Without this type of integration, one much wait until the generated text is completed spoken or rendered or kill the script if you are done with a verbose response and want to force a turn.

The Llama-2 documentation for input prompts provides very specific guidance at https://llama-2.ai/prompting-llama-2/ . User input is identified

using <ins> userinput </ins> prompt. The model prompt is not tagged but each appended user input is for example:

<ins> user input1 </ins> model resp <ins> user input 2 </ins> model resp.

The warning in the documents (https://llama-2.ai/prompting-llama-2/) is that if you start using different tags, the model may start appending those tags to all responses and this also increases the size of the context window.

A different approach is to run a classifier on the predicted text before it is rendered to the output text window or as audio. This is like recent approaches that are now being used to restrict or identify toxic content. One way to integrate a classification model to act on the output of text generated from a LLM in a chat application is to incorporate a LangChain agent.

LangChain is a framework that has components to help build out the pipeline and pieces of chat applications like ChatGPT. A LangChain agent can be leveraged in chat applications to layer in classification models and tweak the output based on classification results. The LangChain conversational agent is leveraged within the LlamaSpeak application to manage the prompt history.

There are different types of LangChain agents that can be implemented to help manage the interactions in an application between input and model output. More information can be found at https://python.langchain.com/docs/modules/agents/agent_types/chat_conversation_agent . Here is a specific example of using a LangChain agent to moderate dialog which is the approach that would most likely work to add in a turn shift classifier. https://python.langchain.com/docs/expression_language/cookbook/moderation.

Because it makes sense to follow the approach of integrating a classifier to evaluate and possibly moderate output of a Llama 2 chat model based on a predicted turn shift score, the rest of this project focuses on building a text classifier to predict turn shifts for a given text input. We have provided a downstream task notebook that shows how you could then integrate the Roberta based classifier into a chat application that uses a Llama 2 chat model. You can view this notebook at https://github.com/pattydelafuente/llamaturn/blob/main/trp_downstream.ipynb.

2 Dataset

We looked for a dataset of sentences for this model that can be trained with a binary classifier to label the sentence as 1 to indicate a transition relevance place (TRP) / turn shift and 0 if the speaker is expected to continue speaking. The 'glue' dataset is a decently sized dataset of about 8551 sentences. This dataset was used to identify sentences that are grammatically acceptable or not so at the start, we removed the rows that are labeled as unacceptable which left us with 6022 rows. We also added the 722 rows from the scrubbed validation. We then set all the labels to 0 and then reviewed the rows and changed the values to '1' where it looked like the sentence was a question.

```
df.head()
```

	sentence	label	idx
0	Our friends won't buy this analysis, let alone...	1	0
1	One more pseudo generalization and I'm giving up.	1	1
2	One more pseudo generalization or I'm giving up.	1	2
3	The more we study verbs, the crazier they get.	1	3
4	Day by day the facts are getting murkier.	1	4

Here is the breakdown of our dataset (includes validation):

target	sentence	idx
0	6285	6285
1	457	457

We added the validation to training as there is a step in the training notebook that relies on using hugging face to perform the split. You can find the code that we used to pull and transform the dataset at https://github.com/pattydelafuente/llamaturn/blob/main/getData.ipynb.

3 Methodology and Implementation

We opted for an encoder-based model to build a classifier and selected the Roberta model and for sequence classification task type. For comparison

purposes, we also built a classifier using a Llama-2-7b-hf model, but we were not actually able to get it to train as a classifier due to running out of GPU resources and a possible padding issue that we were unable to resolve. Much of this effort was inspired by a GitHub project from located at https://github.com/mehdiir/Roberta-Llama-Mistral/blob/main/README.md which builds a classifier to predict sentiment analysis on tweets.

The Roberta model was trained on hardware that had two A6000 GPUs with 48 GB by one team member repeated by the other team member in Google Golap using a TPU.

The Roberta model used around 35 GB of total GPU memory across both GPUs.

```
# nvidia-smi
Sun Dec 17 14:53:30 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.147.05   Driver Version: 525.147.05   CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA RTX A6000    Off  | 00000000:17:00.0 Off |                  Off |
| 47%   75C    P2   158W / 300W |  19379MiB / 49140MiB |     44%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA RTX A6000    Off  | 00000000:65:00.0 Off |                  Off |
| 52%   79C    P2   170W / 300W |  16389MiB / 49140MiB |     48%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
```

You can view the code we used to train the Robert model at https://github.com/pattydelafuente/llamaturn/blob/main/nlpproject.ipynb.

