DREAM technical report for the Alexa Prize 2019

Yuri Kuratov, Idris Yusupov, Dilyara Baymurzina, Denis Kuznetsov, Daniil Cherniavskii, Alexander Dmitrievskiy, Elena Ermakova, Fedor Ignatov, Dmitry Karpov, Daniel Kornev, The Anh Le, Pavel Pugin, Mikhail Burtsev

Neural Networks and Deep Learning Lab
Moscow Institute of Physics and Technology
yurii.kuratov@phystech.edu, i.yusupov@phystech.edu,
dilyara.baymurzina@phystech.edu, kuznetsov.dp@phystech.edu,
burtcev.ms@mipt.ru

Abstract

Building a dialogue system able to talk fluently and meaningfully in an open domain conversation is one of the foundational challenges in the field of AI. Recent progress in NLP driven by the application of the deep neural networks and large language models opened new possibilities to solve many hard problems of the conversational AI. Alexa Prize Socialbot Grand Challenge gives a unique opportunity to test cutting edge research ideas in the real-world setting. In this report, we outline the DREAM socialbot solution and present evaluation results. DREAM socialbot is implemented as a multi-skill conversational agent with the modular micro-service architecture. DREAM agent orchestrates a dozen text preprocessing annotators and more than 25 conversational skills to generate responses in the context of the open domain conversation. Feedback from Alexa users during the evaluation period allowed us to gradually develop our solution by increasing the number of conversational skills and improving the transition between them. As a result, dialogues became 50% longer, and average rating grew from ~ 3 during the initial stage in December'19 to ~ 3.4 during the last two weeks of April'20. The final version of DREAM socialbot is a hybrid system that combines rule-based, deep learning, and knowledge base driven components.

1 Introduction

Today state-of-the-art socialbots and voice assistants share common architecture consisting of multiple skills and a skill manager. Skills are usually implemented as scripts, ranking, or generative trainable models. Skill manager is usually hand-crafted and rule-based, but there are several attempts to make it trainable (e.g., HCN by Alquist [28], topic classifiers by Gunrock [6]). Such design of a dialogue agent looks reasonable in general but still fails to ensure coherent, relevant, and engaging open-domain conversation due to a number of reasons. While scripted skills might provide a coherent dialogue flow, they can do that but only in a very narrow domain, and with poor language variability. Therefore, only the most popular topics of very rich social conversational interactions can be partially covered by scripts. Besides that, templated replies can make interaction boring. On the other hand, generative models can potentially produce fun and engaging phrases but suffer from shallow context understanding, which breaks the meaningfulness of the dialogue as a whole. Ranking models lie somewhere in between with rather high probability of relevant reply but limited domain coverage and depth of the context. All these shortcomings make the task of skill selection very difficult. The skill manager has to balance a coherent but narrow domain and probably boring skills, with broad but

3rd Proceedings of Alexa Prize (Alexa Prize 2019).

sometimes meaningless skills. As a result, skill switching errors are common and frequently shift the direction of conversation irrelevantly.

Recent advances in NLP, such as language models pre-training [8, 27, 29, 10], memory-based architectures, and new conversational datasets [15, 41, 12, 32, 7] give hope to alleviate the majority of the issues described above. Transformer based language models can be easily fine-tuned for almost any conversational task and demonstrate a significant boost in performance. This improves all NLP preprocessing pipelines such as NER, POS-tagging, coreference resolution, as well as ranking models [10, 25, 26] thus making the overall output of a dialogue system more relevant.

Building upon the latest progress in the NLP field, we propose a multi-skill architecture for the dialogue agent that combines state-of-the-art ML models with the modular micro-service pipelines into a scalable asynchronous architecture.

2 DREAM Socialbot System Design and Architecture

DREAM socialbot is implemented and served with DeepPavlov[1] and DeepPavlov Agent[2] frameworks.

DeepPavlov library [4] allows us to describe the text processing pipeline in a declarative style as a series of steps by writing a configuration file. It also provides a number of pre-trained NLP models, including the latest transformer architectures. The library includes a number of predefined pipelines for the most common tasks. Any pipeline can be easily run in the REST API mode, making it a good choice for modular systems with micro-service architecture.

DeepPavlov Agent is a framework designed to facilitate the development of scalable and production-ready multi-skill virtual assistants, complex dialogue systems, and chatbots. Key features of DeepPavlov Agent include (1) scalability and reliability in the high load environment due to micro-service architecture; (2) ease of adding and orchestrating conversational skills; (3) shared dialogue state memory and NLP annotations accessible to all skills. DeepPavlov Agent orchestrates the following types of services:

- `Annotator` is a service for NLP preprocessing of an utterance. It can implement some basic text processing like spelling correction, named entity recognition, etc.;
- `Skill` is a service producing a conversational response candidate for a current dialogue state;
- `Skill Selector` is a service that selects a subset of the available skills for producing candidate responses;
- `Response Selector` is a service that picks the best response out of the available candidates to be sent to the user;
- `Postprocessor` is a service that is responsible for the postprocessing of the response utterance. It can make some basic things like adding a user name, inserting emojis, etc.
- `Dialogue State` stores current dialogues between users and a conversational agent as well as annotations and other meta-data serialized in JSON format. The state supports sharing of stored information across the services.

Detailed description of DREAM services can be found in Section A of the Appendix.

DeepPavlov Agent allows creating dialogue systems with flexible, dynamic, and asynchronous pipelines. The principal architecture of DREAM socialbot in the DP-Agent framework is presented in Figure 1.

DREAM socialbot solution has a modular design with the main components such as `annotators`, `skills` and `selectors` run as independent services. These components are configured and deployed using Docker[3] containers. It allows us to focus on application development instead of focusing on the intrinsic details of the manual low-level infrastructure configuration. Infrastructure and deployment details are provided in Section B of the Appendix.

[1] `https://deeppavlov.ai`

[2] `https://github.com/deepmipt/dp-agent`

[3] `https://www.docker.com/`

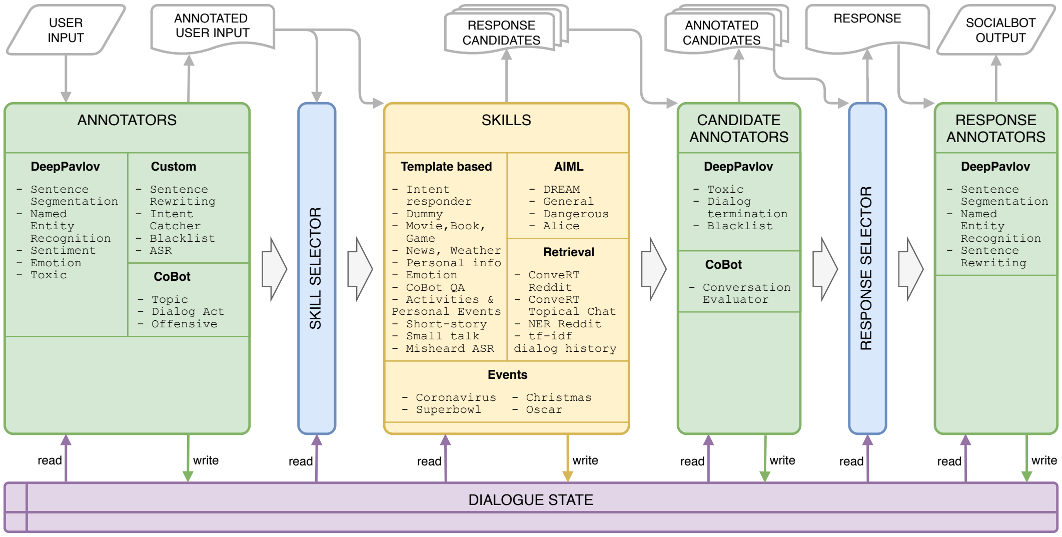


Figure 1: **DREAM socialbot architecture.** Multiple `Annotators` are used to extract information from the user input. `Skill Selector` defines a subset of active `Skills` based on the extracted information. Selected `Skills` propose their response candidates. Finally, `Response Selector` picks a response to be sent to `Response Annotators` and, eventually, to the user. All elements of the pipeline are running asynchronously with two points of synchronization: `Skill Selector` and `Response Selector`. `Dialogue State` serves as a shared memory.

Designing and running a high-quality socialbot is a challenging task for both academic and industrial organizations. In industrial settings the end-user rating of conversational experience is studied mostly through the special beta programs and limited UX lab studies. User rating is considered only as a one of the many key performance indicators used to track product success. A unique specifics of participation in the Alexa Prize competition requires designing analytical infrastructure around the end-user ratings only. We developed a set of tools to track ratings of dialogues and to perform in depth analysis of system's behavior for every turn in a conversation. Description of the analytic tools is presented in Section C of the Appendix.

3 DREAM socialbot Evaluation Results

Over the course of the contest, we made a number of decisions that might significantly contribute to the average daily rating. Based on the timings of decisions and changes in rating, we have identified 11 phases, each of which had its own specifics.

As shown in Figure 2, the first phase (Dec 3-24) has an average rating of 3.01. During this phase, the number of the active skills climbed from 7 to 13, as shown in Figure 3. These skills include retrieval based, like basic TFIDF, Music, and `ConveRT Reddit Retrieval Skills`, as well as `Book Skill`, `Weather Skill`, and `Christmas Skill`. Combined together, they helped to increase the breadth of the covered topics significantly, as well as to increase the average dialogue rating to 3.19. The primary focus of the team was on the task of making the agent more proactive. A limited number of topics were covered by the rule-based and scenario-driven skills such as `Books skill` and `Christmas Skill`, while the rest of the topics were addressed by the retrieval skills.

