GCDF1: A Goal- and Context- Driven F-Score for Evaluating User Models

Alexandru Coca, Bo-Hsiang Tseng, Bill Byrne
Department of Engineering, University of Cambridge, United Kingdom
{ac2123, bht26, wjb31}@cam.ac.uk

Abstract

The evaluation of dialogue systems in interaction with simulated users has been proposed to improve turn-level, corpus-based metrics which can only evaluate test cases encountered in a corpus and cannot measure system's ability to sustain multi-turn interactions. Recently, little emphasis was put on automatically assessing the quality of the user model itself, so unless correlations with human studies are measured, the reliability of user model based evaluation is unknown. We propose GCDF1, a simple but effective measure of the quality of semantic-level conversations between a goal-driven user agent and a system agent. In contrast with previous approaches we measure the F-score at dialogue level and consider user and system behaviours to improve recall and precision estimation. We facilitate scores interpretation by providing a rich hierarchical structure with information about conversational patterns present in the test data and tools to efficiently query the conversations generated. We apply our framework to assess the performance and weaknesses of a `Convlab2` user model[1].

1 Introduction

Remarkable progress has been achieved in many dialogue systems research disciplines, from dialogue state tracking (DST) (Dai et al., 2021; Mehri et al., 2020) to policy- (Wang et al., 2020; Lubis et al., 2020) and end-to-end modelling (Peng et al., 2021; Yang et al., 2020). Progress is usually measured *component-wise* through task-specific metrics and improvements in the overall performance of the systems leveraging advances in component designs are seldom reported (Takanobu et al., 2020). Takanobu et al. (2020) empirically show that component-wise evaluation may not correlate well with the overall performance of the system. They recommend evaluating dialogue systems in an end-to-end, interactive, *multi-turn* setting to capture the effect of error propagation on system performance and approximate the field performance of a system more accurately.

Takanobu et al. (2020) perform extensive user model interactive evaluation for a wide range of dialogue system architectures implemented in the `Convlab` library (Lee et al., 2019). They find that while the simulated user interaction evaluation overestimates the true performance of the systems evaluated, a mild correlation with human performance assessment exists. In this context, this paper seeks to provide a simple and effective tool to measure the predictive power of a user model, arguing that it is important to understand how well current user models perform and how to enhance them to improve system-wise evaluation accuracy.

We propose a simple generalisation of the corpus-based, turn-level F1 score proposed by Schatzmann et al. (2005) as a measure of the similarity between the (semantic-level) simulated user response and the response provided by a real user given the same context. We believe this to be necessary since turn-level F1 favours models which are biased to a potentially restricted set of behaviours learned from a corpus whereas an optimal user model should exhibit a wider variety of behaviours. Similar to the `Convlab2` (Zhu et al., 2020) evaluation, our metric is *goal-driven*. It evaluates, at *dialogue level*, the ability of the user model to express all the constraints[2] (I-GCDF1) and *request* all the information (R-GDCDF1) prescribed by a goal when interacting with an arbitrary agent. In human-human conversation, repetition of constraints occurs due to co-reference, confirmation, emphasis and through other linguistic and conversational processes. Information requests may be specified at the same time with the search constraints and later repeated. Language understanding errors may see agents stuck in conversational loops where

[1] Code available at `https://bit.ly/3hVS55Q`.

[2] A constraint (e.g., `price=cheap`) is formed of a *slot* which constrains the search (price) and its *value* (cheap).

Proceedings of the First Workshop on Evaluations and Assessments of Neural Conversation Systems, pages 7–14
Nov 11, 2021. ©2021 Association for Computational Linguistics

the same question and answer are repeated ad nauseam. Failure to account for these repetitions may thus affect F1 scores. Consequently, GCDF1 scores are also *context-driven*: the dialogue span between repetitions of constraints or requests is analysed to determine whether they are erroneous, elicited by a system behaviour, or due to an intrinsic user behaviour. In addition, the mentioning of *not-in-goal* constraints warranted by the conversational context (e.g., mentions of *don't care* values or entities) are accounted for.