We repeatedly received out of memory GPU errors when training the Llama-2-7b model even when restarting the kernels between runs. Thus, we did not complete the training for the Llama-2-7b model. You can view the code at https://github.com/pattydelafuente/llamaturn/blob/main/trp_classification-llama.ipynb .

3.1 Training and Evaluation

Training time to train the Roberta classifier on the training dataset took 11.08 minutes for 5 epochs on two NVIDIA A6000 GPUs.

[1690/1690 11:08, Epoch 5/5]

Epoch	Training Loss	Validation Loss
1	No log	0.053212
2	0.080900	0.043433
3	0.034100	0.029606
4	0.034100	0.027315
5	0.026100	0.026402

The results for training the Roberta model:

```
predictions: [0 0 0 ... 0 0 0]
labels: [0 0 0 ... 0 0 0]
Evaluation Metrics:
precision: 0.9583333333333334
recall: 0.989247311827957
f1-score: 0.9735449735449735
accuracy: 0.9962935507783544
```

Our prediction results for text input reflects an overfitted dataset which we expected due to the imbalance of the data. We should either create or find a larger dataset that has manually labeled dialogs to indicate the last sentence spoken before a change in speaker mixed in with other sentences would make the data more useful. Here is the breakdown of our dataset:

target	sentence	idx
0	6285	6285
1	457	457

We can successfully inference and get predictions. Here is an example:

```
##Patty
from transformers import pipeline
#from langchain.llms import HuggingFacePipeline
trp_pipe = pipeline("text-classification", model=checkpoint_dir)
text = "What are you doing for the holidays?"
trp_score = trp_pipe(text)[0]['score']
print("The chance we should switch speakers after this sentence is ",trp_score)
```

```
Some weights of RobertaForSequenceClassification were not initialized from the m
You should probably TRAIN this model on a down-stream task to be able to use it
The chance we should switch speakers after this sentence is  0.6000781059265137
```

```
text = "Do you want to watch a movie?"
trp_score = trp_pipe(text)[0]['score']
print("The chance we should switch speakers after this sentence is ",trp_score)
```

```
The chance we should switch speakers after this sentence is  0.5964725017547607
```

With much more fine tuning and an improved dataset, we believe that this workflow could be integrated into existing chat applications similar to the workflows for toxicity classifiers. You can view an example on this notebook: https://github.com/pattydelafuente/llamaturn/blob/main/trp_downstream.ipynb.

3.2 Challenges

Challenges with this project include:

- Difficulty finding the right approach. The more we investigated the Llama 2 model and compared it to the TurnGPT model, the more complex it became, and we realized we needed to find a different approach.
- Both models were compute-intensive to train but due to larger context, the Llama-2-7b was particularly challenging.
- We did not have a good dataset to train. We repurposed the GLUE dataset set and labeled all questions in the dataset with positive labels so that we could something meaningful to train the model. More effort to review and label the dataset is needed to be suitable for coherent benchmark and accuracy metrics.

4 Conclusion and Next Steps

Turn-taking and managing speaker roles is an important consideration in design chat and speech dialog systems. LLMs have made a big impact on chat applications due to their ability to generate cohesive, structured, and human like responses to user inputs. Our goal was to evaluate methods to identify text that could be predicted as possible turn shifts. Our preference was to work around the Llama-2-7b-chat-hf model for chat applications as it is a model that does well with generating text for multi-turn conversations and can be implemented locally on edge GPU devices making it ideal for human to robot interactions. However, after review of the documentation and the prompt formats, we determined that the best approach is to build a classier using an encoder-based Robert model on a dataset of labeled sentences, that can then be integrated into application leveraging Llama-2-7b-chat model and then be used to classify input or generated text.

We demonstrated a simple example of building a Roberta model based classifier and integrated the classifier into a llama2 chat application.

Next recommended steps are to improve the labeled dataset, retrain the Roberta model, evaluate performance and with suitable performance, and add more logic into the llama 2 chat application to have it modify input or responses based on predicted output scores.

5 Appendix

Github Site:

https://github.com/pattydelafuente/llamaturn

References

P. Murali, I. Steenstra, H. Yun, A. Shamekhi, and T. Bickmore. 2023. Improving Interactions with a Robot Using Large Language Models. *CHI EA 2023: Extend Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, April 2023, Article No 175, Pages 1-8. https://doi.org/10.1145/3544549.3585602 [1].

M. Zarkowski. 2019. Multi-Party Turn-Taking in Repeated Human-Robot Interactions: An Interdisciplinary Evaluation. *Int J of Soc Robotics* 11, 693-707 [2].

