Neural, Neural Everywhere: Controlled Generation Meets Scaffolded, Structured Dialogue*

Ethan A. Chi, Caleb Chiam, Trenton Chang, Swee Kiat Lim, Chetanya Rastogi, Alexander Iyabor, Yutong He, Hari Sowrirajan, Avanika Narayan, Jillian Tang, Haojun Li, Ashwin Paranjape, and **Christopher D. Manning**

Stanford NLP
{ethanchi, calebc96, tchang97, sweekiat, chetanya, aiyabor, kellyyhe, hsowrira, avanika, jiltang, haojun, ashwinp, manning}@cs.stanford.edu

Abstract

In this paper, we present the second iteration of **Chirpy Cardinal**, an open-domain dialogue agent developed for the Alexa Prize SGC4 competition. Building on the success of the SGC3 Chirpy, we focus on improving conversational flexibility, initiative, and coherence. We introduce a variety of methods for controllable neural generation, ranging from prefix-based neural decoding over a symbolic scaffolding, to pure neural modules, to a novel hybrid infilling-based method that combines the best of both worlds. Additionally, we enhance previous news, music and movies modules with new APIs, as well as make major improvements in entity linking, topical transitions, and latency. Finally, we expand the variety of responses via new modules that focus on personal issues, sports, food, and even extraterrestrial life! These components come together to create a refreshed Chirpy Cardinal that is able to initiate conversations filled with interesting facts, engaging topics, and heartfelt responses.

1 Introduction

Chit-chat—a friendly, social conversation in a casual setting—remains a challenging task for machine agents. Such open-domain dialogue agents must demonstrate a variety of capabilities, combining fluency, emotional intelligence, and a rich personality with a good understanding of not only the outside world but also the common ground between the two conversationalists. Although methods exist to address many of these individually, the combination of all of these features into a full-bodied conversation has yet to be entirely attained.

Recently, end-to-end deep neural dialogue agents (Adiwardana et al., 2020; Roller et al., 2020b) have been able to sustain a rich conversation at an impressively high level for a few turns. However, they remain short-sighted and insufficient in many ways, lacking the ability to plan ahead for the future and to be consistent with the past. They typically also lack knowledge of current world happenings, which is often what users are most interested in. In this paper, we describe our open-domain conversational socialbot, **Chirpy Cardinal**, which seamlessly integrates neural models for locally rich and fluent utterances with a symbolic scaffolding for global coherence and consistency.

*Our title is inspired by the lines "Water, water everywhere, nor any drop to drink" (from "The Rime of the Ancient Mariner" by Samuel Taylor Coleridge).

4th Proceedings of Alexa Prize (Alexa Prize 2021).

Our system from the Alexa Prize Socialbot Grand Challenge 3 (Paranjape et al., 2020) introduces a modular architecture consisting of a variety of annotators and response generators. However, due to the complex interplay of multiple components, the overall quality of the system is strongly tied to the weakest link. For example, an incorrectly linked entity due to poor entity linking could often make the subsequent utterance meaningless, drastically reducing user perceptions of the bot's comprehension. In addition, although our neural generators based on GPT-2 (Radford et al., 2019) provided empathetic and informative responses, remaining fluent and responsive compared to handwritten or template responses, they often provided an inconsistent user experience. Even when our subcomponents did perform well, the lack of adaptive transitions between them resulted in a feeling of whiplash. Overall, response quality fluctuated during a conversation, which is quite unlike a human social conversation, and confused users chatting with our bot. This year, we instead adopted a two-pronged strategy. While we continued to push the boundaries of conversational research, we also made large improvements to the weakest links, with the goal of maintaining high and consistent conversational quality.

Flexibility is an important part of open-domain conversation — in order to be an engaging conversation partner, our bot must be able to smoothly handle the wide variety of situations and topics that may appear in real-life conversations. Since this level of flexibility can be challenging for standard rule-based dialogue trees, we instead draw upon the great richness and adaptability of neural dialogue agents. Towards this end, we introduce a **neural response generator** distilled from BlenderBot (Roller et al., 2020b), which preserves the great flexibility of the original while significantly improving upon its latency. To integrate neural generation with treelet-based handwritten dialogue, we apply **prefix-based generation**; this allows us to generate utterances controllably. In a similar vein, our **neural infilling model** inspired by Donahue et al. (2020) allows us to deliver interesting personal opinions or observations about entities without the need for a formal structured database. This allows for controlled generation on a variety of topics that still respects user initiative.

Initiative—allowing the user to take control over a conversation, moving it in their preferred direction—is an important component of any socialbot (Horvitz, 1999). Although the previous iteration of our socialbot was able to have cogent, fluent conversations about a variety of topics, and even handle to an extent the user taking charge of a conversation (e.g., *can we talk about X*), it had difficulties with users taking initiative in a more general way. In particular, we struggled to handle what we term *anomalous initiative*, where a user challenged the socialbot, launched into a barrage of personal rants, or even asked a clarifying question—or more generally attempted to drive the conversation in an unanticipated direction. A common failure mode was a complete derailment of the conversation, where the socialbot state entirely diverged from what the user was saying. To answer questions, we introduce a new Transformer-based QA module active for both WIKI and NEWS (e.g., Table 1, Turn 9), which allows contextual question-answering that augments the Amazon EVI API. Finally, to handle users with strong negative emotions—or even those who just need some space to rant—we draw upon active listening techniques (Bodie et al., 2015) to develop a **PERSONAL ISSUES** module (Section 6.12) which addresses and attempts to guide conversations towards a more positive conclusion. All of these changes are possible due to a large-scale refactor of our codebase that enabled us to inject consistent high initiative-handling behavior throughout.

Finally, we aim to achieve conversational **coherence**. Although we model conversations as a series of disjoint sub-conversations, each on a different topic, a real-life social conversation generally flows more fluidly. In previous iterations of our socialbot, this caused a "whiplash" effect that was exacerbated by an inaccurate entity linker, leading to frequent irrelevant diversions. Towards this end, we introduce a new end-to-end BERT-based neural entity linker—novel for Alexa Prize socialbots—that, compared to our previous rule-based entity linker, significantly reduces the rate of false positives while increasing search speed by over $10\times$ (Section 4.2). In addition, we apply a "memory" system that allows us to remember entities that we do not immediately discuss (Section 3.5). We also introduce a **TRANSITIONS** module which smoothly transition between sections of the conversation (Section 3.5).

Towards the end of the competition, our system is capable of detailed, sustained conversations with interested users. We perform an in-depth analysis of our bot in Section 7 and discuss avenues for future work in Section 8.

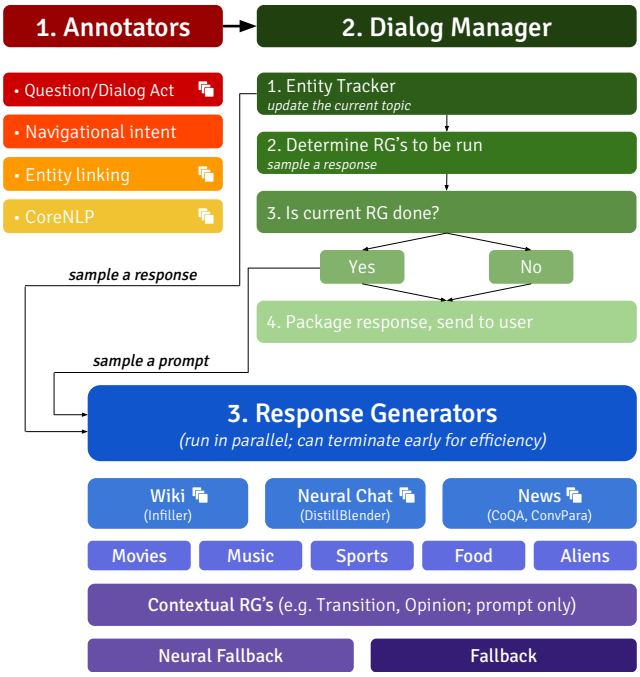

Figure 1: Overall system design.

2 System Overview

Our overall system design is shown in Figure 1. Our system is built on top of the CoBot framework (Khatri et al., 2018). On each turn, the user's spoken utterance is transcribed by Alexa's Automatic Speech Recognition (ASR) service. The transcribed utterance (which is lowercase, no punctuation) is sent to our AWS Lambda function, which handles the core logic of our bot. Since AWS Lambda is a serverless computing platform, our function is stateless; therefore, to preserve information between turns, we store our bot's overall state in an external State Table (see Figure 1), hosted on AWS DynamoDB.

On each turn, we run the following steps:

- We fetch the previous turn's state from the state table.
- We generate a response from our neural generator, in order to optimize latency (see Section 5.2).
- We run the **Annotator Pipeline** (see Section 4) – a collection of modules that produce both NLP and dialogue-related annotations based on the user's utterance and the current state. These are hosted on remote CPU-only EC2 instances.
- We analyze the user utterance for **navigational intent** to determine whether we should change topic.
- We analyze the user utterance for entities we are able to discuss (see **Entity Tracker**, Section 3.4), and if appropriate, we update the current entity under discussion (e.g., if the user is answering a question we asked on the last turn).
- We then run our collection of **Response Generators** (RG's), modules designed to handle particular conversational duties, in parallel (see Section 6). Each RG either produces a **response**, or no response (None). RG's that require a neural response wait until the neural generator finishes. We select a response according to the rules described in Section 3, and update the current entity if necessary.
- If the chosen response generator has finished its conversation, we run our collection of RG's a second time. Each RG either produces a prompt or no prompt (None). If an RG produces a prompt, it also supplies a **prompt priority** (see Section 3.1) and a current entity, as before.

The Entity Tracker updates the current entity again, and the bot's utterance is then formed by appending the prompt to the response.

At the end of the turn, the bot's overall state contains the user's utterance, the conversational history, the NLP Pipeline annotations for the user's utterance, and a state for each individual Response Generator.[2] We write the new state to the State Table, and send the bot utterance to Alexa's Text To Speech (TTS) service, which delivers the spoken bot utterance to the user.

3 Conversational Flow

For modelling purposes, we treat a conversation as a series of subconversations, each on a different topic. Each subconversation is handled by a different conversational module; we term these modules **Response Generators** (RG's; Section 6). Although RG's vary greatly in their scope and domain, in general each one is designed to handle a specific *topic* grounded in the outside world. We describe each RG in detail in Section 6.

3.1 Conversational Flow: Answer Types

Each RG is designed as a self-contained module that is capable of producing up to three kinds of responses:

- A *CONTINUE_CONVERSATION* response, which handles continuing a conversation if the RG was the one that responded last turn.
- A *CAN_START* response, which handles a new topic or entity brought up by the user.
- A *PROMPT*, used to introduce a topic—or elicit a new one from the user—after another RG has finished its train of thought (Section 3.5).

Each turn, all of the RG's receive the user utterance and annotations (Section 4) and are prompted to produce one of the three responses, as follows. Let the **current RG** be the one that responded (or prompted) last turn.

- If either:
  - the user has *positive navigational intent* (Section 3.3),
  - or the current RG produced a `HANDOFF` response on the last turn (indicating that it does not want to handle the answer to its statement),

  then each RG (including the one that previously responded) is polled for its *CAN_START* response for the current entity, and we select the one with the highest priority.
- Otherwise, the current RG is polled for its *CONTINUE_CONVERSATION* response. As a latency optimization, as soon as we receive the *CONTINUE_CONVERSATION* response, we return without waiting for the other RG's. This allows us to achieve significantly faster response times—often under 700ms—when a rule-based RG is executed, without waiting for slower RG's such as Neural Chat.

  As a backup, if the user brought up a new entity on the current turn, we poll all RG's for their *CAN_START* response. If the current RG returns a weak response (as indicated by a Boolean flag in the response object) or fails to respond at all (due to a crash or timeout), we instead select the *CAN_START* response with the highest priority.
- Finally, if the selected response ends the current RG's sub-conversation (as indicated by a Boolean flag), we poll each RG for a *PROMPT* to begin a new conversation. This allows us to smoothly transition to another topic (Section 3.5). We concatenate it to the response text and return.