In the second phase (December 24 - January 10) the average rating was 3.19. This period overlapped with Christmas and New Year Holidays, and no major changes introduced to the socialbot. Work on bug fixing during the previous phrase together with the introduction of the `Christmas Skill` was a major contribution to the average rating growth (+0.18) in comparison with the previous phase.

The third phase (Jan 10-27) brought a serious hit. The average rating decreased to 2.97 (-0.22). DREAM socialbot was disabled twice after having sequences of dialogues with low ratings. A few more skills were added during this phase: retrieval based `TopicalChat ConveRT Retrieval Skill`, and rule-driven `Eliza` and `News Skill`. `Intent Catcher` has been significantly improved,

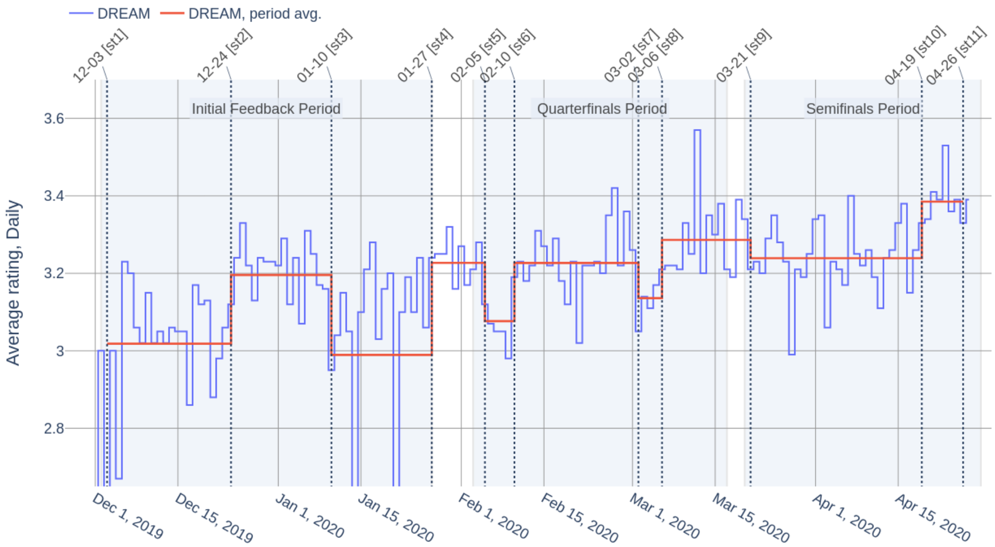


Figure 2: **Average daily DREAM Socialbot rating.** Daily rating is in blue. Vertical dotted lines separate different stages of DREAM socialbot development. Solid red line shows average rating during the stage. Shaded area corresponds to different phases of the competition.

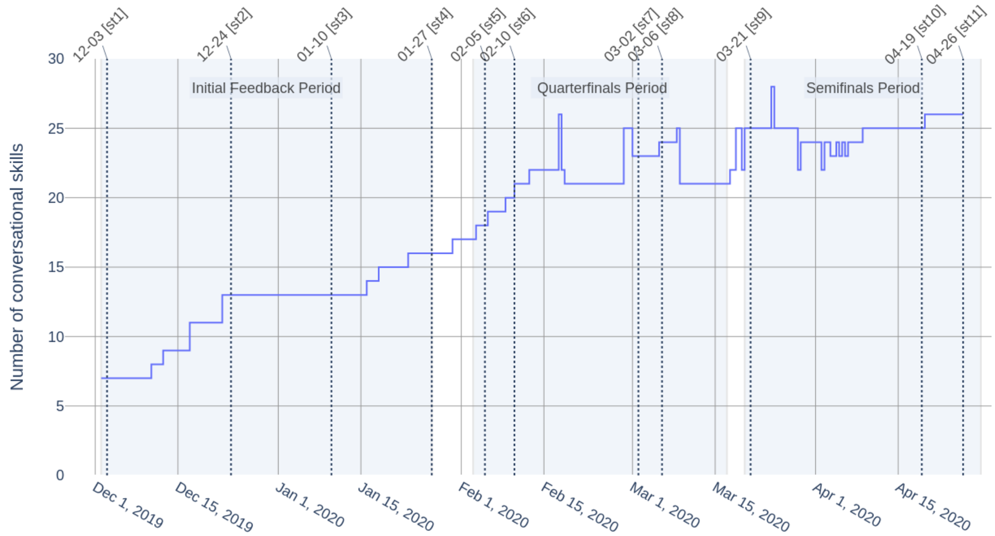


Figure 3: **Number of conversational skills in the DREAM socialbot.** Majority of skills were added till the Quarterfinals and then the focus shifted on delivering smooth dialogue flow and topic switching by improving `Response Selector`, `link to` mechanism and in-depth improving of existing skills.

and a new topic switching intent has been introduced. Another focus of the team was enhancing analytical tools to enable better dialogues and rating analysis.

The fourth phase (January 27 - February 5) was a time between the Initial Feedback Period and the Quarterfinals Interaction Period. Contributions to the socialbot quality led to the average rating growth to 3.22 (+0.25). A new `SuperBowl Skill` has been added and `TFIDF-retrieval Skill` has been significantly improved by adding human response to it.

In the fifth phase (Feb 5-10), the average rating dropped to 3.07. We have the hypothesis that external events such as developing awareness and concern about pandemics might be the cause. During this period, the team was focused on finishing the development of the `Activity Discussion Skill`. Two more skills were also added, including event-specific `Oscar Skill`, as well as a more broad

`Emotion Skill`. These contributions, as well as continued work on bug fixing, led to an increase of average dialogue time at the beginning of the next stage.

The following phase (February 10 - March 1) showed growth of the average rating back to 3.22. During this phase a few more skills have been added, including `Valentine's Day Skill`, `Activity Discussion Skill`, as well as `NER-Skill on Reddit`. Adding the `Activity Discussion Skill` led to a notable growth of average dialogue length as shown in Figure 4. Upon a deeper analysis of the dialogues, the team made a strategic decision to focus on linking dialogue parts with each other to improve the overall user experience further. This phase also got a further increase of negative users utterances (see in Figure 5), which could also be seen as a consequence of the growing pandemics and a continued mood decrease among the US population. However, it is important to note that this increase in the negative sentiment did not lead to the trend for the average rating decrease.

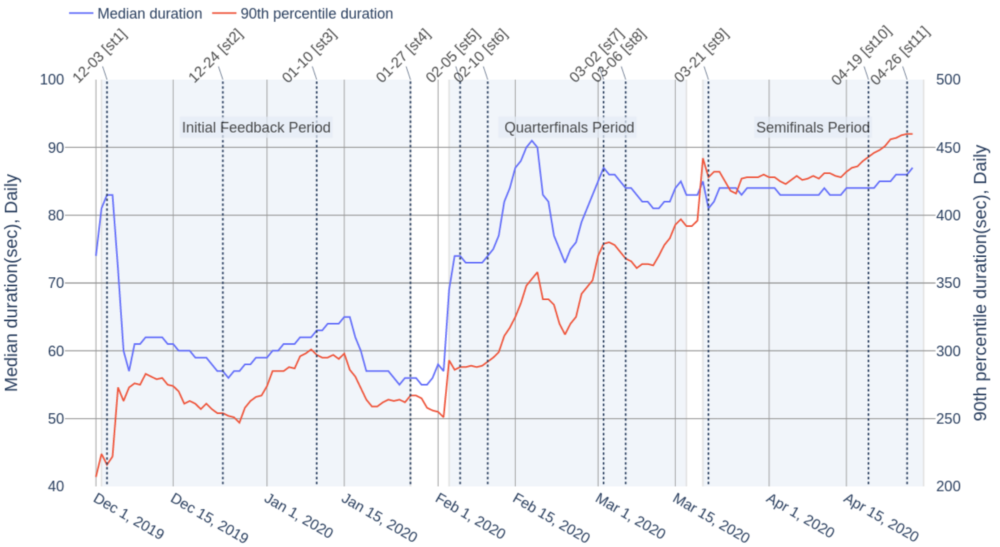


Figure 4: **Daily Average Duration of Conversations.** Median duration is shown in blue (axis on the left) and 90th percentile duration in red (axis on the right). Interactions with the Alexa users began in Quarterfinals Period. Since then, 90 percentile of conversations duration increased from about 300 to 450 seconds in 2.5 months.

Phase 7 (Mar 2-6) got another rating hit. This time unsuccessful deployment of the agent updates led to the incorrect functioning of the `AIML DREAM Chit-Chat Skill`. Also, the updated `TopicalChat ConveRT Retrieval Skill` had a very high confidence level, which led to the unexpected growth in the number of its responses with low quality.

The next phase, Phase 8 (Mar 6-21), was a time of transition from the Quarterfinals Interaction Period to the Semifinals Interaction Period. The average rating was 3.28. This phase was devoted to critical bug fixing; no new skills or other components were added to the DREAM agent during this period.

Phase 9 (March 21 - April 19) had high variability in daily ratings and a slightly lower rating of 3.24. During this time period, we added a new `Small Talk Skill`, as well as made another strategic step. We decided to run A/B experiments to facilitate the growth of the agent's quality. An A/B testing infrastructure has been deployed. During this period, a series of risky experiments were run, which lowered the ratings, however, the learnings made from these experiments allowed to stabilize levels of positive and negative sentiment. On April 9, we released script-based version of `Movie Skill` for particular movies discussion improving it from one-turn opinion-expression version. From April 13, we started to improve scenario-driven skills actively and introduced new `link-to` functionality (see Section A.3.1) to enable a smooth transition between skills during the dialogue. We link these changes to the significant growth of the positive user utterances that can be seen in Figure 5.