The GCDF1 evaluator outputs a rich hierarchical structure where the interactions evaluated are classified according to the results of the context-driven analysis. Additionally, dialogue level score information and other metadata are output. We also developed tools to analyse evaluator output and query the set of interactions to interpret model behaviour. Hence, we hope that our implementation will help developers improve their models.

In summary, our contributions are:

- a dialogue-level, goal- and context-driven metric for evaluating the semantic interaction between a user- and a system model
- a reference implementation for the metric along with a set of tools that help developers interpret the results and find ways of improving their models.

We support these claims by studying the user model employed by Takanobu et al. (2020) in their study of dialogue system performance.

2 Related work

Schatzmann et al. (2005) propose a variety of turn- and dialogue-level statistics to compare generated and real corpora. The histograms of these statistics are used as proxies of user model performance. This is practical in the setting they analysed, but in general the high-dimensional nature of the data that may be extracted makes such comparisons difficult. Later work (Keizer et al., 2010; Cuayáhuitl et al., 2005) employed the Kullback-Leibler (KL) divergence to compare the distributions over extracted statistics. As well as posing estimation problems and being more sensitive to the means of the distributions compared to their shape, as a scalar measure, the KL divergence does not provide any insight into the structure of the generated data or how to improve the model. A divergence measure approach is also proposed by Williams (2007), who ranks user models according to the Cramér-von Mises divergence between the system performance distributions measured in interaction with simulated user populations and real users. Callejas et al. (2012) suggest to overcome the lack of interpretability of these apporaches by using mutidimensional subspace clustering to graphically show the similarity between generated and real data, but their metric is susceptible to the choice of features and clustering algorithm.

Jung et al. (2009) propose to adapt the BLEU score (Papineni et al., 2002) to capture "dialogue level naturalness" by considering a "gram" to be a user or system action and show that this metric correlates well with human judgement. One drawback to applying this metric to compare the sequences generated by user models with references is the arbitrary ordering of action sequences. The similarity of the simulated dialogues to the real data is assessed by evaluating the perplexity of the user model instead. However, this metric may not be a good indicator of the ability of the user model to predict a realistic response in an unknown dialogue situation, so it does not measure models' task completion ability.

To measure the ability to appropriately respond in a given dialogue situation, the turn-level F1 score (Schatzmann et al., 2005) is used. Alternatively, data is generated through interaction of the user model to be assessed with a wide range of system models, a protocol known as cross-evaluation (Schatztnann et al., 2005). System-side metrics of task success computed for each system model are then averaged and used as proxies for the user model performance: a good user model is expected to perform well when interacting with a variety of dialogue systems and should attain a high score.

3 Metric description

The following sections present the I- and R-GCDF1 algorithms. Our implementation is based on the MultiWOZ 2.1 (Eric et al., 2020) corpus, where the behaviours mentioned in this section were detected.

3.1 Inform-GCDF1 algorithm

To robustly measure the precision and recall of the user actions, the algorithm first maps the value in each constraint to its canonical form. It then accounts for not-in-goal constraints and system/user behaviours when counting constraint repetitions. Finally, it checks if missing user constraints have been preempted by the system.

Value normalisation MultiWOZ does not provide canonical value annotations. These are taken to be the values that parametrise the entire set of user goals. Value paraphrases of all the 17 informable slots are extracted from the dialogue acts, curated and mapped to canonical forms. This yields a mapping containing over $6,989$ surface form variations for $2,079$ canonical values. Even a simple slot such as area, which has only 5 canonical values was mapped to 239 distinct values. Not accounting for these surface forms variations would decrease I-GCDF1 accuracy because correct user constraints in non-canonical would be counted as false. It would also not be possible to accurately detect if the system pre-empts user constraints if the system constraints are not in canonical form positives.