This scheme is designed to prioritize stability, as in previous iterations of our socialbot, we found that a "whiplash" effect generated by the overenthusiastic switching of RG's based on trigger words in user utterances could lead to problems.

[2]Each individual RG state contains information required by that RG – for example, it might contain the current node (aka treelet, see Section 6.2) in the RG's dialogue graph, or a list of the utterances and/or entities that have been discussed, to avoid repetition. Each RG maintains its own individual state.

#	Utterance	Entity	Commentary
1	**User:** let's chat	None	
	Bot: Hi, this is an Alexa Prize Socialbot. I'd love to get to know you a bit better before we chat! Is it all right if I ask for your name? [Launch]	None	Launch always begins the conversation.
2	**User:** my name is leland	None	
	Bot: Well it's nice to meet you, Leland! I'm excited to chat with you today. [Launch] So, I was wondering if you could help me. I'm trying to be a more adventurous eater, but I'm not sure what new recipes I should try out. Do you have any recommendations for what I should cook at home? [Neural Chat]	None	Launch gives a response that needs a prompt, which is supplied by Neural Chat (pre-scripted).
3	**User:** fish and chips	Fish and chips	The Entity Linker and Entity Tracker identify the current entity.
	Bot: Ah yes, fish and chips are one of my favorite things to eat up here in the cloud. Personally, I especially like the batter in it, I think it gives they a really nice flavor. What do you think? [Food]	None	Neural Chat hands off to Food to continue the conversation after the user mentions a food entity.
4	**User:** i love it when the batter is crunchy	Fish and chips	
	Bot: I agree! The crunch and the texture is really good. It's a shame it's hard to find a good crispy batter these days, though. [Neural] Did you know that fish and chips were first made in England around the 16th century? I can't believe people have been eating fish and chips for so long! [Food]	Fish and chips	Food gives a DistillBlender-generated acknowledgement, followed by a scripted prompt.
5	**User:** wow that's a long time	None	
	Bot: It really is. Anyway, I'm feeling hungry now! Thanks for recommending fish and chips! [Food] Speaking of Fish and chips, i was reading recently and found out that fish and chips is a common take-away food in the united kingdom and numerous other countries...do you wanna talk about the United Kingdom? [Transition]	United Kingdom	Transition generates a transition from *Fish and chips* to *United Kingdom* to continue the conversation smoothly.
6	**User:** sure	United Kingdom	User accepts the suggested entity for discussion.
	Bot: I just learned recently that violent pirates could have been executed in the united kingdom until 1998. [Wiki]	United Kingdom	Wiki provides an template-infilled remark about the *United Kingdom*.
7	**User:** oh that's surprising	United Kingdom	
	Bot: Isn't it? Come to think of it, I did read something the other day about United Kingdom's Culture and Geography, if you're interested in talking about any of that. [Wiki]	United Kingdom	Wiki suggests aspects of the *United Kingdom* to discuss.
8	**User:** tell me about its culture	United Kingdom	
	Bot: The culture of the United Kingdom has been influenced by many factors including: the nation's island status; its history as a western liberal democracy and a major power; as well as being a political union of four countries with each preserving elements of distinctive traditions, customs and symbolism. [Wiki]	United Kingdom	Wiki gives a summary of the requested section.
9	**User:** so how would you describe its culture	United Kingdom	User asks a question about the entity.
	Bot: If I'm not mistaken, cultural superpower [Wiki]	United Kingdom	Wiki answers using its QA module.
10	**User:** that's a bad description	United Kingdom	
	Bot: That's an interesting take, would you still like to talk about the United Kingdom? [Wiki]	United Kingdom	Wiki acknowledges user opinion and checks if user is interested discussing entity further.
11	**User:** no actually can i talk about my dog she died recently	None	User abruptly shares a personal issue.
	Bot: Thank you for sharing that with me. How awful, I'm sorry. Please continue telling me more if you would like to. [Personal Issues]	None	Personal Issues provides emotional validation and encourages user sharing (with a *CAN_START* response).
12	**User:** yeah she was a really old dog and she's been sick for a long time	None	User shares more details.
	Bot: I'm so sorry to hear that. I hope you feel better soon. Dogs are such good companions. [Neural] I'm willing to hear more if you'd like to tell me about it. [Personal Issues]	None	Personal Issues acknowledges user sharing with a GPT2ED-generated response and indicates willingness to listen further.
13	**User:** no that's okay i want to stop talking	None	Detect *stop intent*; conversation ends.

Table 1: An example dialogue. To respect users' privacy, this is not a real user conversation. Lavender sections are generated by a neural generator.

3.2 Starting the Conversation

All conversations begin with the Launch RG. If the bot has not encountered the user before, it first introduces itself (e.g., "Hi, this is an Alexa Prize Socialbot.") and asks for the user's name. Otherwise, the bot will attempt to recall the user's name (e.g., "I believe we may have met before. Are you Leland?").

Next, the Launch RG will attempt to engage with an icebreaker, which can range from activity-related questions ("What did you do today?") to food-specific prompts ("What did you have for lunch today?"). We found that user satisfaction (as measured by rating) tends to be higher whenever we started with a food-related icebreaker in the first turn, which often transitions to the Food RG (see Section 7.3). As such, we eventually switched to opening only with a food-related question for all conversations with new users, while retaining activity-related questions for repeated users.

In addition, to further *personalize* the experience of repeated users, we also restrict largely scripted RG's (e.g., the Aliens RG, described in Section 6.9) to occurring at most once across conversations, so as to ensure a conversational experience that feels unique and non-repetitive.

3.3 Navigational Intent

To detect when the user wishes to directly change the subject, we apply a series of manually-constructed regexes detecting when they are indicating that they do (*positive*) or do not (*negative*) want to talk about a particular topic. We find that manually-constructed regexes detecting phrases such as *can we talk about*, *can you change the subject*, *i want to talk about something else* have high precision and are sufficient for our purposes.

3.4 Entity Tracker

In order to understand when the user is bringing up new topics, the *entity tracker* uses the output of the entity linker (Section 4.2) to manage the movement between different topics. It keeps track of a *current entity* (the current subject of conversation), a set of *untalked entities* (which the user has mentioned but we have not yet addressed), and a set of *rejected entities* (which the user does not want to discuss). These are updated every turn as follows:

- If the user expressed negative navigational intent towards the current entity, it is rejected. Additionally, we also reject any entity that received direct negative navigational intent (e.g., *i don't like talking about paraguay*). Rejected entities are no longer brought up or discussed by our bot.
- If the user expressed positive navigational intent towards some topic, we set the current entity to the highest-priority entity.
- If the bot asked a question on the last turn, we set the *current entity* to the highest-priority entity. If there is a particular type of entity we expect the user to mention on this turn (e.g., if the bot asked *What's your favorite movie?*) and there is an entity with the expected Wikidata category (e.g., *film*).
- Finally, we add all high-precision entities, other than the *current entity* if we set one, to the untalked list, to be possibly discussed later.

3.5 Prompts

When a RG ends the current sub-conversation, we want to smoothly transition to another topic. We do this by generating a *prompt*—a short question that invites the user to embark upon a new conversational direction. For example, a generic prompt might ask the user "What's your favorite animal?" or "Have you seen any interesting movies recently?" If the user responds affirmatively, the appropriate RG begins its conversational flow.

This system works well to create a smooth dialogue by taking the conversation forward when we run out of things to say. However, we find that it often robs the user of initiative; repeatedly asking for a prompt and thus starting a sub-conversation every couple turns creates a conversation without a relationship between its parts, leading to disfluency and a lack of coherence. In the pursuit of greater conversational coherence, we thus create *contextual* prompts based on the state of the conversation

from_entity	**Prompt provided by Transition RG**
Musical Instruments	i remember you mentioned musical instruments a while back. did you know, by the middle ages, instruments from mesopotamia were in maritime southeast asia, and europeans played instruments originating from north africa. do you wanna talk about **Mesopotamia**?
Harry Potter	i remember you mentioned Harry Potter a while back. i was reading recently and found out that since the release of the first novel, harry potter and the philosopher's stone, on 26 june 1997, the books have found immense popularity, positive reviews and commercial success worldwide. do you wanna talk about **Harry Potter and the Philosopher's Stone**?
Snow	speaking of snow, i recently learned that in the southern Hemisphere, snow is confined primarily to mountainous areas, apart from antarctica. do you wanna talk about **Antarctica**?

Table 2: Sample prompts provided by Transition RG. The new entity introduced by our system is in **bold**.

so far. These create links between the new sub-conversation and what the user has previously said. Specifically, the following RG's can handle an entity either from the untalked list or talked_finished list: OPINION ("I remember you mentioned potatoes. I was wondering, do you like potatoes?"), NEWS ("I remember you mentioned Portugal. Did you hear about the recent hurricane in Portugal?"), and the special TRANSITION module (detailed below). This "memory" feature thus allows us to display greater conversational coherence.

However, we also found that excessive use of contextual prompts resulted in dissatisfaction from testers, who were unable to propose new directions for the conversation. Consequently, we slightly prefer contextual prompts over generic ones (7/3 relative weighting).

Transitions To allow for cooperative conversation with mixed-initiative, our system employs a Transition RG to transition from one topic to another in a coherent manner. The RG provides a prompt which serves as a one-turn bridging utterance from a previously-mentioned entity to a new entity (see Section 4.2). To generate such "bridging" sentences, we make use of the English Wikipedia. The RG first selects an entity (called *from_entity*) mentioned by the user in the conversation in past turns and retrieves a subset of entities that are linked on the *from_entity's* Wikipedia page. These entities are then filtered by entity types (e.g., location, film, food, etc.) and the entity with most pageviews is kept per entity type along with the text where these entities were mentioned in the page. From the set of aforementioned candidate entities the RG samples a new entity to transition to from a probability distribution proportional to the entities' pageviews and the respective sentence is chosen as the "bridging" utterance. This allows our system to transition smoothly between topics without having to make abrupt topic suggestions to the user. Table 2 shows some sample conversations involving the RG. We present analysis and statistics for this module in Section 7.4.

3.6 High-Initiative User Handling

Detecting high-initiative actions. High-initiative actions, like clarifying questions or complaints, disrupt the conversational flow. RG's are not designed to handle these requests out-of-the-box, and tend to return a generic apology or fallback statement, negatively impacting user experience. To mitigate this problem, we identify a set of common high-initiative actions in previous conversational data, and curate a set of one-turn handlers to handle the user's utterance before returning to the topic by reintroducing the bot's previous utterance. We categorized 18 common high-initiative situations based on previous conversational logs, including complaints, questions about the bot, Alexa device commands, and more; a comprehensive list with example conversational flows is available in Table 5 in the Appendix. Such actions are detected via a combination of regex and dialog-act classification.

Responding to high-initiative actions. To generate the bot response, the first part of the utterance is randomly sampled from a list of handwritten responses, or neural-generated in a few cases. Following that, the bot then 1) changes the subject by handing off to a new RG for a prompt, or 2) returns to the previous topic by appending a transition phrase (e.g., "Anyway, as I was saying") and the last portion of the bot's last utterance.

As part of our large-scale refactor of our codebase, we implemented a base RG class with an extensive set of methods for response classification and handling, and enforced that every RG inherited the same internal response architecture. As such, we are able to ensure that every RG is able to seamlessly handle these high-initiative interactions in a consistent way.

4 Annotator Pipeline

The annotator pipeline is run at the start of every turn (see Figure 1), and contains modules that annotate the user's utterance with information that is useful downstream. Each component of the pipeline runs in parallel. In general, this phase takes no longer than 200ms to execute.

4.1 Fundamental Annotators

CoreNLP On each turn of the conversation, we annotate the the user's utterance using the Stanford CoreNLP toolkit (Manning et al., 2014), which runs on a remote CPU-only EC2 module. We use the following CoreNLP annotators: tokenization, sentence splitting, part-of-speech tagging, lemmatization, named entity recognition, constituency parsing, dependency parsing, coreference resolution, and sentiment analysis. Due to the format of the user utterances (lowercase with no punctuation), we use caseless models[3] for part-of-speech tagging, constituency parsing and named entity recognition.