Phase 10 (April 19-27) saw a serious growth of the average rating to 3.39. A new scenario-driven `Game Skill` has been added to the system. Risky experiments were postponed, and the best versions from the previous period were selected and run.

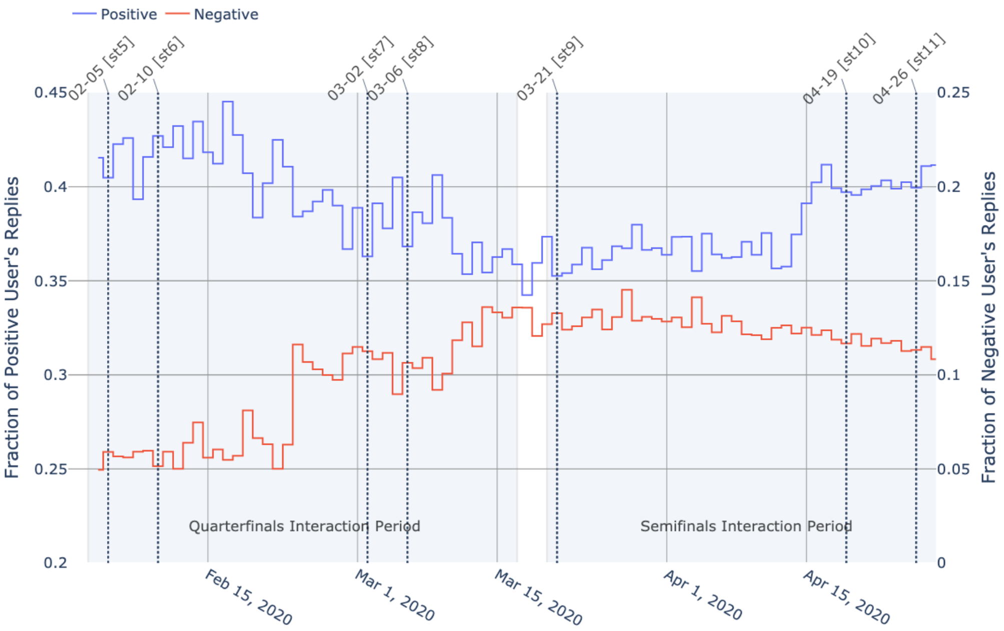


Figure 5: **Daily fractions of user utterances with positive and negative sentiment.** All user utterances were classified into three classes: positive, neutral, and negative. Fraction of positive utterances is shown in blue (left axis) and negative in red (right axis). Prior to February 4 we used a different sentiment analysis model, so this plot only reflects sentiment analysis results we have collected after this change.

4 Selected Science and Technology Contributions

4.1 Conversational Skills with Common Sense

Lack of common sense is one of the most challenging problems for conversational AI today. The good mood of a user earned over the journey along nice scripted sections of the dialogue could be easily broken if the system is unable to answer a "simple" question that requires a basic understanding of the human world. In DREAM socialbot, we explored the possibility of using knowledge graphs to inject commonsense reasoning into the conversations.

`Activity Discussion Skill` briefly described in Subsection A.3.4 simulates motivation of the socialbot to understand human world better. For doing this, the skill seeks help from a user for an explanation of some human activities. Therefore, if a user wants the socialbot to choose the subject of the dialogue, `Activity Discussion Skill` asks about one of the predefined activities, like skydiving, geography, baking. Also, the activity in the form of *verb* + *noun* pair is extracted from user utterances or from the already told news. If the action in the form above was not found, for each noun from the user utterance, we look for a bigram in the vocabulary collected from a large amount of text in English.

The discussion of activities consists of the starting phrase, several questions, and opinion request. The starting phrase either selected from the pool of hand-written templates for some activity related Wiki topics or could be a direct request to explain something. If the user does not refuse to explain the activity, the socialbot asks several clarification questions.

Clarification questions are composed with the help of COMeT Atomic [2] model. The model can generate predictions for the following aspects: *"xAttr"* - what person feels during the activity, *"xIntent"* - what person wanted to get during the activity, *"xNeed"* - what person needed for the activity, *"xEffect"* - what is the result of the activity, *"xReact"* - what person feels as a result of the activity, *"xWant"* - what person wanted to be the result of the activity.

Consider the following example. For the activity *"practice yoga"* the model generates the following common sense aspects for *"xIntent"*: "to be healthy", "to learn yoga", "to relax". Therefore, we can build a question "Is that true that people practice yoga to be healthy?". We expect that `Activity`

Discussion Skill as a part of the dialogue can help user to feel more confident when talking to the socialbot while also to be more loyal to it, understanding its incompetence in some topics. There is also a variety of other applications of COMeT.

Another, *Personal Event Discussion Skill* simulates a chat about user's activities in terms of intents, feelings, effects, and consequences. This skill works in two modes.

In the first mode, if the skill extracts user's action in the form *I + verb + ...*, it then randomly selects a template to ask a related question or comment depending on the verb tense of the user's action. Then the skill sends a request to the COMeT Atomic model to generate assertions of common sense about extracted action to fill out the template of the question. For example, if user says *"I will go to the theater next weekend."* and for *"xNeed"* query, the model returns *"buy the tickets"* then the template-based question might be "Did you buy the tickets?". For a comment with predicted assertions *"happy"*, *"excited"*, *"entertained"* for relation *"oFeel"* the skill can generate the sentence "I feel happy for you!". This part of Personal Event Discussion Skill underlies the socialbot's ability to trace cause and effect relationships, and to establish some emotional connection with the user.

The second mode of Personal Event Discussion Skill returns scripted opinion about the given object that depends on the sentiment of the selected common sense assertion. For example, if user requests opinion about *cats*, and the skill randomly selects template which is based on *"SymbolOf"* common sense assertion then COMeT ConceptNet [34] model generates predictions like *"love"*, *"peace"*, *"innocence"*. The skill takes these predictions and composes an opinion about cats: "I adore cats! For some of us, cats can be seen as a sign of love.". This enables socialbot to express a reasoned opinion on a wide variety of objects excluding sensitive topics.

4.2 Trainable Response Selection Model

Throughout most of the competition Response Selector selected final response with heuristics on top of output from Candidate Annotators. To improve the quality of Response Selector, our team labeled 3400 response candidates from ~ 400 unique dialogue contexts with two classes appropriate response (positive) or inappropriate response (negative). For each dialogue context, multiple candidates could be labeled as positive. As a result, we built a dataset with ~ 750 positive and ~ 2650 negative examples.

Heuristic Baseline is a weighted sum of skill confidence and predictions from Conversation Evaluator. Additionally, it filters response candidates with Toxic Classifier, Dialog Termination annotator and Blacklist Word Detector. As another option, we tried a grid search to adjust weights and thresholds on labeled data (Heuristic Baseline + Grid Search in Table 1).

We used 17 features to train LightGBM[4] Gradient Boosting model [19]: skill confidence (1), outputs from Conversation Evaluator (5), Toxic Classifier (7), Dialog Termination (1), and Blacklist Words Annotator (3). We have also experimented with Textual Entailment (TE) models available at AllenNLP Demo[5] for two last utterances as a premise and response candidate as a hypothesis. Textual Entailment models output probabilities for three classes (entailment, contradiction, and neutral). This allowed us to add 9 more features from three Textual Entailment models: Decomposable Attention + ELMo on SNLI (3), RoBERTa on SNLI (3), and RoBERTa on MultiNLI (3).

Results from Table 1 show that Gradient Boosting models out-performed our baselines and TE features improve the quality of Response Selector slightly further. Currently, we do not use TE models in Response Selector because of the significant computational burden imposed by the RoBERTa-Large model compared to a small gain in metrics with TE features.

4.3 Custom Named Entity Recognition and Sentence Segmentation Models

The model used for the NER was optimized to exploit useful features in the context of the task, including (1) pre-trained word embeddings, (2) character-level features, and (3) contextual word features as well. The word vector representation is created by concatenating (1) GloVe pre-trained word embedding [17], (2) ELMo word embedding [27], and (3) character-level word embedding

[4] https://github.com/microsoft/LightGBM

[5] https://demo.allennlp.org/textual-entailment

Model	Correlation
Heuristic Baseline	0.278 ± 0.039
Heuristic Baseline + Grid Search	0.293 ± 0.038
Gradient Boosting	0.326 ± 0.040
Gradient Boosting with TE features	0.335 ± 0.040

Table 1: **Results of experiments with `Response Selector` optimisation.** Correlation of models predictions and ground truth labels. Results were obtained by averaging across 500 stratified splits on train/test sets.

generated by the CNN network that consists of two stacked convolutional layers followed by a max-pooling layer. The contextual information of words is then included by utilizing a Bi-LSTM network. Finally, a Conditional Random Field layer is used to capture dependencies between output tags.

The model was trained on the CoNLL2003 dataset [13]. This dataset consists of four types of entities, including person names, names of locations, names of organizations, and miscellaneous entities that don't belong to these three groups. The socialbot gets all texts from ASR in lower case, to match this we lowercased CoNLL2003 dataset. The model achieved F1 = 92.27 on the CoNLL2003 test set which is competitive with F1 = 92.40 performance demonstrated by the transformer-based BERT_base from [10] but requires less computing.

NER model was adapted for the sentence segmentation task by reformulating it as a sequence labeling task. It was trained on two conversational datasets generated from Cornell Movie-Dialog [9] and DailyDialog [38]. Since we focus on building a sentence segmentation module that is responsible for extracting two types of sentences (1) statement sentences and (2) questions for downstream modules, the texts with more than three sentences were removed, and three types of tags were used including (1) `B-S` to label the first word of a statement sentence, `B-Q` to label the first word of a question, `O` to label the other words. The model demonstrated F1 = 89.99 on Cornell Movie-Dialog and F1 = 95.88 on DailyDialog. Sentence segmentation model is now available as a part of DeepPavlov library[6].