The normalisation procedure uses the slot name to retrieve all the value paraphrases. The Levenshtein distance between a candidate value and each paraphrase is computed, and a paraphrase is considered a match its distance is less than 0.1. The canonical form of the value is the canonical form of the closest matching paraphrase within the aforementioned tolerance, if it exists.

Not-in-goal constraints A dialogue system might offer multiple entities that satisfy the informed constraints, so the user would have to provide the name to select one. The user may also provide the name when informed that their search did not return results and offered an alternative. The system may also specify entity attributes which are not in the goal that the user may co-refer to in the next turn. Finally, since it does not know the user goal, the system may request the user to provide values for slots outside it. These patterns are detected by the evaluator and the false positive counts are adjusted accordingly.

Constraints repetitions Constraint repetitions occur due to user and system behaviours. For example, if a user search or booking fails, the user may repeat some already mentioned information when updating their criteria. The system may also ask some values to be repeated if uncertain about what was communicated. In addition, the user might repeat information when stating new constraints, while discussing a potential transaction, when requesting information or responding to information requests. It is also possible that information is repeated when multiple domains are discussed simultaneously in one turn and the system only handles one domain in the following turn. Finally, repetitions due to system language understanding errors are also accounted for. If any of these behaviours occur, the evaluator allows up to `max_rep` repetitions before increasing false positive counts.

System constraint pre-emption The system is unaware of the user goal, so it may express some constraints before they can be provided by the user. The user may repeat some of them to confirm the values, but this may not necessarily occur since the user can accept a constraint through other mechanisms (e.g., acknowledgement, accepting an offer made at the same time). I-GCDF1 detects this system behaviour, adjusting the false negative counts.

3.2 R-GCDF1

Requests repetitions The user may repeat requests to overcome system language understanding errors. In addition to this, the algorithm also accounts for situations where the user informs the request before providing all the constraints and when the system omits responses. Up to `max_rep` per request are allowed before increasing the false positive counts when the repetition matches one of these patterns.

System request pre-emption Requests missing from user turns are searched in system turns to determine if the system has pre-empted the request by offering the information in advance (e.g., when confirming booking details or offering an entity).

4 Sample evaluation results

4.1 Experimental setup

System agent The system agent is a pipeline architecture implemented in the `Convlab2` library (Zhu et al., 2020). It is comprised of a BERT-based (Devlin et al., 2019) natural language understanding (NLU) module, a handcrafted policy, a rule-based DST and a retrieval natural language generation (NLG) module. This model outperforms all other `Convlab2` system configurations (Takanobu et al., 2020).

User agent We evaluate the architecture employed by Takanobu et al. (2020) in their user model based evaluation study. It is comprised of an MILU-based (Hakkani-Tür et al., 2016) NLU module, an agenda-based handcrafted policy (Schatzmann et al., 2007) and a retrieval NLG module.

Interaction setup We extend `Convlab2` by driving the interaction between the two agents by the MultiWOZ 2.1 test set goals (Figure 1). This facilitates comparisons with future work and is

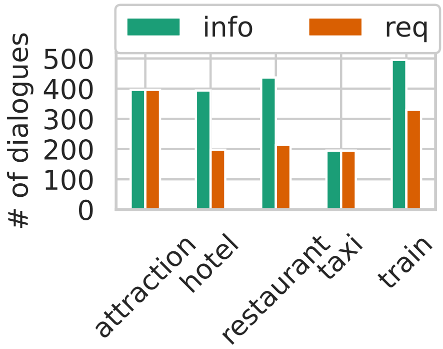


Figure 1: Domain distribution of the user goals in the MultiWOZ 2.1 test set

needed since the Convlab2 goal model does not account for booking failures. The user and system agents interact freely until the conversation ends, for a maximum of 30 turns. The generated utterances, together with the user and system input and output actions are collected.

4.2 Behaviour and error states analysis

4.2.1 Constraints provision

The constraint provision ability of the user model varies significantly across the domains (Table 1).We explain these measurements in detail in the following section.