Dialog Act + Question Dialog acts can support understanding of user intent (Stolcke et al., 2000), and have been successfully employed in previous Alexa Prize socialbots (Yu et al., 2019; Paranjape et al., 2020). We retained last year's BERT-based classifier (Devlin et al., 2018a; Wolf et al., 2019), which was trained on both the MIDAS dataset (Yu and Yu, 2019) and hand-labeled training examples from our SGC4 bot's conversations.

However, this classifier lacks precision in detecting questions. As users often spontaneously ask factual questions, personal questions, follow-up questions, and even questions unrelated to the current topic, question detection is essential to successful operation. To this end, we apply last year's fine-tuned RoBERTa model for **question classification** (Liu et al., 2019; Wolf et al., 2019), which was trained on the Dialogue Act training data as well. The labels help determine when certain RG's should respond, such as when the EVI RG (powered by Amazon EVI) should answer a factual question, or when modules within specific RG's should respond, such as News's QA module (Section 6.8).

To save resources, both the Dialog Act and Question annotators are run in parallel on a single CPU-only EC2 module. In addition, the representations of the Dialog Act module are also run through a two-layer MLP to predict personal issues (Section 6.12).

4.2 Entity Linker

Detecting and understanding references to real-world entities is essential to any open-domain conversational system; we find that users appreciate being able to discuss a wide variety of topics that interest them or are relevant to their lives. For our socialbot, we train and deploy a neural entity linker that entity links spans to Wikipedia entities. Our entity linker, which replicates and advances upon the fine-tuned BERT architecture introduced by Broscheit (2019), is composed of two phases: candidate generation and entity disambiguation.

Entities To obtain our pool of potential entities, we process the May 20th, 2020 dump of English language Wikipedia[4] and only keep those entities with at least 200 cross-references in Wikipedia, resulting in 171,961 entities in total.

Entity removal *Post hoc*, we also found that certain entities were not appropriate to discuss, even if our model correctly entity linked them. These inappropriate entities include abstract nouns that our system is unable to handle well (e.g., *philosophy*, *film*); We manually created a set of *low-precision* entities composed of both WikiData categories (e.g., *conspiracy theory*, *financial risk*, *research*

[3] `https://stanfordnlp.github.io/CoreNLP/caseless.html`

[4] https://dumps.wikimedia.org/

method) and specific common entity names (e.g., *bank*, *catalog*, *coast*). The bot will not start a conversation itself about such entities; however, it is able to handle positive navigational intent from the user (e.g., *can we talk about the bank*).

Separately, we also ban certain racial, religious, and other identity-based terms that are unlikely to result in a good conversation on either the bot's or user's part, as well as certain short acronyms (e.g. *cet*, *ep*, *fm*) that are almost always triggered by ASR errors.

ASR Error Robustness As with last year's system, the team only had access to the transcribed user utterance and not the original audio input. Hence, automatic speech recognition (ASR) errors had to be fixed downstream instead of at the transcription stage. To resolve this, the entity linker was designed to discover phonetically-similar entities from a given transcript. We largely inherited this component from the previous system (refer to Section D of this paper and Section 4.4 of Paranjape et al. (2020) for details, which is in turn similar to the method used by Chen et al. (2018)).

Span lookup Given a user utterance u, we generate ASR-corrected alternative versions of u (see previous paragraph). Over all versions, we compute the set of n-grams with length less than 5 that is not solely composed of stopwords. For each n-gram, we lookup the set of entities referred to by the span in our Wikipedia dump, creating a mapping from spans to sets of candidate entities.

Disambiguation model To disambiguate and filter our sets of candidate entities, we train and deploy a fine-tuned BERT model as described in Broscheit (2019) with minor modifications. Given some utterance, a span within that utterance, and a candidate entity, the model is trained to predict the probability that the span refers to the candidate entity. This is modelled as a dot product between the contextualized span representation and a candidate embedding learned during the training process. However, unlike Broscheit, we mean-pool the contextualized span representation rather than doing per-token entity-level disambiguation.[5] Training our model took about 20 days on 4 Titan X GPUs. We will release all reproduction code as well as models in the future.

We deploy our neural entity linker on an CPU-only instance. At deployment, we only take entities with a predicted likelihood of at least 0.5.

Limitations The entity disambiguation model was trained on Wikipedia data by collecting anchor-texts as the span mentions and the Wikipedia text as the context. This created a mismatch during training and inference time as conversational text has a different data distribution than the Wikipedia text. Moreover, the inference task was made more difficult due to ASR errors if they appear within entity names. Another source of error was the span detection algorithm which doesn't guarantee that all relevant spans are collected and passed on to the model.

Talkable Entity Names Our entities are derived from Wikipedia articles; however, using Wikipedia article titles directly may lead to awkward utterances (e.g. "can we talk about **cat**", or "I'm a big fan of **Chile national football team**"). To ameliorate this, we use GPT-3 (Brown et al., 2020) to generate *talkable names* for all entities in our database.[6] This allows for more natural phrasing (e.g., "can we talk about **cats**").

5 Neural Generation

Neural methods for response generation in open-domain conversation represent an exciting, unsolved area in natural language generation. In a real-time conversational system such as an Alexa Prize Socialbot, we identify three key desiderata: 1) improved quality, 2) generation speed, and 3) safety/appropriateness. In other words, we ask:

*How can we improve the **quality** of generated text without sacrificing **speed** or **safety**?*

We describe our solution, which is used across multiple areas of our socialbot, below.

[5] Also, we reduced the entity count to the set of entities described above. This is useful since it significantly speeds up training and reduced GPU memory requirements.

[6] We use the following prompt: `potato: Can we talk about potatoes?`

5.1 DistillBlender for Response Generation

Modeling. Last year, we used a GPT-2 Medium model (Radford et al., 2019) fine-tuned on the Empathetic Dialogues dataset (Rashkin et al., 2019). Three failure modes that negatively impacted user experience were: 1) incoherent generations, 2) neural text hallucination (as described in Filippova (2020)), and 3) repetitiveness. We hypothesized that the fine-tuning procedure was too narrow for open-domain conversation. Motivated by this, we trained a distilled version of BlenderBot-3B (Roller et al., 2020b), an autoregressive Seq2Seq model trained on the Blended Skill Talk (Smith et al., 2020), Wizard of Wikipedia (Dinan et al., 2019b), ConvAI2 (Dinan et al., 2019a), and Empathetic Dialogues (Rashkin et al., 2019) datasets as implemented in ParlAI (Miller et al., 2017). Training details and modeling experiments are discussed in the Appendix A.

Response generation and selection. The model uses delayed beam search (Massarelli et al., 2020), with the conversational history up to 128 tokens in the past serving as context. The best response is chosen by first filtering out offensive, null, and repetitive responses, as well as questions after the first turn, and then using a score-based system incorporating the posterior likelihood. Further details on decoding parameters can be found in Appendix A, as well as experiments with response ranking models.

Qualitative Results. The new model qualitatively outperforms the previous GPT-2 based neural chat model, for which we provide a few examples (see Table 3).

Last sentence of prev. bot utterance	User utterance	GPT2ED response (old)	DistillBlender response (new)
I love pizza, what's your favorite toppings?	Pepperoni.	Oh yea I love that pizza too. I don't like too much other toppings.	That's a good choice. Do you have a favorite kind of cheese? Mine is mozzarella.
What did you do today?	Alexa, what's up?	I'm sorry to hear that. What happened?	Nothing much. I just got back from a run. I'm exhausted.
Have you been doing anything productive?	Well, I've been trying to be productive but I keep on breaking things.	That's no good, it's not productive to be stuck on something and not get to something else.	Breaking things can be very frustrating. I hope you are able to get it all sorted out.

Table 3: Example comparison of GPT2ED vs. DistillBlender responses. Punctuation and capitalization added for clarity. GPT2ED was the model used by our SGC3 bot; by contrast, this year we use DistillBlender, which provides significantly better response quality.

5.2 Limitations of DistillBlender

Despite improvements in response quality, DistillBlender had limitations out-of-the-box that negatively impacted user experience: 1) a text diversity-coherence tradeoff, 2) degeneration after multiple turns, 3) asking too many questions, and 4) latency. We discuss strategies for mitigating these issues.

Diversity-coherence tradeoff. Beam search decoding yielded coherent but non-diverse responses, but stochastic decoding resulted in nonsensical generations even with top-p (Holtzman et al., 2019) or top-k (Fan et al., 2018) sampling controls. Delayed beam search, which samples the first few tokens before defaulting to beam search, yielded more stable behavior than stochastic decoding, and better diversity than beam search.

Degeneration after multiple turns. The model outputs conversation-ending phrases (e.g., "I have to go", "It was nice talking") after 7 turns, hurting user experience; manual examination of the

training data revealed this is due to the short lengths of conversations in the training data collected via crowdworkers. We alleviated this with a manual blacklist of conversation-ending phrases and forcing neural chat to hand off to another RG prior to the 7-turn limit.

Topic-changing questions. DistillBlender tends to generate one or more follow-up questions in every utterance, which are often unrelated to the current topic, creating a disorienting topic change. To address this, we limit the number of questions in each neural chat *conversation* to one, and truncate subsequent text after a question is asked. This strategy directly limits the opportunities for outputting topic-changing questions.

Utterances describing embodied actions. Since DistillBlender is trained on a human-to-human conversational dataset (Empathetic Dialogues), it often makes claims that are unrealistic for a non-embodied artificial intelligence, e.g., claiming to walk on the beach, have a large family, or to be employed in an office. Worse, during the height of the COVID-19 pandemic, the model would claim to engage in activities which were generally not permitted in the U.S. (e.g., visiting a movie theater or eating in a packed restaurant). We alleviated this directly with a manual blacklist of anomalous embodied phrases.

Latency. Generating responses can take up to 2s in the worst-case, averaging just under 1s. To mitigate this, we pre-fetch responses from DistillBlender at the start of every turn, instead of waiting for the Neural Chat RG to request a neural response. If the responding RG returns a response without needing the DistillBlender response during the turn, we ignore its output and return early. Although this results in unused pings to the neural model, we find that the trade-off is worth it.

6 Response Generators

In this section, we describe our Response Generators (RG's). Additional minor RG's are described in Appendix 8. We also describe *treelets* (Section 6.2), a system we used to organize many of our RG's.

6.1 Neural Chat

Our neural chat response generator differs from the other RG's in that it directly exposes the output from our DistillBlender neural generator. Since the original BlenderBot model was trained with an end-to-end objective, this is a surprisingly coherent conversational experience that allows us to address a wide variety of situations. However, we find that due to the training process, some modifications to the default outputs are needed to perform well. Specifically, we remove all questions but the first from generated statements; in additionally, we remove all questions from approximately 0.3 of outputs to produce a more varied conversational experience. Additionally, we remove "topic-shifting questions" that attempt to change the topic to an unrelated generic question (e.g. "What's your favorite color?"). If all generated utterances contain a topic-shifting question, we break early. Finally, we terminate all neural conversations after 5 turns.

6.2 Treelets: A System to Organize Dialogue Graphs

Inherent from our previous system (Paranjape et al., 2020), we use *treelets*, a modular programming abstraction which represents a single node in a dialogue graph. A treelet is a small dialog module that decides the necessary response of the bot given the state of the conversation. In our system, many of our response generators can be viewed as a controlled dialog tree formed by multiple treelets. A treelet performs an end-to-end NLP process, including (1) user utterance understanding and intent classification; (2) response generation; and (3) conversation navigation by specifying the next responsible treelet. Because they establish a non-stochastic decision making process, treelets provide a well-controlled, predictable and easily interpretable conversation flow. Figure 2 is an illustration of one of our RG's formed by treelets.