5 Discussion

While socialbots are perhaps one of the oldest forms of the conversational AI-driven agents, and have a rich history of research in the academia, building a comprehensive and efficient socialbot for a broad audience of customers is a serious stress test for academic projects. At the very same time, the development of the industrial socialbots and AI assistants is rather limited by the strict requirements to the predictability of the product's functionality, thus doing experiments to create a new technology is a challenge. Constant reorganizations, as well as changes in the priorities typical for the fast-paced startups and even corporate environment, also make a smooth combination of research projects with the actual production systems problematic. Fortunately, the Alexa Prize Socialbot Grand Challenge[7] grants academic teams an opportunity to work with the real end-users while still being able to experiment with the new ideas and apply latest research breakthroughs. Alexa Prize settings allowed us to build our entire development process around user ratings, which gave us lots of insights into what kind of problems our users faced during their ongoing conversations with the DREAM socialbot.

During the competition, we used a large selection of the publicly available datasets to train models for `Annotators` and `Skills`. For `Annotators`, these were mainly the datasets for NER, sentiment and toxicity. Extensive use of the conversational datasets with the high-quality dialogues such as Topical Chat [18] and Daily Dialogs [38] strongly contributed to the development of the retrieval skills. We use them as a source of good responses as well as for training ranking models. Reddit was another main source of the "conversational" data, but its "real-world" nature required careful preprocessing before use.

During our experiments with trainable models, we have learned the following three key things.

[6] `http://docs.deeppavlov.ai/en/master/features/models/ner.html#ner-based-model-for-sentence-boundary-detection-task`

[7] `https://developer.amazon.com/alexaprize/challenges/`

1. Two commonsense conversational skills `Activity Discussion Skill` and `Personal Event Discussion Skill` that combine commonsense knowledge graphs completion models and template-based approach demonstrate higher explicit commonsense compared to open-domain rule-based skills. Regarding implicit commonsense they are similar to retrieval skills.
2. `Dialog Termination` turned out to be a strong feature for the response selector. In addition to it, *isResponseInteresting* feature provided by `Cobot Evaluator`, as well as skill confidence became strong contributors to quality gains for the response selector.
3. All generative models failed. While models such as GPT [29], GPT-2 [30], and Meena [1] showed encouraging progress of the generative models, the real-world application of the generative models in the interaction with Alexa Prize users showed their limited usability for now.

There is a number of insights about good conversational strategies that became evident from the user's ratings and their dialogues with the socialbot.

Facts are not engaging. Wrapping them into a conversational analog of syntax sugar with the small talk components smoothed fact mentioning.

Sharing own opinion can be powerful. However, the use of COMET Atomic & COMET ConceptNet and other knowledge graphs has to be done with caution, as high variability of language makes NLG challenging, leading to the low quality of the socialbot's responses.

It is a valuable strategy to **give people a venue to talk** about their opinions on different subjects, as well as about themselves. But it is not easy for the system to play the role of a good listener.

The user's behavior is highly variable, both across different users and within the dialogues. Socialbot must be able to identify the most appropriate strategy (active vs. listener, different age cohorts) and adapt it to the current dialogue context.

Switching between topics within the same socialbot's utterance smooths the conversation, making transitions between topics less annoying and more natural.

6 Conclusions

In spite of the ubiquity of simple chatbots, the development of engaging conversational agents remains to be a big research and engineering challenge. To succeed in this challenge, a number of problems should be addressed by both academy and industry.

The current state-of-the-art generative models like [29], [30], and [1] while being quite promising in the research settings and public demonstrations, don't work well enough in the real world. More research needed to make generated responses coherent with long dialogue contexts. Our experiments with generative models led to the same conclusions (see Appendix A.3.5).

The availability of the large scale data with the end-user ratings is crucial for the research progress in the conversational AI area. How to generate such data or automatically evaluate dialogue systems are pressing open questions. One promising solution here is academic conversational AI challenges [3, 11] which attract volunteers to chat with research systems to generate public conversational data [23, 24] and evaluate dialogue systems.

Building a socialbot capable of online adaption to the user requires a complex combination of the real-time user behavior analysis, use of "conversational sugar" for making socialbot's utterances more humanized, and efficient use of the commonsense-based knowledge graphs.

A multi-skill dialogue system should orchestrate heterogeneous conversational skills into a coherent but fluid dialogue flow. This is still mainly an unexplored research field compared to other NLP areas.

Fast progress in the Conversational AI field depends not only on bright ideas but also on the engineering tools for rapid prototyping and scalable deployment of the conversational agents. Here, we are looking forward and contributing towards the progress of the open-source libraries and frameworks like DeepPavlov and DeepPavlov Agent. We plan to release an open-source version of the DREAM socialbot and promote DeepPavlov ecosystem as a platform to build and exchange conversational skills and models.

Acknowledgements

DREAM team is deeply grateful to the Alexa Prize organizers for their feedback and advice during the competition. DREAM team also thanks all members of Neural Networks and Deep Learning Lab for their support and making participation in the competition highly productive.

References

[1] Daniel De Freitas 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. Towards a human-like open-domain chatbot. *ArXiv*, abs/2001.09977, 2020.

[2] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy, July 2019. Association for Computational Linguistics.

[3] Mikhail Burtsev, Varvara Logacheva, Valentin Malykh, Iulian Vlad Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W Black, Alexander Rudnicky, and Yoshua Bengio. The first conversational intelligence challenge. In *The NIPS'17 Competition: Building Intelligent Systems*, pages 25–46. Springer, Cham, 2018.

[4] Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, et al. Deeppavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, System Demonstrations*, pages 122–127, 2018.

[5] Daniel Matthew Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *ArXiv*, abs/1803.11175, 2018.

[6] Chun-Yen Chen, Dian Yu, Weiming Wen, Yi Mang Yang, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, et al. Gunrock: Building a human-like social bot by leveraging large scale real user data. *2nd Proceedings of Alexa Prize (Alexa Prize 2018)*, 2018.

[7] Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. Quac : Question answering in context. *CoRR*, abs/1808.07036, 2018.

[8] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087, 2015.

[9] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*, 2011.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[11] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer, Cham, 2020.

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

[13] E. F. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of Conference on Computational Natural Language Learning*, pages 142–147, 2003.

[14] Kurt Shuster Angela Fan Michael Auli Jason Weston Emily Dinan, Stephen Roller. Wizard of wikipedia: Knowledge-powered conversation agents. *Proceedings of ICLR*, 2018.

[15] Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In *Proc. Interspeech 2019*, pages 1891–1895, 2019.

[16] Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Ivan Vulić, et al. Convert: Efficient and accurate conversational representations from transformers. *arXiv preprint arXiv:1911.03688*, 2019.

[17] Jeffrey Pennington, Richard Socher, Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 1532–1543, 2014.

[18] Qinlang Chen Anna Gottardi Sanjeev Kwatra Anu Venkatesh Raefer Gabriel Dilek Hakkani-Tur Karthik Gopalakrishnan, Behnam Hedayatnia. Topical-chat: Towards knowledge-grounded open-domain conversations. *Proceedings of Interspeech*, 2019.

[19] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc., 2017.

[20] Chandra Khatri, Rahul Goel, Behnam Hedayatnia, Angeliki Metanillou, Anushree Venkatesh, Raefer Gabriel, and Arindam Mandal. Contextual topic modeling for dialog systems. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 892–899. IEEE, 2018.

[21] Chandra Khatri, Behnam Hedayatnia, Rahul Goel, Anushree Venkatesh, Raefer Gabriel, and Arindam Mandal. Detecting offensive content in open-domain conversations using two stage semi-supervision. *ArXiv*, abs/1811.12900, 2018.

[22] Chandra Khatri, Behnam Hedayatnia, Anu Venkatesh, Jeff Nunn, Yi Pan, Qihan Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, Ming Cheng, Qinglang Chen, Lauren Stubel, Karthik Gopalakrishnan, Kate Bland, Raefer Gabriel, Arindam Mandal, Dilek Z. Hakkani-Tür, Gene Hwang, Nate Michel, Eric King, and Rohit Prasad. Advancing the state of the art in open domain dialog systems through the alexa prize. *ArXiv*, abs/1812.10757, 2018.

[23] Varvara Logacheva, Mikhail Burtsev, Valentin Malykh, Vadim Poluliakh, Alexander Rudnicky, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W Black, and Yoshua Bengio. A dataset of topic-oriented human-to-chatbot dialogues, 2018.

[24] Varvara Logacheva, Valentin Malykh, Aleksey Litinsky, and Mikhail Burtsev. Convai2 dataset of non-goal-oriented human-to-bot dialogues. In *The NeurIPS'18 Competition*, pages 277–294. Springer, Cham, 2020.

[25] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1101–1104, 2019.

[26] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

[27] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, 2018.

[28] Jan Pichl, Petr Marek, Jakub Konrád, Martin Matulík, and Jan Šedivỳ. Alquist 2.0: Alexa prize socialbot based on sub-dialogue models. *2nd Proceedings of Alexa Prize (Alexa Prize 2018)*, 2018.

[29] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:8, 2019.

[31] Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, Eric King, Kate Bland, Amanda Wartick, Yi Pan, Han Song, Sk Jayadevan, Gene Hwang, and Art Pettigrue. Conversational ai: The science behind the alexa prize, 2018.

[32] Siva Reddy, Danqi Chen, and Christopher D. Manning. Coqa: A conversational question answering challenge. *CoRR*, abs/1808.07042, 2018.

