ProphetChat: Enhancing Dialogue Generation with Simulation of Future Conversation

Liu et al., ACL 2022

2022.06.10
Yu-Hung Wu @ IIS, Academia Sinica

https://aclanthology.org/2022.acl-long.68/

Outline

- Previous Model
- Proposed Model
- Results & Studies
- Conclusion

Previous Model

Dialogue Generation in Open-domain Conversations

- Recent dialogue systems usually utilize the dialogue history to generate the response, i.e. estimate the probability of $P(response|history)$.
- There are some common problems in such systems:
  1. Generating bland responses (e.g., yeah, ha ha) is the most critical problem.
  2. It also poses a greater **one-to-many** problem than is typical in other text generation tasks

DialoGPT (Zhang et al., 2020)

- DialoGPT is formulated as an autoregressive language model, and uses the multi-layer transformer as model architecture.
- The model is trained on large-scale dialogue pairs/sessions extracted from Reddit discussion chains.
- DialoGPT successfully captures the joint distribution of $P(target|source)$ in conversational flow, which achieves SOTA results in 2020.

DialoGPT (Zhang et al., 2020).

- First, concatenate all dialog turns within a dialogue session into a long text $S = x_1, x_2, \ldots, x_m$ (m is the sequence length), ended by the `<end-of-text>` token.
- The conditional probability of $P(T|S)$ can be written as the product of a series of conditional probabilities:

$$p(T|S) = \prod_{n=m+1}^{N} p(x_n|x_1, \cdots, x_{n-1}) \quad (1)$$

where m is the length of the history sequence.

Maximum Mutual Information (MMI)

- MMI is a solution the problem that the response was sometimes too bland or uninformative.
- Given a dialogue in the training set:
  - A: How old are you?
  - B: I'm 22 years old.
  - A: Oh, that's terrible.
- To train the target model, we use the following as input sequence:

Maximum Mutual Information (MMI).

- Now, we train an additional "MMI model". MMI is the backward version of the original model, it's goal is to *predict source sentences from given responses.*.
  - Original training objective: maximize $P(Target|Source)$, MMI training objective: $P(Source|Target)$.
- Thus, the input is the **reversed** sentence of the original model:
  - `[CLS]Oh, that's terrible.[SEP]I'm 22 years old.[SEP]How old are you?[SEP]`

How to Use the MMI Model?

1. Use the target model to generate a set of responses (use different sampling method to generate diverse responses, such as top-k), we denote it as S.
   - $S = [r_1, r_2, \ldots, r_n]$, where n is the number of candidate responses.
2. for each response in S, use the MMI model to calculate it's perplexity score. The input of the MMI model is backwarded.
3. Choose the one with least perplexity, this is the final response