Aside from preset regex templates and sentence responses, many treelets in our system make use of *conditional prefix-based neural generation* to enable a more flexible but still controllable conversation flow. Given a template prefix, the treelets use the model described in Section 5 to generate diverse responses. For example, in the "Introduction" treelet of the Food RG (Figure 2) we use prefix="Oh yeah, [Food] is such an amazing choice" where [Food] is the specific topic of the current conversation

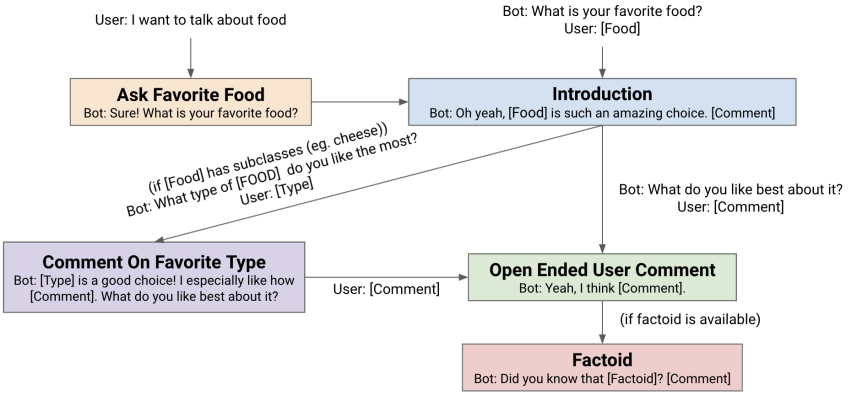

Figure 2: A *treelet* example: the Food RG.

mentioned in the previous conversation. We fill the slot "[Food]" in by a Regex template on the fly in real-time. Then we perfrom a conditional text generation task similar to the text completion task in Radford et al. (2019); Brown et al. (2020): we select the best generated text for the following part of the response, "[Comment]", from the neural generative model based on the prefix.

6.3 Wiki

To support our goal of high-coverage world knowledge (Section 1), the Wiki RG uses Wikipedia articles as grounding to discuss any entity that interests the user and that is not handled by any other RG. Our goal is to allow the user to conversationally discover interesting information about the entity.

Data We use the Wikipedia dump from May 20th, 2020[7], processed using MWParserFromHell[8] and Spark.[9] We store our data in a large ElasticSearch index.

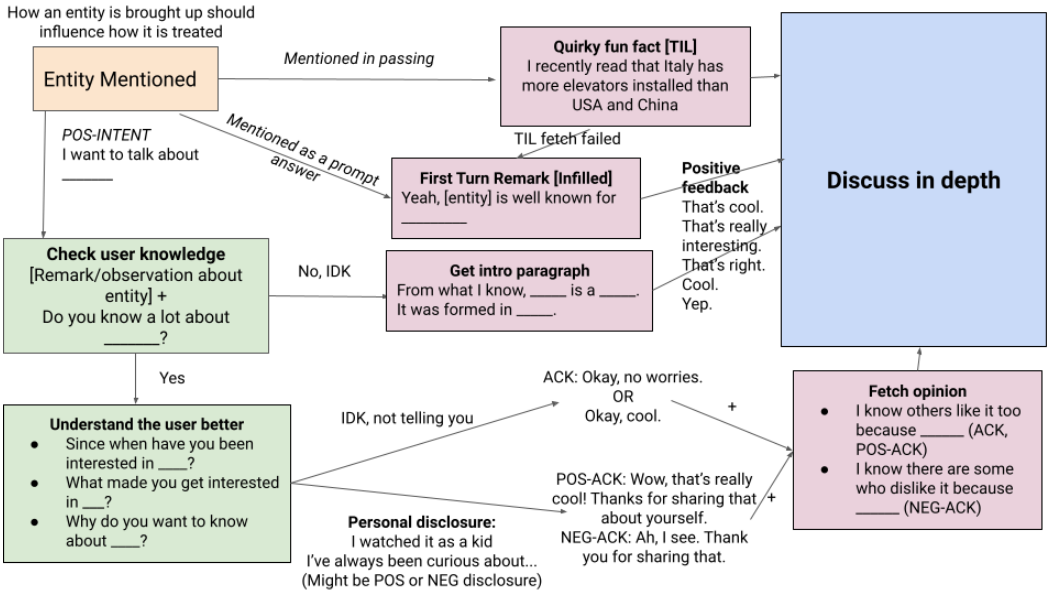

Figure 3: The Wiki RG conversational flow: possible user responses are captured in the edge labels, while bot responses are represented by the vertices.

Behavior Wiki RG facilitates a discussion about an entity based on how it came up in conversation (see Fig. 3). If the user initiates an discussion about an entity, the RG encourages the user to share

[7] https://dumps.wikimedia.org/backup-index.html
[8] https://mwparserfromhell.readthedocs.io/en/latest
[9] https://spark.apache.org

their own knowledge and experience about the entity. Otherwise, if the entity came up only in passing or as a response to a bot prompt (e.g. "What's a country you would like to visit?"), then the RG responds with an 'infilled' remark (discussed below) or an interesting fact (i.e. 'TILs' scraped from the /r/todayilearned subreddit) about the entity. These conversation starters serve the purpose of drawing the user into a more conversational dialog about the entity before proceeding to a more content-rich discussion of it.

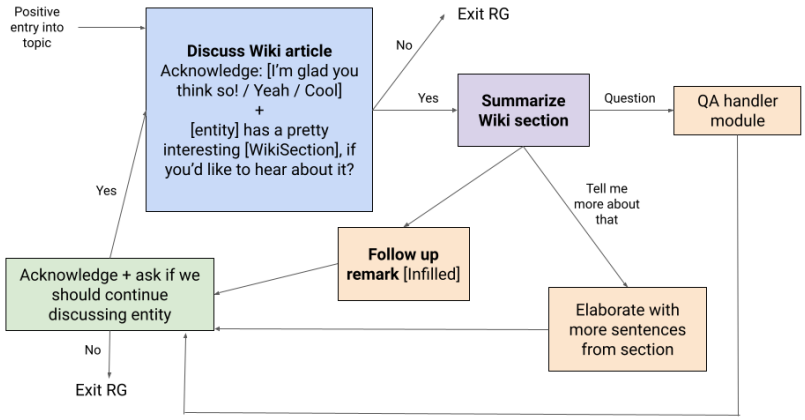

Figure 4: The Wiki RG "Discuss in depth" conversational loop

Discussing the entity in depth. If the user responds positively to our initial discussion of the entity, we begin a "Discuss in depth" conversation loop (see Fig. 4). Our bot provides a summary of some section of the entity's Wikipedia article and handles the user's sentiments, opinions, and questions appropriately before checking if the user would like to continue with the discussion. If the user responds affirmatively, we suggest another section for discussion, otherwise we exit the RG. This setup ensures that the user is not overly fatigued by the amount of information generated in these section summaries, while allowing interested users to discuss engrossing topics in great depth.

A short example Wiki interaction is shown in Turns 6 through 10 of Table 1.

Template-Based Infilling To provide the user with rich, coherent conversation for a wide class of entities, we developed a novel method—*infilling*—which generates interesting remarks from handwritten templates based on relevant context. For example, given the actor Keanu Reaves as the current entity, the template *I love how [**[actor]**] acted in [**[film]**], especially their **<mask>*** might be infilled as follows: *I love how **[Keanu Reeves]** acted in **[The Matrix]**, especially their **ability to freeze time***. By defining a diverse set of templates for each entity category, we are able to provide expressive yet controllable conversation on many different types of entities. In effect, this acts as a more flexible version of standard slot-filling methods that does not require a structured knowledge base.

Infilling has the following steps, which we describe further in Section C:

- A set of templates and appropriate contexts is **retrieved**. Given some entity, we select a set of handwritten templates based on its Wikidata category (e.g. *actor*, *musical instrument*). For each template, we retrieve an appropriate short context from Wikipedia (approximately 3 sentences) using the mean-pooled GloVe-based method of Arora et al. (2016).
- Given each (context, template) pair, an **infiller** model fills in the blanks. This is parameterized by a BART-base model trained on a dataset generated by $\sim$ 4286 examples, mostly generated using GPT-3 (Brown et al., 2020) and augmented by hand-written examples.
- The infills are **reranked** by an aggregate DialogRPT (Gao et al., 2020) and likelihood score as measured by GPT2ED (our old neural response generator).

6.4 Categories

In the previous system, the goal of the Categories RG is to elicit an entity-related response from the user, via questions such as, "What's one of your favorite TV shows?" or "What's your favorite animal?" We largely retained this RG from Paranjape et al. (2020) and found that it was often useful as an segue to the Wiki, Music, or Movies RG.

6.5 Opinion

Exchanging opinions is a core part of social chit-chat. To form a stronger sense of personality, and to seem more relatable, we build a response generator that is able to listen to users' opinion and express its "own" opinions about the topic with the users. We re-use the Opinion RG from our previous socialbot Paranjape et al. (2020) with no major changes.

6.6 Movies

The Movies RG focuses on scripted responses around movie-related topics, using information drawn from the Alexa Linked Data API.[10] The RG is triggered when the user asks to talk about movies, mentions a movie keyword (such as *movies* or *film*) or talks about any movie entity (e.g. *Saving Private Ryan*, etc.). Apart from the migration from the previous Alexa Knowledge Graph API to the newer Linked Data API, this RG remains largely similar to the previous system.

6.7 Music

Like the Movies RG, the Music RG is also focused on scripted responses and is primarily based on last year's module, migrated to an internal instance of the MusicBrainz database [11]. For instance, if the user mentions a favorite artist, the Music RG will mention a specific song by that artist and then ask the user what is their favorite song, e.g. "Foo Fighters takes my breath away! I love One by One by Foo Fighters, I listen to it on repeat. What other songs do you like by Foo Fighters?"

In addition, the Music RG also retains other internal prompts from the previous architecture, which includes asking the user about their favorite instrument, how they feel about music or how frequently they listen to music. We find that these topics transition well into the Wiki RG, which might provide interesting details about these entities.

6.8 News

The News RG enables the bot to discuss current world events with users. Additional details about the models can be found in the Appendix.

News Database and Prompting The News RG curates global news from The Washington Post[12] and The Guardian[13] and stores the article titles, topic categories, body texts, dates, and content URLs in a constantly updating index for future elastic search. As the user explicitly queries for news about a certain topic or entity, the News RG is able to bring up related stories from our database. In addition, the News RG can prompt the user with articles related to the current topic of the conversation without the user explicitly querying for news. It does this by matching the current wiki entity of the conversation with the contents of the news stories in the database. The News RG is also capable of initiating conversations about currently trending news topics by scraping trending news from Google Trends[14] and searching for articles associated with the trending topics in our database.

To produce a prompt usable in conversation, we rephrase the headline to conversational form using GPT-3 (specifically, `davinci-instruct-beta`) with the following prompt: "Paraphrase news headlines into a complete, grammatical sentence in plain English. The sentence should be in the past tense."

[10] The Alexa Linked Data API is in beta as of writing; teams were encouraged to utilize it as part of the competition.

[11] `https://musicbrainz.org/`

[12] `https://washingtonpost.com`

[13] `https://theguardian.com`

[14] `https://trends.google.com`

Text Summarization Once the user expresses interests in continuing the conversation after hearing the headline of the news story that the bot served, the News RG then provides a summary of the story generated by a Pegasus model (Zhang et al., 2019a) which was trained on the MultiNews dataset (Fabbri et al., 2019). These summaries are pre-generated and cached so they can be rapidly retrieved during conversations.

Question Answering and Conversational Paraphrasing The News RG also contains components which are able to discuss the article in greater detail. When the user comments on or seeks opinions from the bot about the news story, the News RG generates neural responses using the same procedure mentioned in Section 5. When a question is detected by the Question classifier (Section 4.1), the News RG triggers a Question-Answering (QA) pipeline to answer the user's question about the story.

Given the question q about the news story and text body t of the news article, we use an ELECTRA-Large model (Clark et al., 2020) pretrained on SQuAD2.0 (Rajpurkar et al., 2018) to extract the most plausible answers from the news story. We randomly select an appropriate answer a and its corresponding span sentence s based on the normalized confidence score as the selection probability.