[33] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[34] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[35] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[36] Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy, July 2019. Association for Computational Linguistics.

[37] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*, 2019.

[38] Xiaoyu Shen Wenjie Li Ziqiang Cao Yanran Li, Hui Su and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of The 8th International Joint Conference on Natural Language Processing (IJCNLP 2017)*, 2017.

[39] Dawei Song Peng Guo Junwei Zhang Peng Zhang Yazhou Zhang, Lingling Song. Scenariosa: A large scale conversational database for interactive sentiment analysis. *arXiv preprint arXiv:1907.05562*, 2019.

[40] Sanghyun Yi, Rahul Goel, Chandra Khatri, Alessandra Cervone, Tagyoung Chung, Behnam Hedayatnia, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tur. Towards coherent and engaging spoken dialog response generation using automatic conversation evaluators. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 65–75, Tokyo, Japan, October–November 2019. Association for Computational Linguistics.

[41] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics.

A DREAM Socialbot Components

A.1 Annotators

A.1.1 User Input Annotators

All annotators except of `ASR Processor` accept raw ASR texts composed by ASR hypotheses with the highest probabilities.

Sentence Segmentation allows us to handle long and complex user's utterances by punctuation recovery and splitting them into sentences. This model takes user's utterance as an input and outputs the list of the punctuated sentences.

Named Entity Recognition (NER) extracts person names, names of locations, organizations from the uncased text.

Sentence Rewriting rewrites the user's utterances by replacing pronouns with specific names that provide more useful information to downstream components.

Intent Catcher classifies user utterances into a number of predefined intents, such as *repeat*, *exit*, *what_is_your_name*, *what_can_you_do*, *yes*, *no*, *lets_chat_about*, *donot_understand* and etc. It uses a set of regular expressions and classification model. The total number of intents is 21. The classifier is based on the latest[8] version of Universal Sentence Encoder [5].

Blacklist Word Annotator detects words and phrases from several predefined blacklists: inappropriate, profanity, restricted topics. If user utterance contains phrases from restricted topics list, we turn on "safe mode" in the Skill Selector.

Automatic Speech Recognition Processor calculates overall ASR confidence for a given utterance and grades it as either a very low, low, medium, or high. This output is then used by the Misheard ASR skill (see A.3.3).

Toxic Classifier identifies whether an utterance contains insults, threats, obscene words, identity hate, sexual explicit talk, or other toxicity manifestations. The classification head on top of DeepPavlov English Conversational BERT-model[9] was trained on Kaggle Toxic Comment Classification Challenge[10] dataset.

Sentiment Classifier indicates if the utterance is positive, negative, or neutral. A classifier on top of DeepPavlov conversational BERT [9] was trained on Stanford Sentiment Treebank dataset [33] with five classes: very positive, positive, neutral, negative, and very negative. During inference, very positive (negative) labels are assigned to positive (negative). The model is available in DeepPavlov[11].

Emotion Classifier, is a BERT-based classifier trained on the mix of two datasets. The first one was the dataset with the examples of 6 emotions: anger, fear, joy, love, sadness, and surprise. We originally found these datasets on the Kaggle page of Eray Yildiz[12] but it is already unavailable a the time of writing. To make the dataset more balanced, we augmented it with the neutral examples from ScenarioSA dataset [39]. The final dataset for training is presented in DeepPavlov[13]. The train set contained more than 390k samples and the test set included 50k samples.

CoBot Annotators are built as API services on top of the Amazon Conversational Bot Toolkit (CoBot) [22]. `Topic Classifier`, `Dialog Act Classifier` [20] and `Offensiveness Classifier` [21] are one-label multi-class models which return topic, dialogue act, toxicity annotation, and blacklist indicator. We annotate user utterance sentence-wise to provide results in the format more similar to multi-label classification results and indicate if user expressed different intents or covered multiple topics in separate sentences. We also use the CoBot-provided code for noun phrases extraction, which returns filtered noun phrases from the user's response.

[8] https://tfhub.dev/google/universal-sentence-encoder/4

[9] http://docs.deeppavlov.ai/en/master/features/pretrained_vectors.html#bert

[10] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/overview

[11] https://github.com/deepmipt/DeepPavlov/blob/0.9.0/deeppavlov/configs/classifiers/sentiment_sst_conv_bert.json

[12] https://www.kaggle.com/eray1yildiz

[13] http://files.deeppavlov.ai/datasets/EmotionDataset.rar

		Predicted						
-	-	Anger	Fear	Joy	Love	Sadness	Surprise	Neutral
Ground Truth	Anger	5933	49	38	2	22	291	1
	Fear	263	4624	18	0	12	41	419
	Joy	17	5	14697	1138	4	27	112
	Love	1	1	14	3867	0	4	1
	Sadness	6	3	2	1	3109	0	0
	Surprise	48	229	36	7	9	13275	16
	Neutral	1	2	44	0	0	2	1609

Table 2: **Confusion matrix for the `Emotion Classifier`.** Dataset is not balanced. *Joy* is often misclassified with *Love*, which is not critical in our setup. Precision is more important than recall for the application of `Emotion Classifier`.

A.1.2 Candidate and Response Annotators

Response Candidate Annotators include `Toxic Classifier` and `Blacklist Words Detector` described in A.1.1 as well as `CoBot Conversation Evaluator` and original `Dialog Termination`.

Dialog Termination annotator predicts user intent to finish dialogue (i.e., the user said "Alexa, stop"). The model on top of DeepPavlov conversational BERT [9] was trained on conversational data generated during the interaction of the socialbot with Alexa users.

CoBot Conversation Evaluator is trained on the Alexa Prize data from previous competitions and predicts whether candidate response is interesting, comprehensible, on-topic, engaging, and erroneous [40]. CoBot Conversation Evaluator was provided to participants as an existing remote service.

As soon as the final response has been selected by `Response Selector`, we further process it with `Sentence Segmentation`, `NER`, and `Sentence Rewriting` `Response Annotators`. The final response annotations allow us to work with the outputs from the heterogeneous skills such as template-based ones with punctuation, retrieval, or generative skills in the same way.

A.2 Skill Selector

Skill Selector is rule-based. It reads dialogue context with annotations from the `Dialogue State` and selects the skills to generate candidate responses. If intents that require a specific response are detected, then only `Intent Responder` skill is requested for the response. If opinion request on sensitive topics or any toxicity in user's utterance are detected, then only *"safe mode"* skills including `CoBotQA` and several template-based skills are activated. In all other cases, the final selection of the skills to run is based on extracted topics, dialogue acts, length of the dialogue, as well information if the skill was active on the previous turn. `Dummy skill` is always enabled for a backup response.

A.3 Conversational Skills

A.3.1 Linking Skills

Appropriate transitions from one skill to another create smooth user experience. Skills can add templated triggers to enable other skills on the next dialogue turn. At the point when active skill decides to switch it calls `link to` function with "target" skill as the parameter. `link to` adds "target" skill invocation phrase at the end of the bot response. Thus, `Skill Selector` runs "target" skill on the next turn, and generated candidate response has increased score at response selection.

There is also a heuristic that randomly adds a link to template-based skills (Section A.3.3) to the output of retrieval skills (Section A.3.6) and some 1-step conversation skills like `CobotQA`, if the output is a statement (not a question). This heuristic helps to make a better user experience, because linked conversational skills provide well-designed scripted multi-step conversations.

A.3.2 AIML Skills

Artificial Intelligence Markup Language (AIML) is an XML dialect for creating dialogue agents. The main advantage of AIML is that it is well-documented, widely used, and easy to start language to implement chatbots. In DREAM system we use Program Y[14] framework.

AIML DREAM Chit-Chat is based on Template-y bot[15]. We curated and updated bot's rules to add a greeting scenario, several general conversation templates, as well as jokes. **AIML General Chit-Chat** supports templates for common phrases. **AIML Dangerous Topics Conversation Skill** addresses potentially dangerous situations arising from abusive, insulting, and inappropriate user utterances. **Alice** is an open-source AIML chatbot[16]. It has a comprehensive set of grammars and was especially helpful at the beginning of the competition.

A.3.3 Template-based skills

Intent Responder provides template-based replies for some of the intents detected by `Intent Catcher` annotator.

Eliza[17] is one of the Python implementations of the well-known natural language processing program inspired by the classical AI paper [35].

Dummy Skill is a fallback skill with multiple non-toxic candidate responses. It retrieves responses related to the subject of the conversation from more than 6500 facts and tips from the different Subreddits and 1800 questions from the Topical Chat dataset [15]. It also returns a `link-to` question, which steers the conversation to one of the script-based skills. This question is sampled by taking into account previously asked linking questions, and it can also be attached to responses of some of the skills by the `Response Selector`.

Response candidates provided by the `Dummy Skill` have significantly lower confidences compared to other skills. So, if the system cannot directly answer to the user's utterance, the skill mimics "recollection" of something relevant (noun- or topic-based questions and facts) to the context, or leads the conversation to the topic which can be supported by one of the script-based skills.

Dummy Skill Dialog returns the next turn from the Topical Chat dataset if the response of the user to the `Dummy Skill` is similar to the corresponding response in the source data.

Personal Info Skill queries and stores user's name, birthplace, and location. The user profile can be further used by other skills in order to start the socialbot's response with the user name or to offer a weather forecast in the user's location.

Emotion Skill returns template responses to emotions detected by the `Emotion Classification` annotator. Upon successful extraction of user's emotional state, this skill tries to react accordingly. It can ask the user to calm down, tell a joke, cheer up, or provide a bit of advice when negative emotions were detected. The skill has a few scripted dialogue parts, and it can go beyond a one-phrase answer.