Domain	Combined	Attraction	Hotel	Restaurant	Taxi	Train
I-GCDF1	0.623	0.786	0.505	0.467	0.482	0.706

Table 1: I-GCDF1 scores. The combined score disregards domain information during counting.

Constraint repetitions Most commonly repetitions occur when the user discusses a booking or after a recommendation is made (Figure 2, recom_book_rep), a user behaviour which is detected by checking if the system has made a recommendation, offered an entity (i.e., presence of {inform, select, recommend}(name|trainID=*) actions) or prompted the user to make a booking (e.g., *Would you like to book a table?*). For the attraction domain, the slots name, type parametrise repeated constraints in 34 dialogues and in 3 diaologues repetitions are parametrised by the area slot. Analysis reveals that these repetitions are triggered by the MILU model, which generates the request(name|type) actions when encountering the phrases *Do you have any specific ideas in mind?* and *Anything in particular that you are looking for?*. Accordingly, the user say they don't care about the name or type of attraction. These questions are very frequently sampled by the NLG model, despite being superfluous: the user always provides enough constraints in the previous turns. The conversation only continues once the aforementioned sentences are not sampled. The NLU model also generates the request(area) action when the words *location, area* are mentioned in a response where the system intention is not to request information, so the user repeats the slot.

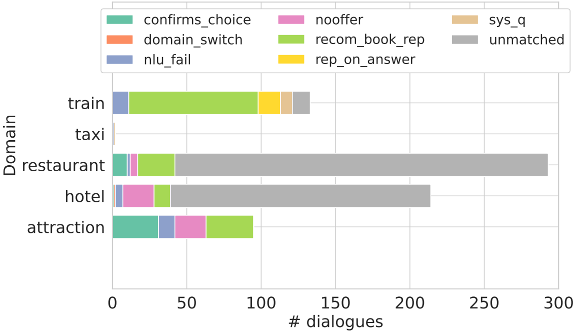


Figure 2: Prevalence of behaviours that lead to constraint repetitions

In the train domain, repetitions occur in 87 dialogues with the constraints on day slot being repeated in 86 of these, departure in 4 and destination only once. day is repeated so often because the systems' booking confirmation *Would you like to take the train on [day]?* is recognised as request(day) or select(day=[day]), which triggers the user policy to repeat the constraint. The NLU model does not correctly identify the offerbooked[3] dialogue act, which leads to this repetition pattern.

Repetition when answering a system request about a different slot (rep_on_answer) is not common and the system does not usually ask questions about constraints the user has already provided (sys_q). Additionaly, system understanding appears robust, and the user does not often have to re-provide information to overcome understanding errors. The user model repeats constraints when providing new information after the system informed the user they could not complete the current task with the specified constraints (no_offer).

A large number of repetitions are unmatched for the restaurant and hotel domains. For the former, in 251 dialogues (57.44% of di-

[3] This act annotates the booking details confirmation on the system side for the train domain.

alogues) these are *booking constraints* (i.e., which are parametrised by the slots `time`, `day`, `people`). The repetitions occur because the user model NLU mislabels the `restaurant-inform(reference=*)` action as `train-offerbooked(reference=*)`. This error causes the constraints to be repeated continuously until the dialogue session ends, so any domains that should have been discussed after the booking are missed, explaining the low taxi I-GCDF1 (Table 1). The issue occurs in the hotel domain for 96% of the 175 dialogues with constraint repetitions. The rest of the repetitions are of the `type=guesthouse` constraint, which the user repeats because the system responses to information requests contain the word *hotel* (e.g., *The hotel adress is `[address]`?*) which is interpreted as `inform(type=hotel)` by the NLU whereas the goal contains the `type=guesthouse` constraint.