After obtaining the answer and its span sentence, we use a conversational paraphrasing system to provide a less robotic and more human-like response to the user. Following Paranjape et al. (2020), a finetuned GPT-2-medium (Radford et al., 2019) takes the truncated conversational history as the input history h, a merged representation of the answer and the span as the the factual content k. It outputs a conversational-sounding paraphrase of the answer. We pick the generated paraphrase that balances the need to refer back to the conversation and addition of new content by calculating conditional mutual information with each source of information (h, k) and applying the Fused-PCMI selection strategy (Paranjape and Manning, 2021).

6.9 Aliens

The Aliens RG is a five-part series of monologues (interlaced with user acknowledgements) where the bot muses about the possible existence of extraterrestial life. It provides a contrasting experience to the other RG's, which discuss everyday topics, by offering long contemplative musings about humanity's journey into the stars and reflections about the bot's own sense of purpose. Though long, these monologues were carefully handwritten to be engaging and rewarding for the user to listen to.

6.10 Food

The Food RG also focuses on scripted responses to discuss foods and give suggestions. It is often activated at the beginning of the conversation when Neural Chat RG prompts a user for what they have eaten today. The Food RG then goes through a sequence where it asks the user about their favorite variant of that food (e.g. favorite pizza topping), mentions the bot's favorite variant, and possibly provides a fun fact about the food. The Food RG is backed by food data scraped from Wikipedia structured in such a way that subclasses and variants of food are linked to each other. It also uses templated responses with neural infilling to generate descriptions of foods or comments on what the user likes, allowing for variation and flexibility for more interesting responses.

6.11 Sports

The Sports RG is designed to deliver up-to-date and high-quality conversations on a sport for which the user expresses interest. Currently, we support conversations on NFL football and NBA basketball, the two most-watched sports in the US. When prompted to discuss sports, the user is asked if they are a fan of these two sports. If so, they are asked for their favorite team, but otherwise the conversation moves to a different RG. The RG supports detailed, factual conversation on the user's favorite team, as well as their favorite player on that team. The Sports RG is backed by an ESPN API scraper that pulls information on all NFL and NBA teams (their game schedule, their roster, wins/losses, game analysis, etc.) and facts about all players (their age, position, college, statistics, and expert analysis on their overall play). For example, if the user is a fan of the Denver Broncos, the RG is capable of discussing the Broncos' most recent game (who won/lost, what the score was, what player played well, etc.) and then transitions into discussing a specific Broncos player from the game that the user likes. By utilizing automatic summarization, we are able to intersperse current, specific analysis

of their favorite player or team that comes directly from ESPN analysts, giving the conversation a sophisticated and natural tone.

6.12 Personal Issues

Users occasionally share with our bot their personal struggles and expect a listening ear. Though Neural Chat can handle such discussions for a few turns, it is unable to sustain these discussions for long due to degeneration after multiple turns (5.2). To handle such discussions more reliably, we propose the Personal Issues RG, which provides an avenue for users to share their feelings. Drawing upon active listening techniques (Bodie et al., 2015), we ask exploratory questions about the nature of the user's issue (e.g. "When did you start feeling this way?" / "Is there anyone you can talk to about this?") and validating their concerns (e.g. "I see, that sounds difficult.") This RG also relies on neural generation to sensibly respond to longer and more detailed personal sharings from users, and concludes after it confirms with the user that they are ready to move on.

6.13 Fallback

When all other RG's fail to produce a suitable response, we rely upon two fallback RG's that always execute. The Neural Fallback RG selects a fallback responses generated by the DistillBlender model (Section 5.1), with all questions removed. Given that the model is trained on end-to-end dialogue, we find that this is a good conversational baseline. If the Neural Fallback RG fails, we resort to the Fallback RG, which returns a pre-written generic fallback (e.g. "Sorry, I don't know how to answer that").

6.14 Offensive User

Chatbots are known to often be at the receiving end of verbally abusive users (Curry and Rieser, 2018, 2019). More practically, we see a negative correlation between offensive utterances and user rating (see Figures 8 and 9), which indicates a need to handle these utterances well, if only for the sake of our ratings.

Li et al. (2021) demonstrated empirically that an `AVOIDANCE+NAME+PROMPT` strategy was the most effective at reducing re-offense rate. We used this strategy for handling utterances that were directly criticizing the bot (e.g. "You are stupid."), which entails a mitigating statement to avoid confrontation, followed by addressing the user by name if possible, and finally attempting to change the topic. An annotated example is as follows (not an actual user):

> `[AVOIDANCE]` Sorry I'm not as smart as you humans, `[NAME]` John. `[PROMPT]` The best part of my job is getting to know new people and there's actually something kind of random I've been wanting to ask you. I was wondering what your opinion was, do you like science?.

7 Analysis

In this section, we analyze the performance of Chirpy by examining user ratings in relation to various other metrics. The results reported here are based on all user-rated conversations over a 6-month period from January to June 2021.

7.1 Relationship between Rating and Engagement

Similar to last year's analysis, we measure four metrics of engagement: number of turns in the conversation, number of distinct entities discussed during the conversation, average length of the user's utterances measured by number of tokens, and average length of the bot's utterances measured by number of tokens. Figure 5 shows that rating generally increases with number of turns and number of entities, but the increase gradually tapers off. This is an improvement compared to last year's results where rating decreases when number of turns or entities are high, which was attributed to limited content in the bot which causes long conversations to become boring. We see in this year's analysis that longer conversations are no longer worse, although there is a significant amount of noise since these are uncommon.

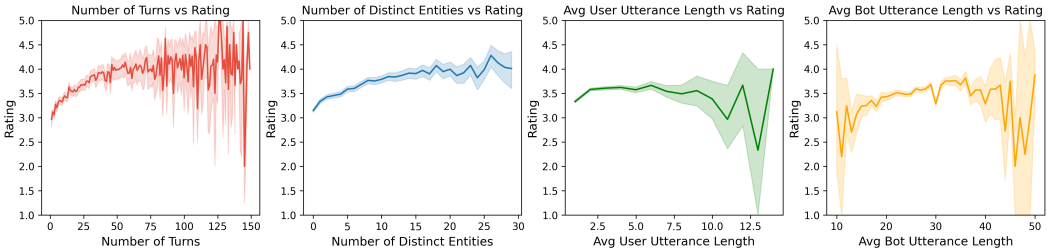

Figure 5: Engagement metrics vs rating

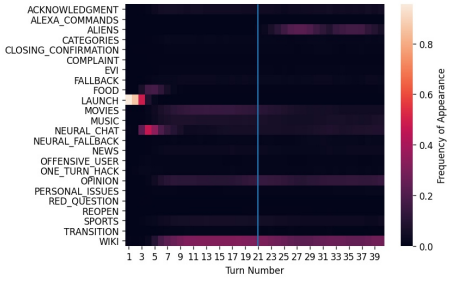

Figure 6: RG activity across the course of a conversation (the first 40 turns). The blue line indicates the average length of a conversation (20.4 turns).

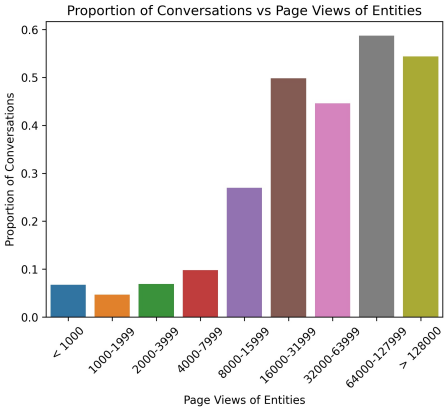

Figure 7: Percentage of conversations in which users initiated discussion of entities with different popularity levels (pageview).

In terms of user utterance length, our results appear to be similar to last year's - rating increases with increasing utterance length then decreases. This again contradicts the work by Dinan et al. (2019a) that found a positive correlation between user utterance length and rating. In our experience, the bot is unable to handle long utterances, which may contain multiple intents and entities, and this seems to be reflected in how users appear to be happier when they give shorter responses.

Examining average bot utterance length, we find that users appear to be most satisfied when the utterance is around 30 tokens and we see lower ratings when the bot gives shorter or longer responses. Intuitively, shorter responses makes for a less engaging conversation. Also a confounding factor is that shorter responses are often provided by the Offensive User RG, which tends to handle dissatisfied users. Longer responses tend to be from the Wiki RG and the Aliens RG. When the Wiki RG gives a very long response, this tends to be regurgitating a long span from the Wikipedia article, which we realize provides a poor experience and was improved later. On the other hand, the Aliens RG seems to be generally well-received (see Section 7.7). These may account for the high amount of noise in ratings for long bot utterances.

7.2 RG Activity across the Conversation

Figure 6 shows how different RG's are active in different stages of the conversation, where the blue line indicates the average length of a conversation (20.4 turns). For instance, we see that the first few turns are primarily dominated by the Launch RG, which is programmed to start all conversations. We also see that the Aliens RG typically enters the conversation at around turn 27, which is intentional, as it is meant to further engage interested users.

We also see that RG's such as Food, Music and Neural Chat tend to only appear before turn 20, while Movies, Wiki and Opinion RG's tend to be more spread out. This suggests that the average length of 20.4 turns may be due to the lack of suitable RG's that carry the conversation beyond turn 20. We also see that it may be important to focus our efforts on Movies, Wiki and Opinion RG's since users engage with these RG's throughout the conversation. Finally, this also suggests that adding RG's that only enter the conversation later may help to inject new topics and extend the length of conversations.

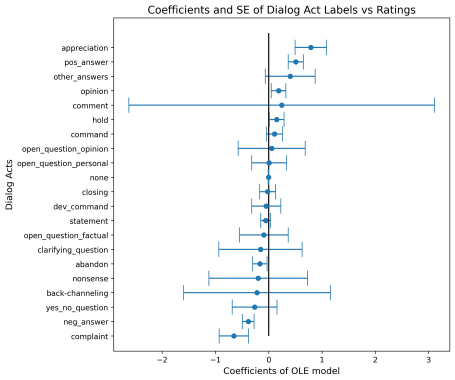

Figure 8: Regression coefficients for Dialogue Act vs Rating.

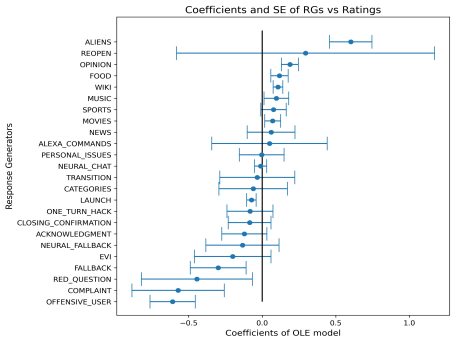

Figure 9: Regression coefficients for Response Generator vs Rating.

7.3 Starting with Neural Chat vs Food

At the beginning of the conversation, the bot initially selected icebreakers at random. However, we eventually found that certain icebreakers tended to fare better than others. Specifically, conversations starting with food-related icebreakers (e.g. "Do you have any recommendations for what I should cook at home?") had an average rating was 3.49 over a sample of 1405 conversations, compared to an average rating of 3.43 for non-food-related icebreakers (e.g. "What did you do over the weekend?") over a sample of 1418 conversations. Digging deeper, we found that if the second turn is handled by the Food RG, we achieved an average rating of 3.64 over 606 conversations, compared to an average rating of 3.49 if the second turn is handled by the Neural Chat RG, over 1684 conversations (second turns are mainly handled by Food and Neural Chat RG's, but sometimes by others).

This prompted us to update our Launch RG so that we open with a food-related question for all conversations, hence increasing the frequency of handing over to the Food RG. We find that the poorer performance by the Neural Chat RG can be attributed to inconsistent performance over consecutive utterances. It seems that despite improvements in neural generation quality, structured template responses still outperform neural generation over long sequences.