Movie Skill takes care of the conversations related to movies. It provides responses to the frequently asked movie questions like "What is your [less-]favorite [movie/actress/movie genre]?". In addition to that, this skill can detect user's opinion and express its own opinion on a variety of subjects, including movies, movie genres, and actors. Expressed attitude to movies is rating-based, attitude to genres is manually scripted, while attitude to actors depends on the average rating of movies they played in.

`Movie Skill` detects user's responses for questions about movies including `link to` questions and any other user's statements labeled as related to the movie topic. If it finds a movie title with more than 10k votes on IMDb, then scripted dialogue focused on this title is started. Otherwise, this skill clarifies whether the extracted title is correct. The script includes opinion expression and request, the question about the movie genre or cast, facts about awards or tagline of the movie, and at the end, just some interesting facts. The conversation flow can be switched with the questions related to the movie's topic. If the user directly asks to change topic, the skill calls `link to` method to add dialogue steering question for activation of `Book Skill` or `Short Story Skill`.

[14] `https://github.com/keiffster/program-y`

[15] `https://github.com/keiffster/program-y/wiki/Available-Bots`

[16] `https://github.com/sld/convai-bot-1337/tree/master/ALICEChatAPI`

[17] `https://github.com/wadetb/eliza`

Book skill detects book titles and authors mentioned in the user's utterance with the help of Amazon Evi[18] and discuss them. The skill provides facts about extracted book titles, authors, and recommends books by leveraging information from the GoodReads database[19].

Activity Discussion Skill provides a multi-turn dialogue around human activities. The skill uses COMeT Atomic [2] model to generate common sense descriptions and questions on several aspects (e.g. what person wants/feels during an action) of human activities in natural language. More details about implementation of this skill are presented in Subsection 4.1.

Personal Event Discussion Skill uses COMeT ConceptNet [2] model to express an opinion, to ask a question or give a comment about user's actions mentioned in the dialogue. The generated opinion depends on the sentiment of the predicted assertions of common sense. More details about `Personal Event Discussion Skill` can be found in Subsection 4.1.

Small-talk Skill asks questions using the hand-written scripts for 25 topics, including but not limited to love, sports, work, pets, etc. The script is started if the user directly asks to talk about one of these topics or suggest topic if the user expresses no preference. All scripts consist of 4-10 questions with the simple branching based on the `yes/no` user's replies.

Event-oriented Skills support FAQ , facts, and scripts for Christmas and New Year, Super Bowl, Oscar, and Valentine's Day.

Misheard Automatic Speech Recognition Skill uses the `ASR Processor` (Section A.1) annotations to give feedback to the user when ASR confidence is too low.

A.3.4 Template-based Skills with External Services

CoBotQA answers factoid questions as well as provide facts about extracted noun phrases and named entities for "fact about" and to "fun fact about" requests. It is implemented on top of the remote Q&A CoBot service, which works with plain text. The output from Q&A CoBot service is limited to 1-2 sentences and augmented with small opinion-like phrases. In case of opinion request on restricted topics, `CoBotQA` refuses to express an opinion and provides a fact about mentioned topic.

Weather Skill uses the OpenWeatherMap [20] service to get the forecast for the user's location. Weather intent is detected by the `Intent Catcher` annotator.

News Skill presents the top-rated latest news about entities or topics using the News API[21]. The skill is activated in two cases: (1) a user requests news, or (2) breaking news suggestion generated with the `link to` method is accepted by the user. A three-step scenario starts by presenting the headline of the latest news or the news on a particular topic. If the user wants to get more details, then the skill reads out the description of the news and follows up by asking user's opinion. Finally, `News Skill` gives a choice between two randomly chosen popular news topics (e.g., sports, politics, etc.) to further continue the conversation. At this step, the user can pick up a suggested topic or request another one. If the `NER` annotator detects some entity at this step, the skill restarts. When user wants to wrap up the discussion about the news, the skill uses `link to` option to switch the topic to another one supported by other skills.

Game Skill provides user with a conversation about computer games. It can talk about the charts of the best games for the past year, past month, and last week. It can also give details about a specific game, as well as perform a search for it. This skill uses game-related content like games databases, their ratings, etc., retrieved from the RAWG API [22].

Coronavirus Skill was created in response to the coronavirus pandemics. It retrieves data about the number of coronavirus cases and deaths in different locations from the sources of the John Hopkins University Center for System Science and Engineering[23]. Then the skill uses the set of hand-coded phrases about facts and recommendations from the CDC (Centers for Disease Control and Prevention). `Coronavirus Skill` takes into account annotations from the `Emotion Classifier` annotator.

[18] `https://www.evi.com/`

[19] `https://www.goodreads.com/`

[20] `https://openweathermap.org/`

[21] `https://newsapi.org/`

[22] `https://rawg.io/`

[23] `https://github.com/CSSEGISandData/COVID-19`

Short-Story Skill tells user short stories from 3 categories: (1) bedtime stories, such as fables and moral stories, (2) horror stories, and (3) funny ones. It is triggered on the `tell_me_a_story` intent by `Intent Catcher` or can be invoked on its own if the context is appropriate.

A.3.5 Generative Skills

TransferTransfo is a sequence to sequence model with the conditional text generation based on a Hugging Face [37] repository[24]. This model was developed for the Persona Chat task from the ConvAI2 competition. It is trained to generate chit-chat based on the persona description. When the model was added to the socialbot, its generated responses faced issues such as self-repetition and contradictions with previous utterances. We used beam search to generate a variety of answer candidates. To exclude repetitions, we have added the rule to filter out hypotheses that exceed the number of common words with the latest utterances within the dialogue context. To choose hypotheses that do not contradict the context, we used the model trained on the Dialog NLI dataset [36]. We have also tried to use a summary of a news article instead of the person description, but the responses of the model often contained information only weakly related to the summary of a news article, and the consistency of the responses deteriorated as the number of conversation turns grew. We were unable to reach a sufficient quality level of the model to eliminate contradictory answers.

A.3.6 Retrieval Skills

ConveRT Reddit Retrieval Skill uses a ConveRT [16] encoder to build efficient representations for sentences. ConveRT is a smaller and faster transformer compared to encoders based on BERT but with the quality of the similar representations. The model retrieves candidate responses by ranking response-context pairs by cosine similarity of the corresponding embeddings. Context is created by concatenation of utterances in a dialogue history.

The model was trained on the large dataset from Reddit, so it is specifically optimized for conversational experiences. The dataset for training consisted of comment and response-comment pairs. About 2 million comments were collected from Reddit and filtered by CoBot `Conversation Evaluation` service and `Toxic Classifier`. As a result, only 80K of the comments remained in the final retrieval dataset.

NER-Skill on Reddit takes an entity recognized by Amazon Evi[18] in the user input and makes a lookup for it in the dataset of Reddit posts. After that, the response is formulated as if the socialbot recently learned something about that entity from Reddit. The dialogue is then continued by talking about a number of linked entities (with the number of links constrained by the Amazon Evi Information Graph).

TF-IDF-retrieval retrieves a response from the history of the highly-rated dialogues. We built retrieval set from the last month dialogues rated with five stars. This set consists of pairs where the user phrase corresponds to the bot phrase. Specifically, retrieval model uses TF-IDF vectorizer trained on the dataset combined from Topicalchat [18], PersonaChat[41] and Wizards-of-Wikipedia [14]. For each user utterance, the model looks for the closest (by cosine distance) phrase of user or bot. Then the model returns the next phrase with the confidence equal to the cosine distance. this confidence is capped by some constant value.

TF-IDF-retrieval Skills on Topical Chat is the set of retrieval skills for books, entertainment, fashion, movies, music, politics, science & technology, sport, animals. The sets of candidate responses for the skills were collected from the Topical Chat [18] dataset.

Topical Chat ConveRT Retrieval Skill uses the same model as the `ConveRT Reddit Retrieval Skill` but retrieves from the Topical Chat dataset. Depending on the current topic, it takes the corresponding dataset and finds the response with the highest score.

A.4 Response Selector

`Response Selector` is a DREAM agent component that makes the final decision about the content of the response to be surfaced to the user. `Response Selector` reads from the `Dialogue State` candidate responses generated by the active conversational skills and annotated by the `Response`

[24] `https://github.com/huggingface/transfer-learning-conv-ai`

`Annotators`. `Response Selector` is not restricted to select the final response only from the response candidates but can also generate a final response as a combination of available candidate responses.

Current implementation of the `Response Selector` is heuristics driven, but the upcoming version will include a trainable ranking model (see Section 4.2 for details). `Response Selector` makes a choice of the final response through the several steps. It starts by filtering response candidates by the `Blacklist Words` annotations, predictions of the `Toxic Classifier` and the `Dialog Termination` Annotators. Then the confidences for the repeating candidates are penalized. On the next step, every candidate is scored with a weighted sum of its confidence and a score generated by `CoBot Conversation Evaluator`. Finally, the response with the highest score is selected. Then it can be concatenated with the user name if it is already known, with an engaging phrase, or `link to` questions. The resulting utterance goes to the postprocessing by `Response Annotators` and then presented to the user.

B DREAM Socialbot Infrastructure and Deployment

We used Docker for AWS Cloud Formation[25] for the initial setup of the CPU cluster of Docker Swarm. GPU machines required a little bit of handwork to be manually added to the Docker Swarm cluster. The DREAM socialbot setup requires 6xCPU-only m5.xlarge and 2xGPU g4dn.xlarge instances[26] to reliably support a load of at least 5 requests per second. All configurations are described in the docker-compose files, so any developer could run the entire socialbot locally[27]. We used a separate EC2 instance with MongoDB for storing DREAM agent `Dialogue State` history. A diagram of DREAM socialbot infrastructure is presented in Figure 6.