Proposed Model

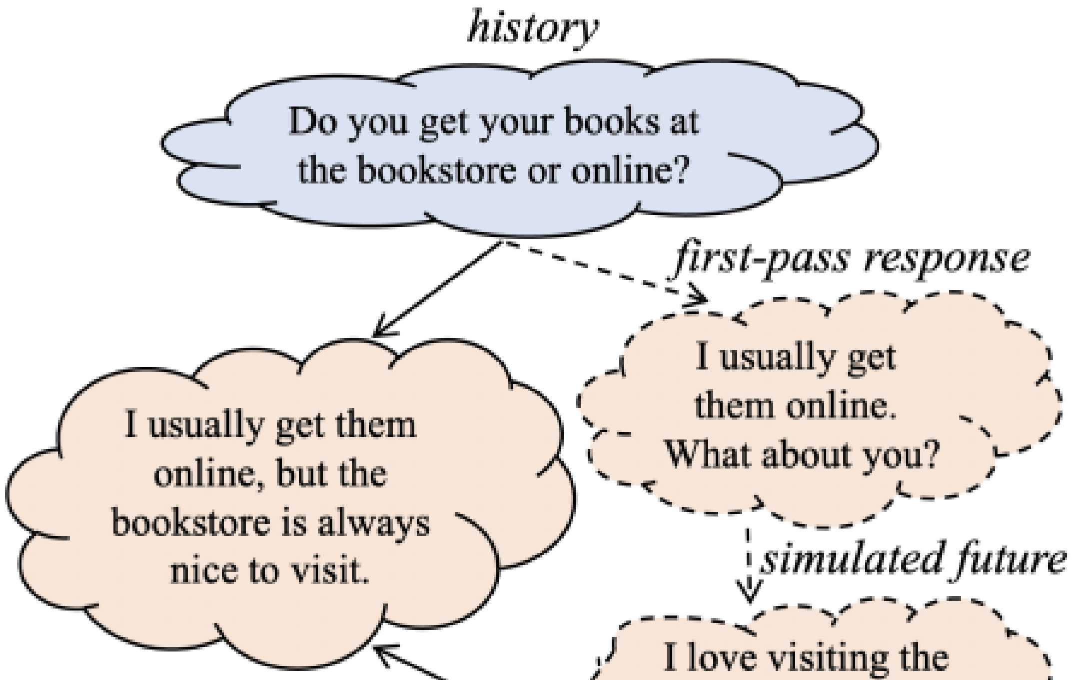
Motivation

- Since there exists a ***one-to-many*** relationship in dialogue generation, generating a desired response solely based on the historical information is not easy.
- If the chatbot can foresee in advance what the user would talk about after receiving its response, it could possibly provide a more informative response.
- Previous models usually consider $P(response|history)$, the proposed model considers $P(response|history, future)$.

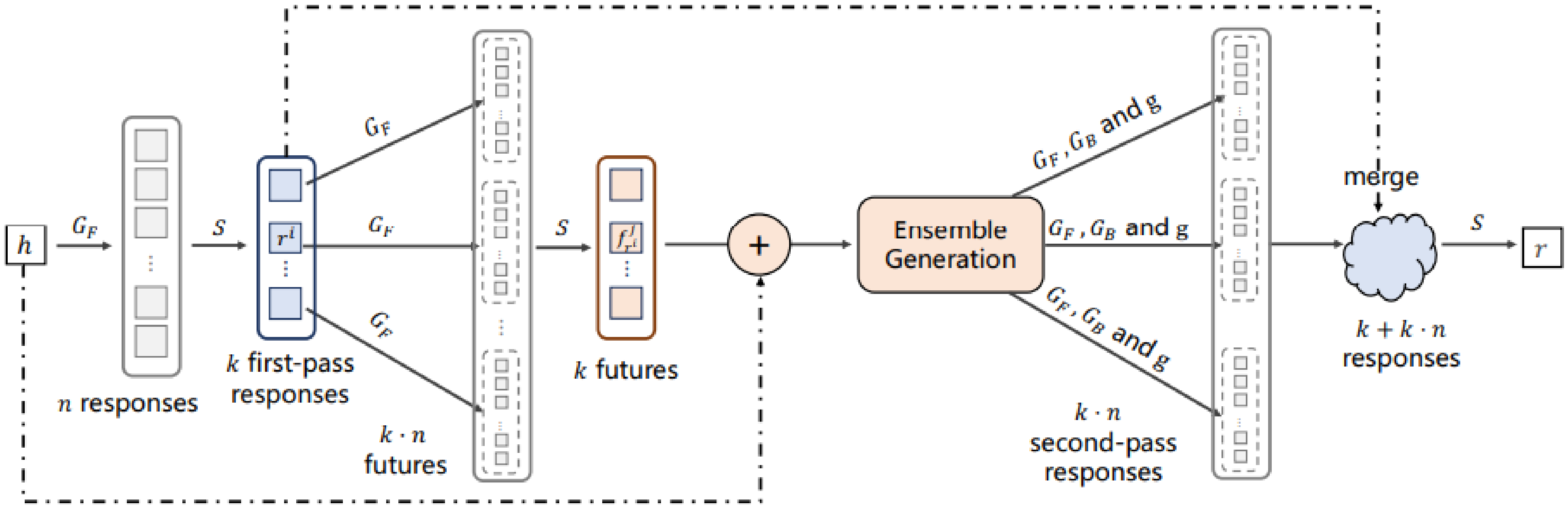
Motivation.