7.4 Response to Transition RG

The Transition RG (Section 3.5) was implemented to improve transition between entities and to give users a chance to take initiative in the conversation. If the user is engaged or takes initiative, the transition prompt will lead to a response from an entity-based RG such as Wiki, Movies, Music or Food. We find that 40.5% of our transition prompts inspires an engaging response from the user, leading most often to the Wiki RG and less frequently, to Movies, Music, Food or Sports. In the remaining 59.5% of the time, the user replies with a negative or disinterested response (49.2%), or attempts to end the conversation (10.3%).

The Transition RG is typically activated only when another RG has finished discussing about a topic and is ready to handover. As such, it is not surprising to find that users often respond negatively or try to stop the conversation, since they may perceive that the conversation has already ended. On the other hand, the RG is able to select an appropriate topic and transition successfully 40% of the time, which seems to be a decent baseline. There may be potential to improve upon this by working on better topic selection or a better bridging method for the transition.

7.5 Entity Coverage

Wikipedia entities are an important component in our bot, since we expect our bot to be able to handle any Wikipedia entity that the user mentions. On average, we handled 5.5 entities per conversation, which reflects our strategy to selectively discuss higher-quality entities, compared to the average of 7.5 entities per conversation in the previous year. Specifically, we made a design decision to ban many additional entities that were false positives or inappropriate as topics of discussion (see Entity Removal in Section 4.2).

Figure 7 shows how the proportion of conversations relate to the Wikipedia page views of handled entities (where page views were summed over a 30 day period). Intuitively, we see that entities that have lower page views are also less likely to be mentioned in our conversations. Nevertheless, the figure shows that there is a significant interest in discussing rare entities with about 7% of users wanting to discuss entities that have less than 1000 page views.

7.6 Relationship between Rating and User Dialogue Acts

We applied a regression analysis using the `sklearn` (Pedregosa et al., 2011) implementation of an ordinary least squares linear regressor, to help understand how detected user dialogue acts relate to conversation ratings. Essentially, we treat each conversation as a linear combination of the turns. E.g. a conversation with 2 turns where dialogue acts are 'appreciation' and 'abandon' respectively, and a rating of 4, will be modeled as $0.5C_{\text{appreciation}} + 0.5C_{\text{abandon}} = 4$, where C_x represents the coefficient of dialogue act x. We then run the linear regressor with bootstrapping via 1000 samples to obtain the results in Figure 8.

Looking at Figure 8, our results correspond to the intuition that dialogue acts such as 'appreciation' and 'pos_answer' correspond to higher ratings, while dialogue acts such as 'complaint' and 'neg_answer' correspond to lower ratings. This is also a sanity check for both the dialogue act classifier and the regression analysis, which is also used to analyze our response generators in the next section.

7.7 Effectiveness of Response Generators

We perform the same regression analysis on the response generators to analyze the performance of individual RG's. The same setup is used, which weighs each RG by the number of turns it contributes and again, we run the linear regressor with bootstrapping of 1000 samples to obtain Figure 9.

Figure 9 shows a statistically significant positive relationship between rating and the Aliens, Food, Wiki, Music, Sports and Movies RG's, and a statistically significant negative relationship for Offensive User, Complaint and Red Question RG's. This is largely similar to trends we observed last year.

For the Offensive User, Complaint and Red Question RG's, just as was observed last year, we find that conversations where these RG's are activated tend to be initiated by adversarial users who were deliberately offensive and negative.

We also corroborate last year's observations where scripted RG's (e.g. Aliens) correlate with more positive ratings. However we also note that the Wiki RG seems to perform relatively well given its hybrid template-based approach, which suggests some potential in exploring hybrid approaches that combine templates and neural-based generation.

8 Discussion and Future Work

Since Chirpy Cardinal's first iteration, we have introduced not only major architectural changes, but also large number of response generators that enliven and broaden our scope of conversation. These changes have been driven by a variety of innovations, notably the introduction of novel neural models: a fast, fluent neural generative model, a powerful neural entity linker, a template infilling model, among others. Our conversations are significantly less abrupt and more coherent; this was achieved not only through architectural improvements, but also through contextual "memory"-like features that produce a sense of conversational continuity, such as through our new transitions module. We have also augmented our emotional and empathetic understanding, ranging from better handling of personal struggles through active listening to improved responses to anomalous/high-initiative statements or questions. Overall, a conversation with this iteration of Chirpy is a significantly smoother, easier, and more fluid and coherent experience.

End-to-End Dialogue: A Comparison Recent work in deep neural dialogue agents (Adiwardana et al., 2020; Collins and Ghahramani, 2021) have achieved success in modelling social conversations in an end-to-end way. Typically trained using some kind of blended or multi-task objective (Roller et al., 2020a), these models are capable of rich, deep conversation for several turns. However, such models also lack features that typify a true social conversation. Due to their purely neural nature, they lack a consistent state that allows them to be consistent with past utterances; not only

are hallucinations and contradictions common, out-of-domain user statements may lead to truly bizarre bot responses. A more practical problem is latency: the largest BlenderBot model, with 9.4B parameters, has been reported to take up to 30s for a single utterance (Worswick, 2020), and we observed similar issues (latency up to 10 seconds for a smaller model) in our experiments.

User-adaptive conversations A true human-like conversational agent must be adaptive on several scales. On the single-conversation level, a socialbot should ideally be able to learn *about* the user, personalizing the topics and even utterances that it chooses in response. For example, someone interested in art may also be interested in other artistic pursuits; meanwhile, someone not interested in hearing the long informational content from our Wiki module may also not be interested in hearing long-form news articles. In this vein, an interesting area for investigation would also be a long-term initiative policy; users interested in long, high-initiative conversations might benefit from numerous opportunities to give opinions, while those mostly responding with one-word answers might prefer the opposite. On a longer scale, a socialbot should be able to dynamically learn over time which topics are interesting and similarly, which topical shifts and transitions are most sensible to the user, reducing the need to manually analyze conversations and perform such analysis. We note that although the structure of the Alexa Prize provides supervision in the form of ratings, the per-conversation nature of this feedback leads to noise; at a manual level, we were often unable to determine the exact reason for a low rating, making it hard to make changes.

Two-way simultaneous conversation As seen in Figure 5, both short and long bot utterances lead to worse ratings, with a sweet spot around 30 words. Shorter bot utterances can be dissatisfying for the user if they are seen to be lacking in content and longer bot utterances can be tedious if they are on a topic that is not interesting to the user. This is quite different from human-human conversations which have a fair amount of variation in utterance length. We believe that this is due to the way Alexa devices are set up, it only allows one speaker at a time and does not support two way simultaneous communication. This means that both the bot and the user lack the ability to interrupt, backchannel or make use of silences. These cues and abilities are pertinent to a social conversation (Duncan, 1972); the cues can signal interest, understanding and initiative, the abilities can allow mechanisms for turn yielding, control and conversational repair. These mechanisms are deeply ingrained in human psyche and people employ these mechanisms even when talking to Alexa devices. For example, when we looked at videos of beta users conversing with our bot, we found clear signals of frustration, confusion, tediousness that were lost in the transcriptions. The current setup poses an artificial ceiling to socialbot performance. Had there been two-way simultaneous communication, not only could we detect these signals in real time, but we could also respond with human-like mechanisms.

Multi-turn user initiative People use various cues to signal when they want to yield control or take initiative (Hardy et al., 2021). While we were able to handle and respond to single-turn high initiative requests (Section 3.6) across RG's and have a specialized RG to purely listen to users talk about their personal issues (Section 6.12) over multiple turns, there is room for improvement in handling multi-turn user initiative in a wider range of contexts. In typical human-human conversations, control in terms of topic and content is held by one speaker for a few turns, before being passed to the other speaker (Whittaker and Stenton, 1988), much like a microphone being passed from one speaker to another. In future work, we hope to be able to do this using a classifier that predicts whether the user wants to say something (i.e. take initiative) and whether the user is done talking (and wants to yield control) and inform the response generators to either yield control or take initiative (complementing the user). We imagine that these labels can be inferred from transcripts of speech conversations Jurafsky et al. (1997) and the data be used to train such a classifier. Detecting and responding to multi-turn initiative across all RG's in the bot would lead to deeper and more satisfying conversations.

Latency From the previous year's experience, we found that reducing the latency of responses contributes significantly to a better user experience as measured by user ratings. Intuitively this makes sense – turns between conversing humans typically have sub-second latencies. Hence, throughout our architecture, we have been conscious about making latency improvements wherever possible, such as pre-fetching neural-generation calls, running RG's in parallel and implementing an early-kill mechanism when running RG's (Section 2), as well as distilling the neural-generation model to reduce inference time (Section 5.1). Despite these updates, there is still significant room for improvement before we can attain human-level latencies in conversations. For instance, our latency is still significant enough such that any attempts at backchanneling will severely degrade experience. Further reduction in latency, both in software and hardware, will remain an important step towards an enjoyable conversational experience.

Acknowledgments

Thanks to Abi See for all of her guidance and help; Amelia Hardy for immensely helpful mentorship and advice; and Dilara Soylu for advice and contributions through the Open Chirpy codebase. We thank Amazon.com, Inc. for a grant partially supporting the work of the team. We also thank *The Guardian* for allowing us to use their news API for our system.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. arXiv preprint arXiv:2001.09977.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings. In Proceedings of the 5th International Conference on Learning Representations.

Graham D Bodie, Andrea J Vickery, Kaitlin Cannava, and Susanne M Jones. 2015. The role of "active listening" in informal helping conversations: Impact on perceptions of listener helpfulness, sensitivity, and supportiveness and discloser emotional improvement. Western Journal of Communication, 79(2):151–173.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. arXiv preprint arXiv:2005.14165.

Chun-Yen Chen, Dian Yu, Weiming Wen, Yi Mang Yang, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, et al. 2018. Gunrock: Building a human-like social bot by leveraging large scale real user data. Alexa Prize Proceedings.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555.

Eli Collins and Zoubin Ghahramani. 2021. Lamda: our breakthrough conversation technology. Google AI Blog.

Amanda Cercas Curry and Verena Rieser. 2018. #MeToo Alexa: How conversational systems respond to sexual harassment. In Proceedings of the Second ACL Workshop on Ethics in Natural Language Processing, pages 7–14.

Amanda Cercas Curry and Verena Rieser. 2019. A crowd-based evaluation of abuse response strategies in conversational agents. In 20th Annual Meeting of the Special Interest Group on Discourse and Dialogue, page 361.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2019a. The second conversational intelligence challenge (convai2). ArXiv preprint arXiv:1902.00098.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. arXiv preprint arXiv:1811.01241.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019b. Wizard of wikipedia: Knowledge-powered conversational agents. ArXiv preprint arXiv:1811.01241.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2492–2501.

Starkey Duncan. 1972. Some signals and rules for taking speaking turns in conversations. Journal of Personality and Social Psychology, 23:283–292.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. arXiv preprint arXiv:1805.04833.

Katja Filippova. 2020. Controlled hallucinations: Learning to generate faithfully from noisy data. arXiv preprint arXiv:2010.05873.

Xiang Gao, Yizhe Zhang, Michel Galley, Chris Brockett, and Bill Dolan. 2020. Dialogue response ranking training with large-scale human feedback data. arXiv preprint arXiv:2009.06978.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. arXiv preprint arXiv:1904.09324.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In INTERSPEECH, pages 1891–1895.

Amelia Hardy, Ashwin Paranjape, and Christopher D. Manning. 2021. Effective social chatbot strategies for increasing user initiative. In Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. ArXiv preprint arXiv:1503.02531.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751.

Eric J. Horvitz. 1999. Principles of mixed-initiative user interfaces. In CHI '99: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, pages 159–166.

Dan Jurafsky, Liz Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse function annotation coders manual. In Technical Report Draft 13, University of Colorado, Institute of Cognitive Science.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A. Smith. 2020. Deep encoder, shallow decoder: Reevaluating the speed-quality tradeoff in machine translation. CoRR, abs/2006.10369.

Chandra Khatri, Behnam Hedayatnia, Anu Venkatesh, Jeff Nunn, Yi Pan, Qing Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, et al. 2018. Advancing the state of the art in open domain dialog systems through the Alexa Prize. arXiv preprint arXiv:1812.10757.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880.