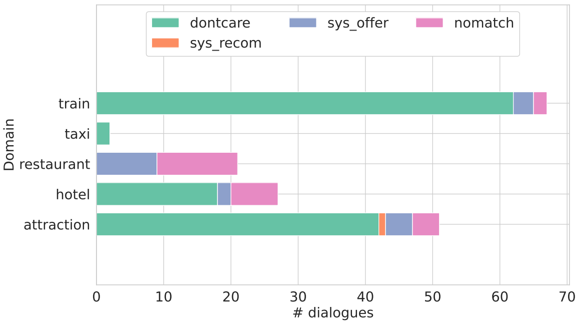
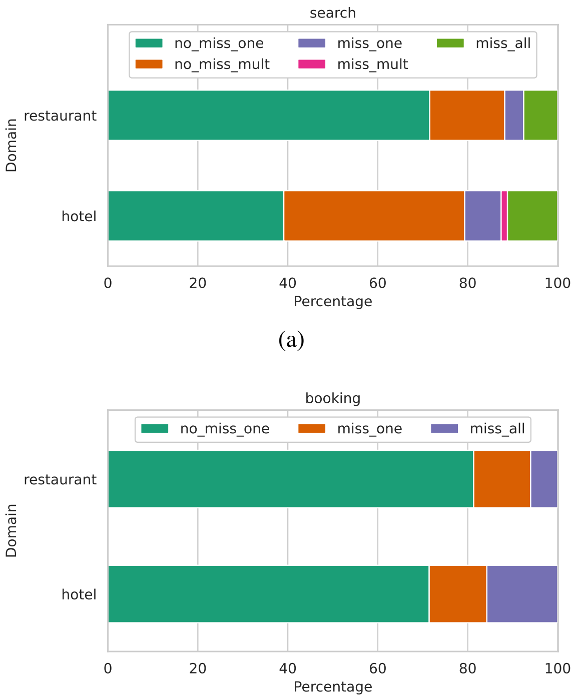


Figure 3: Not-in-goal constraints matching patterns

Not-in-goal constraints The user model frequently generates the `inform([slot_name]=dontcare)` action across all domains except restaurant (Figure 3). As discussed above, for the train and attraction domains these are triggered by the system language choice when confirming entity attributes. However, the slot-value `train-people=dontcare` is generated, suggesting that the model does not appropriately decline the system invitation to book train tickets. In the hotel domain, `dontcare` is generated to handle system requests of information not specified in the goal. In `sys_offer` conversations (Figure 3), the user model selects an entity by naming it or invites the system to choose an option for them by generating the action `inform(choice=any)`. In `notmatch` dialogues, the action `inform(notbook=none)` is generated by user model to decline reservation proposal. The evaluator does not match this act because the MultiWOZ 2.1 annotation system does not contain this slot-value pair. However, the evaluator has a configuration file where not-in-goal slot-value pairs that should be automatically matched can be listed, so modifying the algorithm to account for this situation, unknown at development time, is straightforward.

Constraint expression patterns Long sentences containing a lot of information are challenging for system NLU components. If all the information is provided at once, state tracking modules operating on NLU output are insufficiently tested. We analyse *constraints expression patterns* to understand whether all the search (or booking) constraints for a given domain are communicated in a single turn or across multiple turns.



(a)

(b)

Figure 4: Constraint expression patterns. In `no*` dialogues all goal constraints have been communicated.

The user model is biased toward expressing all the search constraints at once (Figure 4a, `no_miss_one`) for the restaurant domain. Booking constraints are always expressed in the same turn. The baseline fails to search for an entity in just over 10% of the dialogues in the hotel domain and in close to 20% of the conversations in the same domain it does not attempt booking (Figure 4a, `miss_all`). In fact, Figure 5 shows that the baseline user model often fails to complete multi-