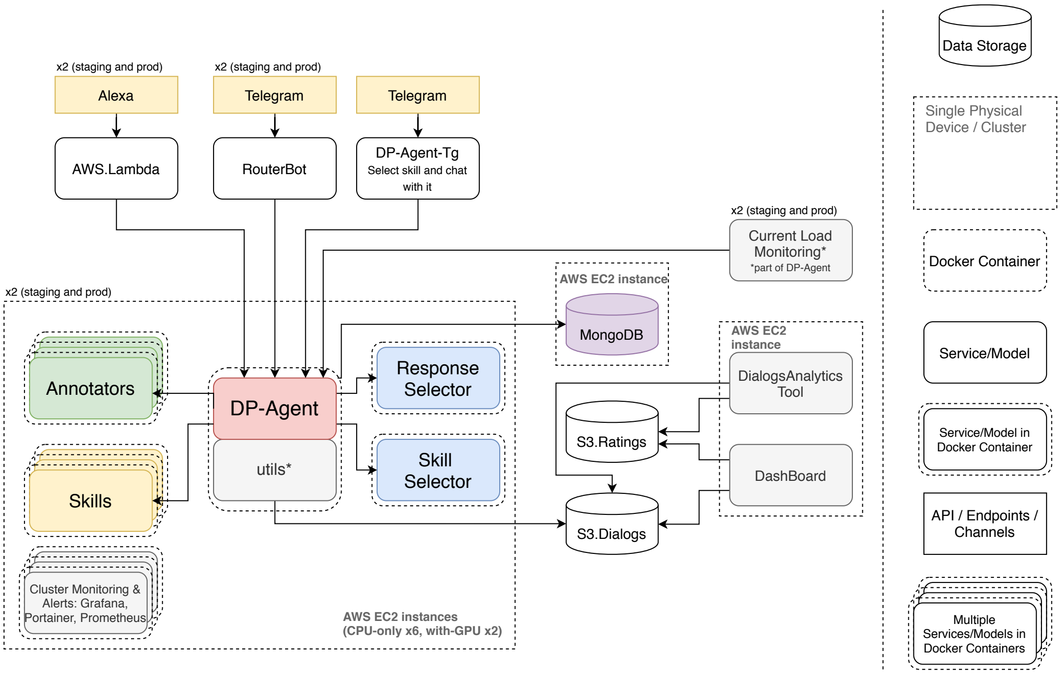


Figure 6: **DREAM socialbot infrastructure.** The core of the DREAM socialbot is implemented with DeepPavlov Agent (DP-Agent) framework. It orchestrates services for `Skills`, `Annotators`, `Skill Selector` and `Response Selector`, and is located on AWS EC2 instances with Docker Swarm. `Dialogue State` history is stored on a separate instance with MongoDB. We have AWS Lambda that performs HTTP requests to the DREAM-agent by sending ASR tokens. Testing infrastructure consists of Telegram bots for interacting with the `dev` version of the socialbot or with selected conversational skill only. Dialogue analytics tool and dashboard are located in a separate EC2 instance. Also, we have cluster and application monitoring with configured alerts to email and Slack.

For the cluster monitoring, we used Swarmprom[28]. It is a starter kit for Docker Swarm monitoring with Prometheus, Grafana, cAdvisor, Node Exporter, Alert Manager, and Unsee for the cluster monitoring. Swarmprom allows us to monitor CPU and memory usage out of the box with alerts in Slack. To manage all Docker containers in one web-interface, we used Portainer[29].

We had three separate infrastructure environments for staging, production A, and production B. In staging, we deployed our latest changes and tested them manually by ourselves. Production A and production B environments were used for A/B tests of stable releases. We usually deployed one release per day. If critical bugs were found after initial deployment, one or two more releases followed. Selecting between production A and B for a user was defined in the AWS Lambda side. Usually, we assigned users to different groups in a 50/50 ratio.

[25] `https://docs.docker.com/v18.09/docker-for-aws/`

[26] `https://www.ec2instances.info`

[27] hardware requirements: a machine with 32GB RAM, 16GB GPU (g4dn.2xlarge instance)

[28] `https://github.com/stefanprodan/swarmprom`

[29] `https://www.portainer.io`

Since each skill is a separate application and container inside our repository, it allowed us to work without getting worried that it will affect any other skills. Also, we have configured the Continuous Integration pipeline with Jenkins[30] that runs code style, unit, and integration tests.

Application-level (dp-agent, annotators, skills, services) logs go to the CloudWatch[31]. Errors in applications are logged with Sentry[32]. It provides an application monitoring platform that helps to identify issues in real-time, especially it sends a notification to email and Slack when any skill, annotator, or another part of the DREAM socialbot raises an exception.

Also, we serve two Telegram[33] bots that allow us to test `dev` version of socialbot without Echo devices and Amazon Developer Console. The first one is a text interface to the whole socialbot, while another one allows us to chat with a chosen skill separately.

[30] https://www.jenkins.io/

[31] https://aws.amazon.com/cloudwatch/

[32] https://sentry.io

[33] https://telegram.org

C Analytical Tools

At the end of each conversation, the Alexa Prize platform collects a rating from the user by asking, "on a scale of 1 to 5, how do you feel about speaking with this socialbot again?" [31]. We designed a feature-rich analytical system to monitor the status of the socialbot from different perspectives, ranging from the number of dialogues, average dialogue rating, and skill ratings to A/B tests, dialogue ending reasons, last skill in dialogue, as well as returning users. All this information is presented at the web-based dashboard with visualisations produced with `plotly`[34].

DREAM socialbot is a multi-skill conversational agent, and understanding the contribution of every skill to the overall dialogue rating is very important. However, as ratings are available only for the whole dialogue, direct measuring of individual skill performance is not possible. Instead, we estimated the contribution of each individual skill by the number of times it has been used within the dialogue, as well as how close the skill is to the end of the dialogue. With the assumption that contribution of the utterance to the dialogue rating w decays exponentially with the distance from the last turn each skill's rating for the given conversation is calculated as:

$$w_t = \begin{cases} Y_1, & t = 1 \\ \alpha \cdot Y_t + (1 - \alpha) \cdot w_{t-1}, & t \in [2, T] \end{cases} \tag{1}$$

here $Y_t = 1$ if t^{th} utterance belongs to the skill and $Y_t = 0$ otherwise; T is entire dialogue's length.

For skill s rating across all dialogues R_s can be calculated as follows:

$$R_s = \frac{\sum\limits_{j=1}^{m} (w_{sj} \cdot R_j)}{\sum\limits_{j=1}^{m} w_{sj}}, \tag{2}$$

here R_j is a rating of j-th dialogue, m is the total number of dialogues; w_{sj} is a weight of skill s in a dialogue j. An example of skill rating visualization for different releases is shown in Figure 7. The dashboard also has additional skills rating plots for short (7 or fewer turns) and long dialogues (more than 7 turns).

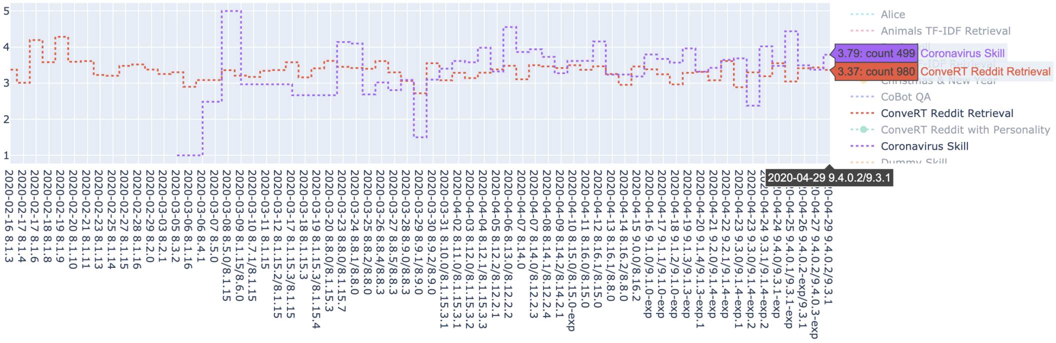


Figure 7: **Skills ratings by releases, EMA (0.5).** We monitor how ratings of different skills are changing during the development process. `Coronavirus Skill` ratings fluctuate while `ConveRT Reddit Retrieval` ratings are stable. `Coronavirus Skill` was continuously improving while rating of `ConveRT Reddit Retrieval` is stable in this period.

The visualization shown in Figure 7 allows us to enable or disable ratings of the individual skills, or see all of the ratings at the same time, therefore making it easy to analyze the dynamics of every skill individually or in comparison with each other.

A number of more detailed charts were introduced to track skill ratings over the last dialogues, average dialogue time, and the average number of utterances. For example, a chart for tracking skill ratings over the last dialogues has been designed to track changes that happened during the

[34] `https://plotly.com/`

day to enable prompt response of the team to users' feedback based on the ratings. Chart with the version-based rating distribution allowed us to identify the reason behind rating changes quickly, be that a growing number of high (5) or low (1) ratings. Dialog ending also has a few more charts used to analyze further the role of the skills in the final outcome of the dialogue, as well as to see who initiated dialogue finishing: a user, Alexa, or the socialbot itself.

Finally, while the unique specifics of the Alexa Prize competition seriously limited users from coming back to a given socialbot, a separate chart has been built to track the number of dialogues run with the returning users.

To track the performance of the constantly updated skills, as well as the experimental components of the DeepPavlov Agent platform, we performed A/B testing by running multiple controlled experiments. Usually, each day one new A/B test was run. The dashboard provides access to a separate page with the list of all A/B experiments up to date.