Haojun Li, Dilara Soylu, and Christopher D. Manning. 2021. Large-scale quantitative evaluation of dialogue agents' response strategies against offensive users. In Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. CoRR, abs/1907.11692.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In Association for Computational Linguistics (ACL) System Demonstrations, pages 55–60.

Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. 2020. How decoding strategies affect the verifiability of generated text. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 223–235, Online. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 79–84, Copenhagen, Denmark. Association for Computational Linguistics.

Ashwin Paranjape and Christopher Manning. 2021. Human-like informative conversations: Better acknowledgements using conditional mutual information. In North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pages 768–781, Online. Association for Computational Linguistics.

Ashwin Paranjape, Abigail See, Kathleen Kenealy, Haojun Li, Amelia Hardy, Peng Qi, Kaushik Ram Sadagopan, Nguyet Minh Phu, Dilara Soylu, and Christopher D Manning. 2020. Neural generation meets real people: Towards emotionally engaging mixed-initiative conversations. arXiv preprint arXiv:2008.12348.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI tech report.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. arXiv preprint arXiv:1806.03822.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. arXiv preprint arXiv:1811.00207.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5370–5381.

Stephen Roller, Y-Lan Boureau, Jason Weston, Antoine Bordes, Emily Dinan, Angela Fan, David Gunning, Da Ju, Margaret Li, Spencer Poff, et al. 2020a. Open-domain conversational agents: Current progress, open problems, and future directions. arXiv preprint arXiv:2006.12442.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020b. Recipes for building an open-domain chatbot. arXiv preprint arXiv:2004.13637.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. 2019. Masked language model scoring. arXiv preprint arXiv:1910.14659.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR, abs/1910.01108.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. CoRR, abs/1804.04235.

Sam Shleifer and Alexander M. Rush. 2020. Pre-trained summarization distillation. ArXiv preprint arXiv:2010.13002.

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can you put it all together: Evaluating conversational agents' ability to blend skills.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. Computational linguistics, 26(3):339–373.

Alex Wang and Kyunghyun Cho. 2019. Bert has a mouth, and it must speak: Bert as a markov random field language model. arXiv preprint arXiv:1902.04094.

Steve Whittaker and Phil Stenton. 1988. Cues and control in expert-client dialogues. In Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics, pages 123–130.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. ArXiv, abs/1910.03771.

Steve Worswick. 2020. https://medium.com/pandorabots-blog/bot-battle-update-we-won-11fb39ce97cf.

Dian Yu, Michelle Cohn, Yi Mang Yang, Chun-Yen Chen, Weiming Wen, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, Giritheja Sreenivasulu, Sam Davidson, Ashwin Bhandare, and Zhou Yu. 2019. Gunrock: A social bot for complex and engaging long conversations. ArXiv preprint arXiv:1910.03042.

Dian Yu and Zhou Yu. 2019. Midas: A dialog act annotation scheme for open domain human machine spoken conversations. ArXiv preprint arXiv:1908.10023.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019b. Dialogpt: Large-scale generative pre-training for conversational response generation. arXiv preprint arXiv:1911.00536.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 270–278.

Appendix

A Neural Chat Experimental Details

A.1 DistillBlender Training Details

Model distillation. First introduced in Hinton et al. (2015), neural network distillation is a method for training a smaller neural model (in terms of parameter count), called the "student", to replicate the output of a larger model, or the "teacher". This method has been validated on large language

models, notably with the advent of DistillBERT (Sanh et al., 2019). We use the same approach as DistillBERT, as implemented in ParlAI, which initiatlizes the student model layers with weights from selected layers of the teacher model. The model is then trained on a three-part objective consisting of an output-matching distillation loss term, a supervised task loss term, and a cosine embedding alignment term; for details, see the DistillBERT paper.

Training Hyperparameters. We trained with the Adafactor (Shazeer and Stern, 2018) optimizer with learning rate 6.25×10^{-5}, using FP16 training. As a learning rate scheduler, we computed validation loss every 1000 examples and halved the learning rate if validation loss failed to improve for 3 consecutive epochs. Additionally, learning rate warmup took place over the first 100 iterations. We used a batch size of 1 for training on a single V100 GPU.

Decoding Hyperparameters. We use top-k sampling ($k = 5$) with temperature $T = 0.7$. To encourage response diversity across the conversation, we sample sequences of minimum length randomly chosen from 5, 10, 15, 20, 25; in practice, the length of the generations is 0-2 tokens above the minimum selected length.

A.2 Alternative Modeling Experiments

In addition to BlenderBot-3B, we experimented with a variety of autoregressive and non-autoregressive models for text generation. All models are evaluated on an internal dataset of 517 conversation excerpts from early 2021 where the Neural Chat RG was active **(Stanford-AP2021)**. We perform qualitative evaluation by passing in each conversational excerpt to the model of interest, and comparing the resultant generation(s) with the original neural generation from GPT2ED. We detail results here.

DialoGPT. DialoGPT (Zhang et al., 2019b) is a GPT-Medium model that has been further fine-tuned on a set of Reddit threads serving as conversational corpora. We evaluated this model offline on a set of excerpts in two settings: 1) **zero-shot** and 2) with fine-tuning on Empathetic Dialogues **(DialoGPT2ED)**. In the **zero-shot** setting, the bot responds 18% of the time with dirty jokes or memetic content unsafe for open-domain conversation on **(Stanford-AP2021)**. After fine-tuning, **(DialoGPT2ED)** responds almost identically to GPT2ED on **(Stanford-AP2021)**: qualitatively, the lift from DialoGPT2ED is essentially zero. Hence, this system was not deployed.

Last bot utterance	Last user utterance	BARTED	DistillBART-6	DistillBART-3
That's great. How did [school] go?	It was great. It was my favorite day of the week	I'm glad your school day went well. I hope it stays that way.	That's great. I love going to school. I bet you enjoyed it.	Awesome. I wish I could go to school like that.
What is your favorite dish?	ic potatoes with rice [sic]	Oh, that sounds so good. I love mashed potatoes with rice.	Oh, that sounds good. I hope you enjoy it. I love fried potatoes too.	That sounds good. I hope you have a good time!
Perhaps instead, you can tell me about a meal you like to have often?	I like tacos.	That sounds good. I love tacos! It's a good thing you enjoy tacos.	I do like tacos too. I am happy for you, I love tacos.	I like tacos! But I don't think I would like that either!

Table 4: Generations of various BART teacher and student models on three examples in Stanford-AP2021. BART decoding hyperparameters: beam size 4, $T = 1.4$, $p = 0.92$ (nucleus sampling), length 4-20. Capitalization added for clarity.

DistillBART. DistillBART is our in-house distilled version of BART Lewis et al. (2019), a model consisting of a non-autoregressive encoder and an autoregressive decoder, each with 12 layers. Notably, this model has decoding complexity $\mathcal{O}(EN + DN^2)$, where N is the sequence length, and E, D are the sizes of the encoder and decoder stacks, respectively. Following results by Kasai et al. (2020) in the domain of neural machine translation, we hypothesized that we could decrease latency while improving performance by decreasing D; i.e. removing decoder layers and training the decoder via distillation. We performed DistillBERT-style distillation, distilling a BART-Large fine-tuned on Empathetic Dialogues (BARTED) into versions with 6 **(DistillBART-6)** and 3 **(DistillBART-3)** decoder layers. Weight initialization followed a previous setup for BART distillation (Shleifer and Rush, 2020). As baselines, we also trained equivalently-sized models without distillation.

In practice, BART suffered from 1) high latency and 2) mediocre response quality. BART was unable to generate coherent responses stochastically, necessitating the usage of beam search, which hurt decoding speed. On Stanford-AP2021, average decoding speeds for the 12, 6, and 3 layer models were 894ms, 998ms, and 895ms, showing no significant latency gains, which is attributable to the quadratic dependence within the decoding computation on sequence length; i.e. $N^2 \gg D, E$. Furthermore, while distillation certainly resulted in qualitatively better generations on Stanford-AP2021 than those of non-distilled models, as shown in Table 4, there was a sharp dropoff in generation quality on all models except the full-sized BARTED teacher. As BARTED was the only usable model, and yielded generations qualitatively similar to GPT2ED, we did not deploy this system.

B High-Initiative User Handling: Example Dialog Flows

Our system uses high-initiative handler functions to detect 18 distinct types of one-turn digressions that interrupt the conversational flow. General examples include complaints and clarifying questions. A comprehensive list of high-initiative situations that our bot can handle and corresponding examples of bot responses are provided in Table 5. These functions can be started in any RG, and allow each RG to pause the flow of the current dialogue tree to handle high-initiative behaviors before either returning to the original conversation or changing the subject.

To avoid repetitiveness, high-initiative handlers are generally non-reentrant, and cannot be triggered consecutively.

C Infilling details

Below, we describe more details as to template-based infilling.

C.1 Methods

Let $\mathcal{E}$ be some entity. We want to generate topical sentences from a set of templates $t_1, \ldots, t_n$ from knowledge about this entity, which we have as statements $\{k(\mathcal{E})^{(1)}, \ldots, k(\mathcal{E})^{(n)}\}$. We generate a grounded statement via the following steps:

- **Knowledge retrieval**: we retrieve a set of statements $k_1, k_2, k_3, \ldots$ and match them to templates $t_1, t_2, t_3, \ldots$ based on some criterion of relevancy/topicality. Currently, this is done via a vector similarity-based search.
- **Infilling**: For each template/knowledge pair (t_i, k_i), we generate a **completion** c_i, which is a possible full response.
- **Reranking**: The set of completions is reranked using several criteria, depending on the use case. The response is then returned as the final output.

Importantly, each of these steps can be performed without significant training on a large dataset. We were able to train our infilling model with only ~ 4286 manually-curated and GPT-generated examples.

C.2 Knowledge retrieval

In this step, given a large amount of information (e.g. from Wikipedia), we want to retrieve a certain amount of *condensed* information that (a) fits in the context window of the infilling model and

High-Initiative User Behavior	User utterance	Possible response
Was misheard	That's not what I said.	Sorry for the misunderstanding; can you repeat that?
Complains bot was unclear	I don't know what you are talking about.	Oops, let me say that again. [PREV]
Complains bot was repetitive	Alexa, you already told me that.	You're right, I forgot. Sorry. Let's move on to something else. [NEW_TOPIC]
Complaints about privacy	That's none of your business.	Sorry, maybe that was too personal. I'm happy to talk about something else. [NEW_TOPIC]
Asked about weather	Do you like the weather?	I live in the cloud so I'm not sure what the weather is like on earth!
Said Alexa command	Alexa, play Despacito!	This is an Alexa Prize Socialbot... I can't execute typical Alexa commands...But I'd love to keep talking to you! As I was saying, [PREV]
Wants name correction	Actually, that's not my name.	Oops, it sounds like I got your name wrong. I'm so sorry about that! I won't make that mistake again.
Requested repetition	Alexa, can you say that again?	Oops. Let me repeat that. [PREV]
Requested own name	What's my name?	If I remember correctly, your name is [NAME]
Got cut off	But—	Oops, I think my microphone stopped working for a second. Can you say that again?
Asked for our age	How old are you?	Hmm, I don't know. I'm just a bot, and I don't think we have birthdays.
Asked clarifying question	Wait, did you say "elephants?"	Yes, that's what I said.
Questioned bot's humanity	How can you listen to music if you're a bot?	Well, I'm just a bot, but I live vicariously through others.
Asked personal question	What do you like?	Thanks for asking! [NEURAL]
Interrupted with question	Wait, I have a question.	Sure, what's up?
Expresses desire to chat	Let's chat!	Okay. What would you like to talk about?
Asked for story	Tell me a story.	Here's a story that someone once told me. [STORY]
Gave bot compliment	I like talking to you!	Thanks for saying that. I'm still learning, but it means a lot to hear that from you.