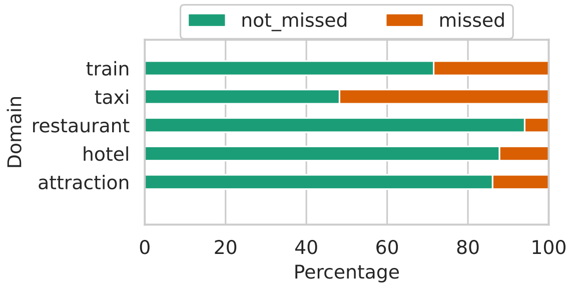


Figure 5: Percentage of dialogues where a domain in goal is not discussed by the baseline model

domain conversations. The taxi and train domains are frequently not discussed, explaining the poor performance reported in Table 1. Often, this is due to the failure of the NLU model to detect the `reference` and `entrance_fee` slots, which cause the user to indefinitely repeat the booking constraints or request the entrance fee. Hence, multi-domain dialogue simulation is very sensitive to natural language understanding capability.

Analysis of ability to request information

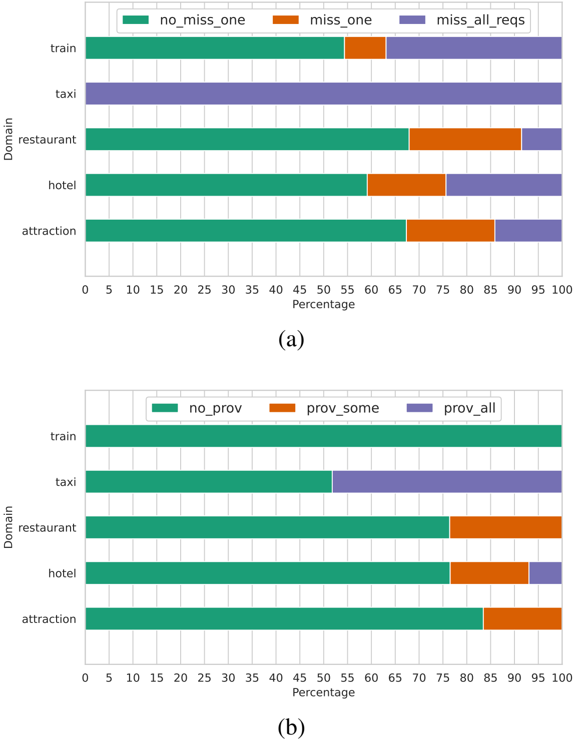
Model's ability to request information also varies significantly across domains (Table 2) and requests are always expressed in the same turn (Figure 6a).

Domain	Combined	Attraction	Hotel	Restaurant	Taxi	Train
R-GCDF1	0.692	0.719	0.770	0.873	0.482	0.658

Table 2: R-GCDF1 scores

The model either requests all information (`no_miss_one`), misses one or more requests (`miss_one`) or does not request any information (`miss_all_reqs`). The last pattern occurs because the two agents may get stuck in a question-answer loop. The `prov_all` category in Figure 6b shows the system may occasionally provides all the requests before the user can inform them. Taken together Figures 6a and 6b show that the user model makes all requests unless the system already provides the information: the scores in Table 2 are affected by the model's inability to complete multi-domain conversations and by requests repetition.

Figure 7 shows causes of requests repetition. The `delayed_resp` and `nlu_fail` categories contain the same dialogues, identifying conversations where the user model repeatedly requests information because the system does not immediately provide an answer. In `repeat_after_answer` dialogues user NLU errors for slots such as `attraction-type` or `*-reference` lead to dialogue loops.



(a)

(b)

Figure 6: Requests expression patterns

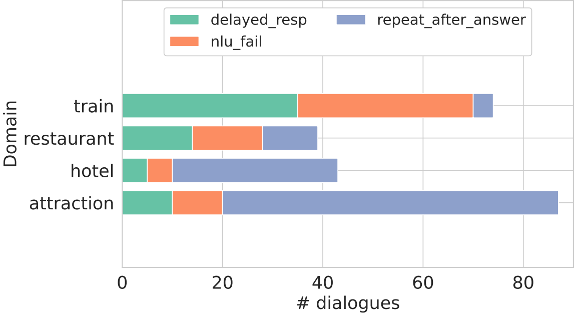


Figure 7: Information requests repetition reasons

5 Conclusion

We proposed the GCDF1 framework and used it to conduct a detailed performance analysis of a user model. Understanding error states supports model improvement: for example, we identified that the user model analysed does not understand the `reference` slot. Hence, the user NLU model could be finetuned to resolve this error. The template NLG module was also shown to affect dialogue structure and quality. Future work could assess other `Convlab2` user models and extend our approach to larger corpora with more complex dialogue flows such as SGD (Rastogi et al., 2020).