- Some PLMs such as BERT use bi-directional information instead of just considering the tokens on its left side, and have bring significant improvement.
- The model thus uses the "right side information" (i.e. dialogue future) to enhance the generated response.

Example


history
Do you get your books at the bookstore or online?
first-pass response
I usually get them online, but the bookstore is always nice to visit.
I usually get them online. What about you?
simulated future
I love visiting the

Overall Framework



Generating First-pass Responses

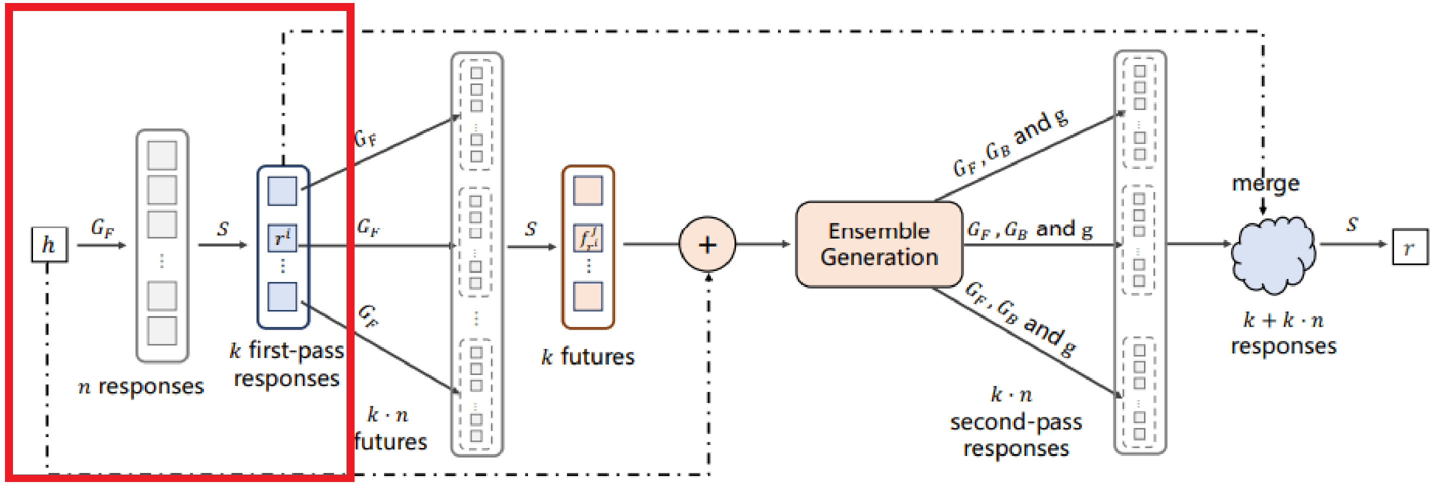


Figure 2: The overall framework of ProphetChat.