Each A/B test page represents both the high-level statistics of each version in the test and several key charts, allowing comparison of those versions with each other during and after the end of the test. These include median and mean ratings of the dialogues for the both versions' rated dialogues, median and mean number of utterances per dialogue, and the total count of dialogues. Distributions of rating for the versions were compared with the Mann-Whitney test.

The primary focus of the A/B test charts is detecting differences in skill performance between experiments. We have found that a frequency of each skill's calls in each version gives simple and useful guidance for further investigation, the example is shown in Figure 8.

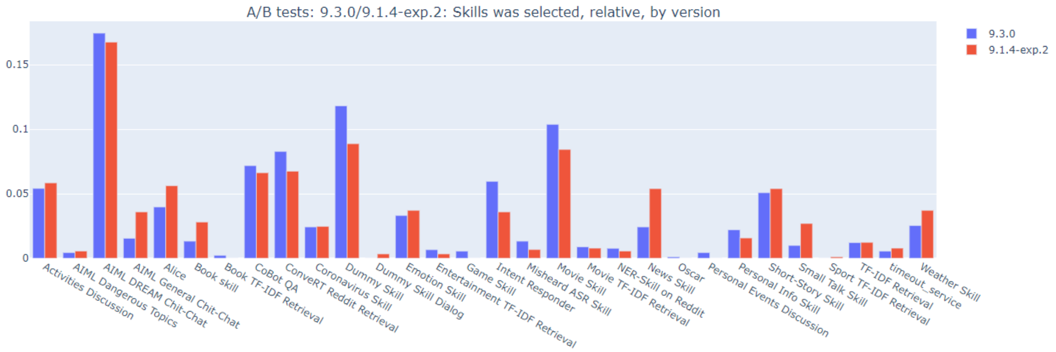


Figure 8: **A/B tests: Skills selected, relative, by version.** The number of `Movie Skill` responses significantly varies between A/B versions due to different movie databases.

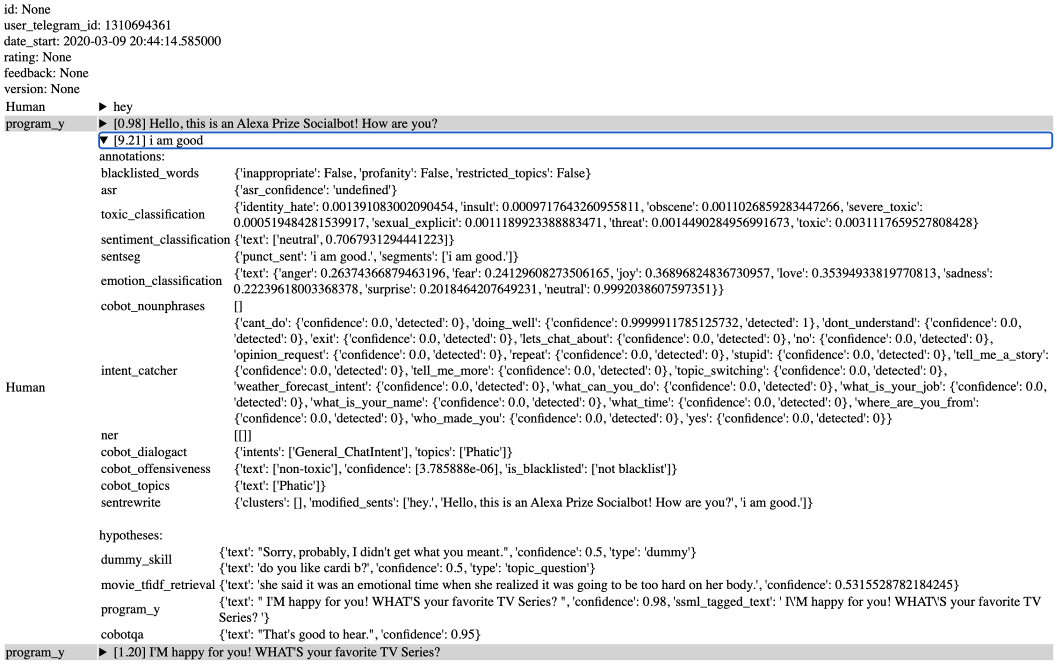
C.1 Conversation Profiling

While statistical data provides high-level insight into the overall performance of the socialbot itself as well as its individual versions running day by day in the A/B experiments, deeper understanding requires detailed profiling of the conversations. It is performed by looking into the actual data either from the conversational or utterance perspective. For this task, we developed a profiling tool with the web-based interface to the database with dialogues performed by DREAM Socialbot.

With the conversation profiling tool, developers can search for the conversations of interest across different measurement axis, including date periods, conversation length, user feedback, ratings, active skills, utterance texts, users, as well as versions of the system. The researcher can select a number of conversations and use an export method to save the selected conversations in the internal JSON format for further offline analysis.

A dialogue from the database can be opened in the conversation profiling page shown in the Figure 9. It has been designed to provide full information about the dialogue as seen by the system itself. Each human's utterance can have a number of annotations, and this page enables diving deep into the details of each annotator output for a given utterance. In addition to annotations, each human utterance has a set of corresponding response candidates provided by skills selected for this turn.

Bot responses are also provided with the debug information comprised with annotators information including results of the sentence segmentation annotator, an optional list of the named entities



```
id: None
user_telegram_id: 1310694361
date_start: 2020-03-09 20:44:14.585000
rating: None
feedback: None
version: None
Human       ▶ hey
program_y   ▶ [0.98] Hello, this is an Alexa Prize Socialbot! How are you?
            ▼ [9.21] i am good
            annotations:
             blacklisted_words        {'inappropriate': False, 'profanity': False, 'restricted_topics': False}
             asr                      {'asr_confidence': 'undefined'}
             toxic_classification     {'identity_hate': 0.001391083002090454, 'insult': 0.0009717643260955811, 'obscene': 0.0011026859283447266, 'severe_toxic':
                                      0.000519484281539917, 'sexual_explicit': 0.0011189923388883471, 'threat': 0.0014490284956991673, 'toxic': 0.003111765952780842 8}
             sentiment_classification {'text': ['neutral', 0.7067931294441223]}
             sentseg                  {'punct_sent': 'i am good.', 'segments': ['i am good.']}
             emotion_classification   {'text': {'anger': 0.26374366879463196, 'fear': 0.24129608273506165, 'joy': 0.36896824836730957, 'love': 0.35394933819770813, 'sadness':
                                      0.22239618003368378, 'surprise': 0.2018464207649231, 'neutral': 0.9992038607597351}}
             cobot_nounphrases        []
                                      {'cant_do': {'confidence': 0.0, 'detected': 0}, 'doing_well': {'confidence': 0.9999911785125732, 'detected': 1}, 'dont_understand': {'confidence': 0.0,
                                      'detected': 0}, 'exit': {'confidence': 0.0, 'detected': 0}, 'lets_chat_about': {'confidence': 0.0, 'detected': 0}, 'no': {'confidence': 0.0, 'detected': 0},
                                      'opinion_request': {'confidence': 0.0, 'detected': 0}, 'repeat': {'confidence': 0.0, 'detected': 0}, 'stupid': {'confidence': 0.0, 'detected': 0}, 'tell_me_a_story':
             intent_catcher           {'confidence': 0.0, 'detected': 0}, 'tell_me_more': {'confidence': 0.0, 'detected': 0}, 'topic_switching': {'confidence': 0.0, 'detected': 0},
Human                                 'weather_forecast_intent': {'confidence': 0.0, 'detected': 0}, 'what_can_you_do': {'confidence': 0.0, 'detected': 0}, 'what_is_your_job': {'confidence': 0.0,
                                      'detected': 0}, 'what_is_your_name': {'confidence': 0.0, 'detected': 0}, 'what_time': {'confidence': 0.0, 'detected': 0}, 'where_are_you_from':
                                      {'confidence': 0.0, 'detected': 0}, 'who_made_you': {'confidence': 0.0, 'detected': 0}, 'yes': {'confidence': 0.0, 'detected': 0}}
             ner                      [[]]
             cobot_dialogact          {'intents': ['General_ChatIntent'], 'topics': ['Phatic']}
             cobot_offensiveness      {'text': ['non-toxic'], 'confidence': [3.785888e-06], 'is_blacklisted': ['not blacklist']}
             cobot_topics             {'text': ['Phatic']}
             sentrewrite              {'clusters': [], 'modified_sents': ['hey.', 'Hello, this is an Alexa Prize Socialbot! How are you?', 'i am good.']}

            hypotheses:
             dummy_skill          {'text': "Sorry, probably, I didn't get what you meant.", 'confidence': 0.5, 'type': 'dummy'}
                                  {'text': 'do you like cardi b?', 'confidence': 0.5, 'type': 'topic_question'}
             movie_tfidf_retrieval {'text': 'she said it was an emotional time when she realized it was going to be too hard on her body.', 'confidence': 0.5315528782184245}
             program_y            {'text': " I'M happy for you! WHAT'S your favorite TV Series? ", 'confidence': 0.98, 'ssml_tagged_text': ' I\'M happy for you! WHAT\'S your favorite TV
                                  Series? '}
             cobotqa              {'text': "That's good to hear.", 'confidence': 0.95}
program_y   ▶ [1.20] I'M happy for you! WHAT'S your favorite TV Series?
```

Figure 9: **Conversation profiling page.** This conversation is provided as an example and it is not from a real customer.

extracted from the previous utterance, a history of the previous utterances normalized by the sentence rewrite annotator, and an optional list of the hypotheses returned by the currently running skill.