Acknowledgements

This work was supported by EPSRC grant EP/R513180/1. Bo-Hsiang Tseng is supported by Cambridge Trust and the Ministry of Education, Taiwan.

References

Zoraida Callejas, David Griol, and Klaus-Peter Engelbrecht. 2012. Assessment of user simulators for spoken dialogue systems by means of subspace multidimensional clustering. In *INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association, Portland, Oregon, USA, September 9-13, 2012*, pages 250–253. ISCA.

Heriberto Cuayáhuitl, Steve Renals, Oliver Lemon, and Hiroshi Shimodaira. 2005. Human-computer dialogue simulation using hidden markov models. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2005.*, pages 290–295. IEEE.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 879–885. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar ands Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428. European Language Resources Association.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 715–719. ISCA.

Sangkeun Jung, Cheongjae Lee, Kyungduk Kim, Minwoo Jeong, and Gary Geunbae Lee. 2009. Data-driven user simulation for automated evaluation of spoken dialog systems. *Computer Speech & Language*, 23(4):479–509.

Simon Keizer, Milica Gasic, Filip Jurcícek, François Mairesse, Blaise Thomson, Kai Yu, and Steve J. Young. 2010. Parameter estimation for agenda-based user simulation. In *Proceedings of the SIGDIAL 2010 Conference, The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 24-15 September 2010, Tokyo, Japan*, pages 116–123. The Association for Computer Linguistics.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. 2019. Convlab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 3: System Demonstrations*, pages 64–69. Association for Computational Linguistics.

Nurul Lubis, Christian Geishauser, Michael Heck, Hsien-Chin Lin, Marco Moresi, Carel van Niekerk, and Milica Gasic. 2020. LAVA: latent action spaces via variational auto-encoding for dialogue policy optimization. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 465–479. International Committee on Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. SOLOIST: building task bots at scale with transfer learning and machine teaching. *Trans. Assoc. Comput. Linguistics*, 9:907–824.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New*

York, NY, USA, February 7-12, 2020, pages 8689–8696. AAAI Press.

Jost Schatzmann, Kallirroi Georgila, and Steve J. Young. 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue, SIGdial 2005, Lisbon, Portugal, 2-3 September 2005*, pages 45–54. Special Interest Group on Discourse and Dialogue (SIGdial).

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve J. Young. 2007. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*, pages 149–152. The Association for Computational Linguistics.

Jost Schatztnann, Matthew N Stuttle, Karl Weilhammer, and Steve Young. 2005. Effects of the user model on simulation-based learning of dialogue strategies. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2005.*, pages 220–225. IEEE.

Ryuichi Takanobu, Qi Zhu, Jinchao Li, Baolin Peng, Jianfeng Gao, and Minlie Huang. 2020. Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 297–310. Association for Computational Linguistics.

Kai Wang, Junfeng Tian, Rui Wang, Xiaojun Quan, and Jianxing Yu. 2020. Multi-domain dialogue acts and response co-generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7125–7134. Association for Computational Linguistics.

Jason D. Williams. 2007. A method for evaluating and comparing user simulations: The cramér-von mises divergence. In *IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU 2007, Kyoto, Japan, December 9-13, 2007*, pages 508–513. IEEE.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2020. UBAR: towards fully end-to-end task-oriented dialog systems with GPT-2. *CoRR*, abs/2012.03539.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 142–149. Association for Computational Linguistics.