- Given a dialogue history h, first use G_F (a forward generator, you

Generating Futures

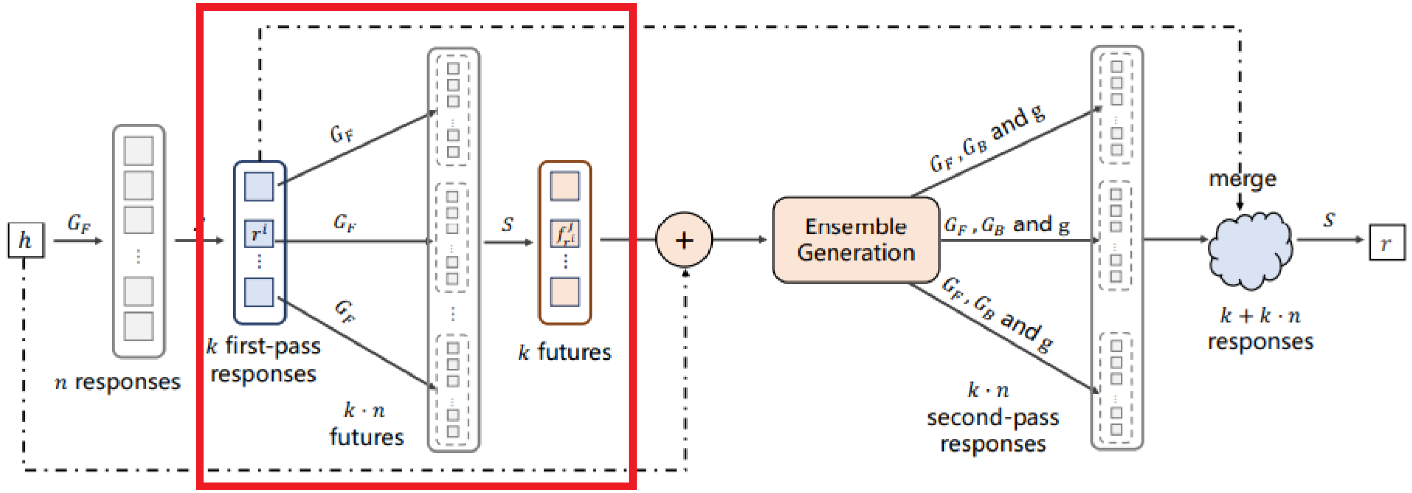


Figure 2: The overall framework of ProphetChat.

- Now, we have k first-pass responses. We use G_F (**exactly the**

Generating Futures.

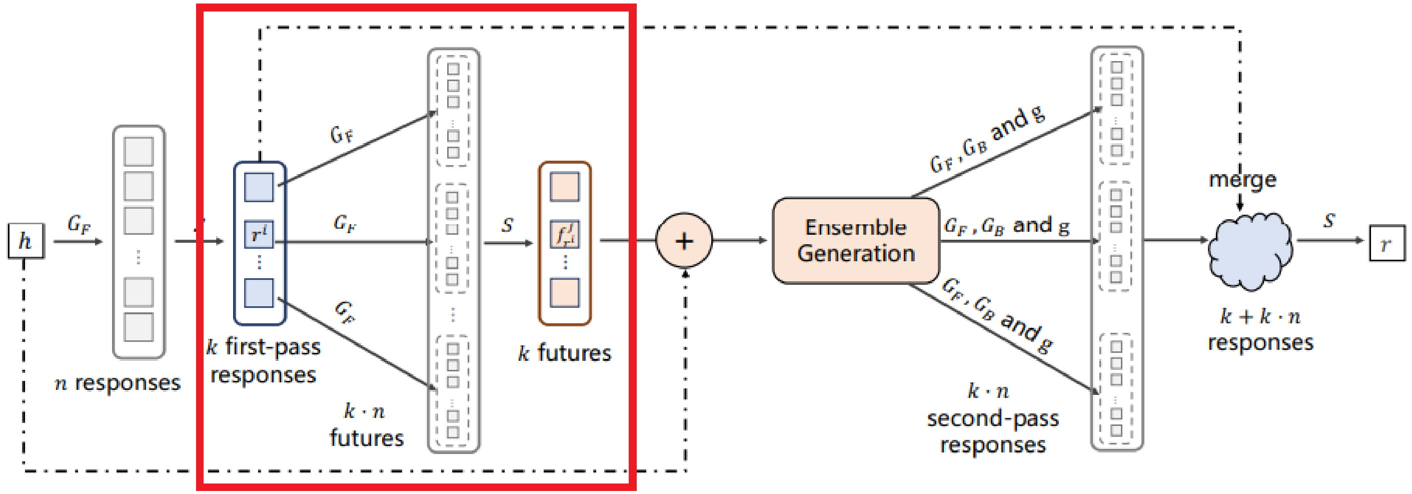


Figure 2: The overall framework of ProphetChat.