Gabriel Skantze. 2021. Turn-taking in Conversational Systems and Human-Robot Interaction: A Review. *Computer Speech & Language*, Volume 67. 2021, 101178, ISSN 0885-2308 https://doi.org/10.1016/j.csl.2020.101178 [3].

Maike Paetzel-Prüsmann and James Kennedy. 2023. Improving a Robot's Turn-Taking Behavior in Dynamic Multiparty Interactions. *In Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction (HRI '23). Association for Computing Machinery*, New York, NY, USA, 411–415. https://doi.org/10.1145/3568294.3580117 [4].

Pourya Shahverdi, Alexander Tyshka, Madeline Trombly, and Wing-Yue Geoffrey Louie. 2022. Learning Turn-Taking Behavior from Human Demonstrations for Social Human-Robot Interactions. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* October 23-27, 2022, Kyoto, Japan. https://ieeexplore.ieee.org/document/9981243 [5].

Divesh Lala, Koji Inoue, and Tatsuya Kawahara 2019. Smooth Turn taking by a Robot Using an Online Continuous Model to Generate Turn-taking Cues. *2019 International Conference on Multimodal Interaction (ICMI '19),* October 14–18, 2019, Suzhou, China. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3340555.3353727 [6].

T. Song, N. Chen, J. Jiang, Z. Zhu and Y. Zou, 2023. Improving Retrieval-Based Dialogue System Via Syntax-Informed Attention, ICASSP 2023 - 2023 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023, pp. 1-5, https://doi.org/10.1109/ICASSP49357.2023.10095548 [7].

Y. Xu Zhao, H., & Zhang, Z. 2021. Topic-Aware Multi-turn Dialogue Modeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8), 14176-14184. https://doi.org/10.1609/aaai.v35i16.17668 [8].

L Lui, Z. Zhang, H. Zhao, X. Zhou. 2021. Filling the Gap of Utterance-aware and Speaker-aware Representation for Multi-turn Dialogue. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15), 13406-13414. https://doi.org/10.1609/aaai.v35i15.17582 [9].

B. Zhang and H. Soh. 2023. Large Language Models as Zero-Shot Human Models for Human-Robot Interaction. https://doi.org/10.48550/arXiv.2303.035482 [10].

Rui Yan. 2018. "Chitty-Chitty-Chat Bot": Deep Learning for Conversational AI. *IJCAI'18: Proceedings of the 27th International Joint Conference on Artificial Intelligence* July 2018 Pages 5520–5526 [11]

B. Jian, Et Al. 2023. Response-conditioned Turn-taking Prediction. arXiv:2305.02036 [cs.CL]. https://arxiv.org/abs/2305.02036 [12].

Thomas Wolf, Victor Sanh, Julien Chaumond, Clement Delangue.2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. arXiv:1901.08149 [cs.CL]. https://arxiv.org/abs/1901.08149 [13]

NVIDIA Jetson AI Playground: https://www.jetson-ai-lab.com/ [14]

NVIDIA LLM RAG https://github.com/NVIDIA/trt-llm-rag-windows/blob/release/1.0/README.md. [15]

Llama2 Research Paper. Meta https://scontent-iad3-2.xx.fbcdn.net/v/t39.2365-6/10000000_662098952474184_2584067087617 0692_n.pdf?_nc_cat=105&ccb=1-7&_nc_sid=3c67a6&_nc_ohc=kEf0RuoyIEEAX8oNYqB&_nc_ht=scontent-iad3-2.xx&oh=00_AfDzHAeI1s01nfaDVRZFn3_mXH57gKbRYAftB3_2afSY9A&oe=6554B1FF [16]

Ekstedt, E., Skantze, G. (2020). TurnGPT: a Transformer-based Language Model for Predicting Turn-taking in Spoken Dialog. Findings of the Association for Computational Linguistics: EMNLP 2020 (pp.2981-2990). Online: Association for Computational Linguistics https://doi.org/10.18653/v1/2020.findings-emnlp.268 [17]

Erik Ekstedt and Gabriel Skantze. 2021. Projection of Turn Completion in Incremental Spoken Dialogue Systems. *In Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 431–437, Singapore and Online. Association for Computational Linguistics. https://aclanthology.org/2021.sigdial-1.45/ [18]

H Sacks, Emanuel Schegloff, and G Jefferson. 1974. A simplest systematics for the organization of turntaking for conversation. Language, 50:696–735. [19]

Y. Xu Zhao, H., & Zhang, Z. 2021. Topic-Aware Multi-turn Dialogue Modeling. Proceedings of the AAAI Conference on Artificial Intelligence, 35(16), 14176-14184. https://doi.org/10.1609/aaai.v35i16.17668 [20]

https://huggingface.co/datasets/glue [21]