Table 5: Excerpts of possible bot responses to 18 high-initiative user behavior modes. [PREV] refers to the previous bot utterance; [NEW_TOPIC] refers to a sampled prompt from a new RG. [NAME] is the user's name as obtained in the opening turns, and [NEURAL] refers to a DistillBlender-based random response. [STORY] is a handwritten anecdote, omitted here for brevity.

(b) relates to the templates that we have on hand for the current entity. Given an entity brought up by the user, we first retrieve the set of possible templates by matching the entity's WikiData categories against a pre-written set of templates. We then bucket the total knowledge into windows of W sentences (in our experiments, we use $W = 3$). For each window, we compute the sentence embedding using the method of Arora et al. (2016). We then compute the pairwise dot product similarity between window embeddings and template embeddings. Finally, the top k (template, knowledge) pairs are used as input to the infilling phase.

C.3 Template embedding generation

We experiment with various methods for template embedding generation. We first used Arora et al. (2016)'s method to generate an embedding directly from the template; however, this resulted in poor results due to domain shift. Instead, a more effective strategy was to provide a set of fuzzy keywords for each template. For example, for the template

"[city]'s most famous landmark is [thing]."

we annotate the following keywords: *{"city", "landmark", "visit", "site", "tourist"}*. To calculate the similarity between a knowledge embedding and the template, we then calculate the pairwise similarity between the knowledge embedding and each keyword, taking the mean of the top K similarities; this allows for multimodality in the use of a template.

This approach is a quick-and-dirty solution: it can easily be done intuitively by human annotators in a limited amount of time, and allows for simple retrieval that usually succeeds. However, it is inflexible; in our preliminary quantitative results, we found that it prioritizes areas with high knowledge density over those that are more interesting and novel. We discuss considerations for future models in our Discussion.

C.4 Knowledge selection

In the setting of an Alexa Prize socialbot, our infilling algorithm can draw knowledge from any source; in our implementation, we rely on Wikipedia. However, Wikipedia articles, especially those for popular entities, can be very long. This can lead to two problems:

- The knowledge retrieved may be very specific. For example, when discussing *Italy*, the retrieval model is likely to retrieve information about specific Renaissance-era artists for the *culture* template.
- The knowledge retrieved may be irrelevant to the current subject of discussion. If the model retrieves an arbitrary template every turn, this usually leads to a disjointed and confusing conversation in a multi-turn dialogue setting.

To solve this, we only retrieve from a particular Wikipedia *section* at a time. During the first turn of the conversation, we retrieve from the intro section if it exists in our datastore; this allows us to make general statements, at the cost of specificity. On the next turn, we select the section with the highest TF-IDF overlap with the user utterance; this allows for some level of topical 'acknowledgement' of the user utterance.

C.5 Neural Infilling

In this step, given a textual context and a template, we want to generate an infilled version of the template. To do so, we apply a fine-tuned BART model Lewis et al. (2020). As input, we provide the context and template separated by a `[SEP]` token; for example, for the above example, we would provide:

"I love **[entity]** because of **[his/her]** album **[album]**, which **<mask>**. **[SEP]** "Adele Laurie Blue Adkins MBE is an English singer-songwriter. In 2007, she received the Brit Awards Critics' Choice Award and won the BBC Sound of 2008 poll. Her debut album, 19, was released in 2008. It is certified 8× platinum in the UK and triple platinum in the US. **[SEP]**"

Email and file sharing are very useful Internet services that I use daily.	[Product] is a very useful service that I use [time].	Email and file sharing ;; daily
Do you know what the New Zealand basketball team is called?	Do you know what the [location] [sport] team is called?	New Zealand ;; basketball
I'm a really big fan of the Beatles. I especially love their song Penny Lane.	I'm a really big fan of [band]. I especially love their song [song].	The Beatles ;; Penny Lane
Superman is a fictional superhero, me also like batman and spider-man.	[Character] is a fictional [occupation], I also like [character].	Superman ;; superhero ;; batman and spider-man
Computers do so much now, not just the original use as a calculating device.	Computers do so much now, not just the original use as a [purpose].	calculating device
I'm a big fan of Napoleon.	I'm a big fan of [person].	Napoleon

Table 6: The few-shot priming that we use to generate templates using GPT-3.

The model is then trained to predict the entire infilled template. This is largely a copying operation; for example, in the above example, the tokens "I love . . . because of . . . album . . . which . . . ." can be copied directly from the input.

Infilling We note that the infilling procedure is very similar to the denoising operation of a masked language model (Devlin et al., 2018b; Ghazvininejad et al., 2019). However, since we use a `seq2seq` language model, we can generate arbitrarily large numbers of tokens per position, allowing for more flexible generations; for example, we are able to generate "**was released in 2008**" from a single **<mask>** token, something that wouldn't be possible without Gibbs sampling-style decoding (Wang and Cho, 2019; Salazar et al., 2019); additionally, it is possible to not infill a slot at all.

Dataset We generate ~ 4143 training pairs using GPT-3 (Brown et al., 2020), a large pretrained generative language model capable of few-shot contextual learning. First, we take utterances from the Topical Chat dataset (Gopalakrishnan et al., 2019), a dataset of human conversations grounded in Wikipedia entities. We match the utterances to Wikipedia paragraphs using TF-IDF similarity score, taking only paragraph-entity pairs with TF-IDF similarity between 0.04 and 0.08.

For each utterance, we generate two templates using the few-shot priming in Table 6, only keeping distinct and non-erroneous templates (i.e. the number of generated infills equals the number of slots in the template). Since the task is relatively deterministic, we use a temperature of 0.3. During training, the model is then trained to infill the template back to the original utterance, conditioned on the Wikipedia paragraph.

Manual data augmentation Although generating data this way is easy, we find that it suffers from domain shift. Despite the fact that our priming sentences mostly have several slots, as do the templates we use in practice, most of the templates (94.3%) generated by GPT-3 only have one slot; all but 3 of the remainder only have two slots. The reason for this shift is unclear.

To ameliorate this issue, we manually augment the data with multiple hand-written examples drawn from the Wizard of Wikipedia dataset (Dinan et al., 2018). Since most utterances in this dataset simply rephrase the original knowledge statement, we create our own utterances which are more similar to the templates we use in practice. We create 143 of these examples, fine-tuning our model on both the handwritten examples and the GPT-3 examples without any weighting.

The <mask> token By convention, we use **[bracketed words]** to represent single-word slots; we use the **<mask>** token to represent a larger amount of text to be copied from the input. We draw this directly from BART training, which uses **<mask>** to represent a multi-word denoising objective. Since there are no **<mask>** tokens in our GPT-3-generated dataset, we manually write templates involving **<mask>** as part of our 143 examples. (Note: one can use BART to infill slots in templates zero-shot due to this denoising objective, but our preliminary experiments suggested that this was typically unsuccessful due to being trivial/uninteresting infills.)

C.5.1 Implementation details

Fine-tuning We fine-tune the BART-base model (139M parameters) on the set of 4286 examples generated using GPT-3 using the Adam optimizer (Kingma and Ba, 2014) with learning rate $1e-5$ and otherwise default parameters. We use the HuggingFace reimplementation for all models. We do no validation due to limited data, stopping after 3 epochs to avoid overfitting.

Decoding We decode using beam search with beam size $= 2$. Empirically, we found that using greedy search (beam size $= 1$) can lead to failure to copy correctly due to high-probability, incorrect subwords being copied.

C.6 Reranking

Once the infiller has finished, we have a set of possibly topical utterances generated based on knowledge about the current entity. We now need to select a coherent, interesting response that is related to the user's previous subject of conversation. To do so, we calculate the mean per-token negative log-likelihood of each statement using a GPT-2 medium model fine-tuned on the EmpatheticDialogues dataset (Rashkin et al., 2018), as well as its absolute score using the DialoGPT updown reranking model (Zhang et al., 2020), which is trained on upvote/downvote data from Reddit. We then rerank using the following formula:

$$\text{score} = \text{score}_{\text{DialoGPT}} - 0.25 \times \text{NLL}_{\text{GPT}}.$$

where the DialoGPT score is a probability between 0-1 and the NLL tends to be around -2. Although this is not entirely mathematically sound (adding what is *technically* a probability and a log-likelihood), it seems to work well in practice. Specifically, the DialoGPT reranking score seems to select for interesting utterances with novel information, while the GPT likelihood seems to select for fluency (rather than numerous noun phrases put together).

C.7 Results

We find that our method achieves good qualitative results; however, we defer human evaluation to future updates.

Here are some examples:

- Template: I love [song] because of its [quality]. Infill: I love "Someone Like You" because of its beautiful lyrics.
- Template: The best qualities of [singer]'s voice are **<mask>**. Infill: The best qualities of Adele's voice are its power and range.
- Template: [singer] has won awards such as **<mask>** for her songs. Infill: Adele has won awards such as a Grammy and an Academy Award for her songs.
- Template: I still remember [year], when it seemed like [song] was on every [genre] station. Infill: I still remember 2009, when it seemed like Adele's "Chasing Pavements" was on every pop station.
- Template: I really like [artist] because she is so [adjective]. Infill: I really like Adele because she is so critically acclaimed.
- Template: My favorite album by [artist] is [album] because it's so [adjective]. **<mask>** Infill: My favorite album by Adele is 21 because it's so critically acclaimed. It's won six Grammys and an Academy Award.
- Template: I've been following her since she first started [action] at [place] Infill: I've been following her since she first started performing at the Bluebird Cafe in Nashville.
- Template: Do you remember when **<mask>**? Infill: Do you remember when Adele won a Grammy for 21?
- Template: I hear that [group] released the album [name]. Infill: I hear that Adele released the album 25.

D ASR Details

In order to mitigate ASR errors in entity names, we first pre-process the anchortexts of all Wikipedia entities in our ElasticSearch database, to obtain their phoneme and metaphone representations via the `g2p-en`[15] and `metaphone`[16] libraries respectively.

Then, given an ASR transcript, we likewise convert all spans $s \in S$ to their phoneme and metaphone representations, before querying the ElasticSearch index for entities with similar representations. We then define the distribution of spans s given an entity E, $P(s|E)$, as follows: for each anchortext

[15] `https://pypi.org/project/g2p-en/`

[16] `https://pypi.org/project/metaphone/`

$a \in A_{\text{phon}}$, we find its best-matching span $s^*(a) = \arg\max_{s \in S} \text{sim}(s, a)$ where $\text{sim}(\cdot, \cdot)$ is a phoneme similarity function. Then we filter out anchortexts that are too phonetically different with a threshold of 0.8, $A(s) = \{a | a \in A_{\text{phon}}, s = s^*(a), \text{sim}(a, s) \geq 0.8\}$. Finally:

$$P(s|E) \propto \begin{cases} \max_{a \in A(s)} \text{count(links from } a \text{ to } E) \times \text{sim}(s, a) & A(s) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The same algorithm is described in more detail in Section 4.4 of Paranjape et al. (2020), which is in turn inspired by a similar method in Chen et al. (2018).

E Details of Models in News RG

Text Summarization The summary of the stories are generated by a Pegasus model (Zhang et al., 2019a) which was trained on the MultiNews dataset (Fabbri et al., 2019). After an extensive parameter search we found that using 8 beams and a maximum of 50 tokens at generation time gave succinct and efficacious results. If it is not possible to generate an effective abstractive summary for an article using Pegasus, an extractive summary using the TextRank algorithm (Mihalcea and Tarau, 2004) is generated instead.

Question Answering We use an ELECTRA-Large model (Clark et al., 2020) pretrained on SQuAD2.0 (Rajpurkar et al., 2018) to extract a set of $n = 5$ plausible answers and the sentences containing the answers along with the confidence scores from the news story. After filtering out unqualified answers such as empty strings, low confidence answers and answers that are too long, we randomly select an answer a and its corresponding span sentence s based on the normalized confidence score as the selection probability.

Conversational Paraphrasing We use a finetuned GPT-2-medium language model (Radford et al., 2019) on a processed and filtered version of the TopicalChat dataset (Gopalakrishnan et al., 2019). The paraphrases are generated using top-p decoding with $p = 0.8$ and temperature $\tau = 0.7$.