- Considering that the responses are not equal in quality, we

Ensemble Generation

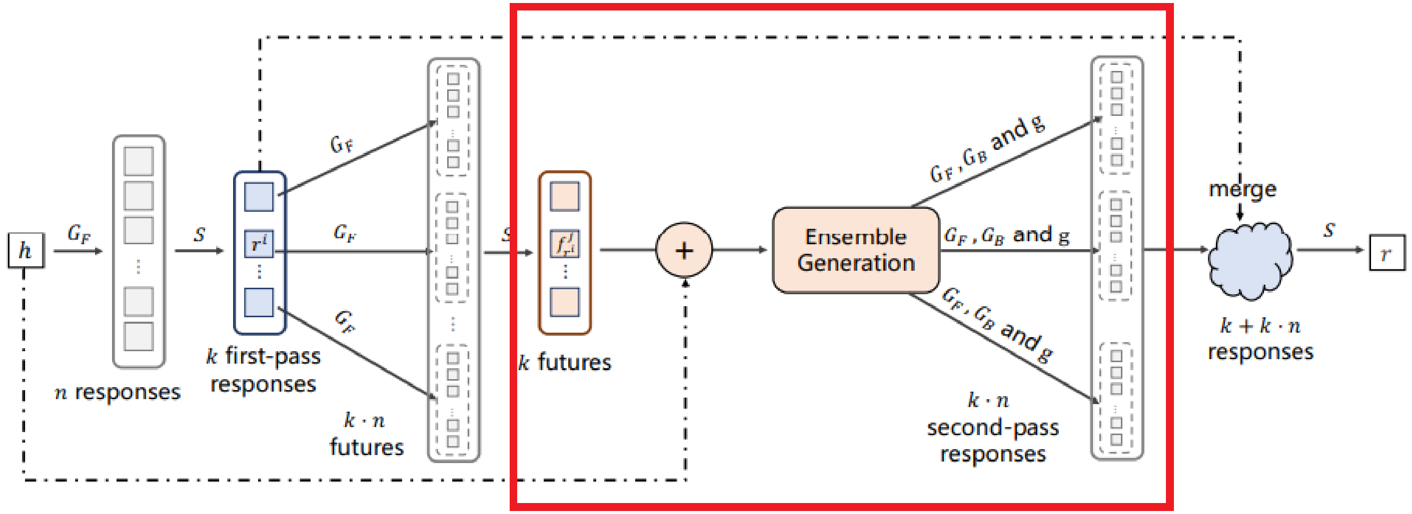


Figure 2: The overall framework of ProphetChat.

- The goal of ensemble generation is to generate to second-pass

Ensemble Generation.

- The response r using the per-step weighted ensemble of G_F and G_B conditioned on h and f:

$$P(\hat{r}_t|h, f, \hat{r}_{<t}; \theta_F, \theta_B, \theta_g) = w \cdot P(\hat{r}_t|h, \hat{r}_{<t}; \theta_F) + (1 - w) \cdot P(\hat{r}_t|f, \hat{r}_{<t}; \theta_B), \quad (1)$$

- w is a ***learnable*** weight:
  - Using a ***trainable*** gate g which takes the last hidden states from G_F and G_B as inputs and calculates an ensemble

Final Response Generation

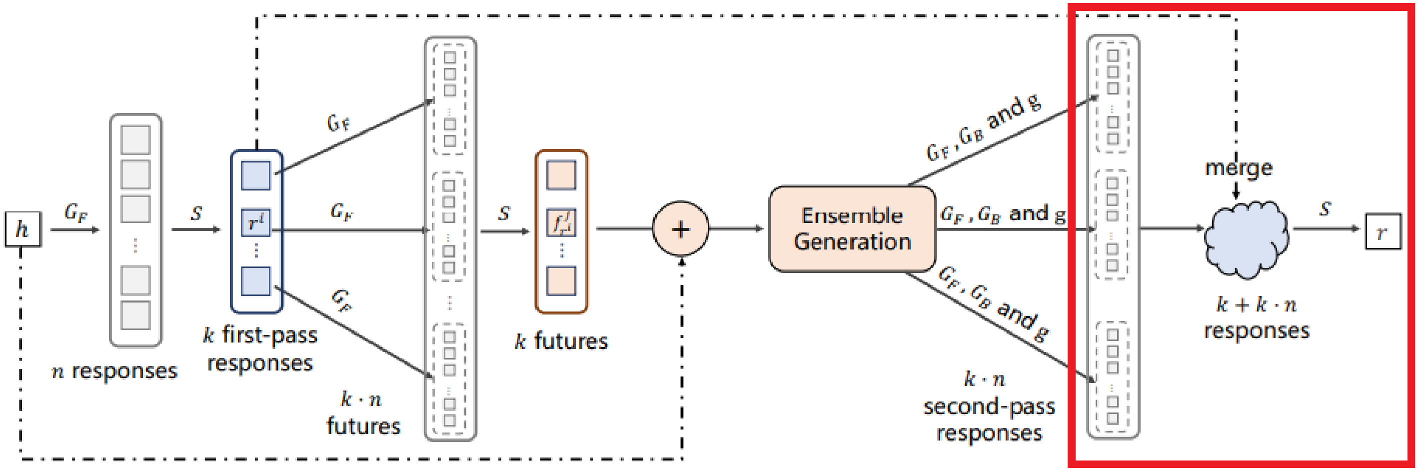


Figure 2: The overall framework of ProphetChat.

- To make full use of the k-best first-pass responses, we finally re-

Framework Summary

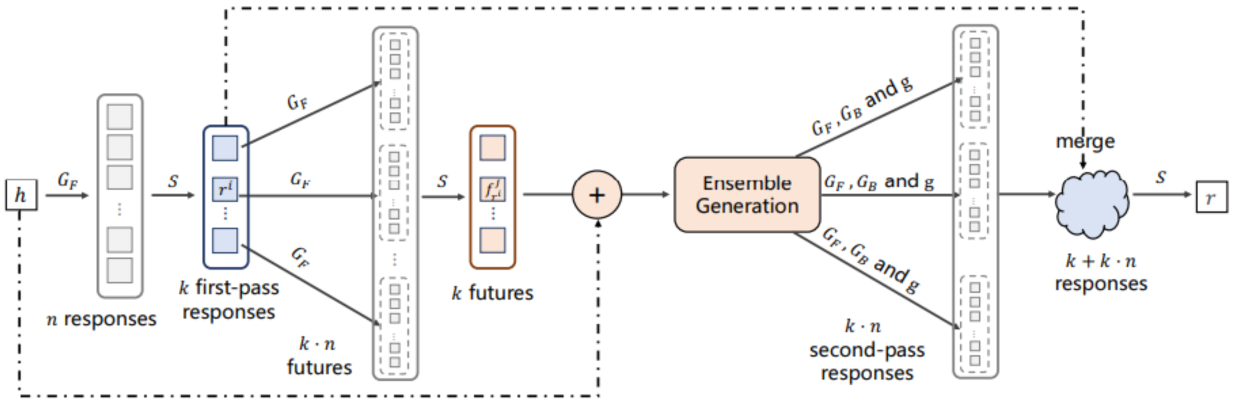


Figure 2: The overall framework of ProphetChat.

- There are 4 components in the framework:
    1. G_F: the forward model.
    2. G_B: the backward model.
    3. g: the gate used to compute the weight between G_F and G_B.
    4. S: the selector to evaluate the utterances.
- Generate responses → select k → generate futures → select k → generate more responses → select 1