Combining Open Domain Question Answering with a Task-Oriented Dialog System

Jan Nehring, Nils Feldhus, Akhyar Ahmed, Harleen Kaur
German Research Centre for Artificial Intelligence (DFKI)
Berlin, Germany
firstname.lastname@dfki.de

Abstract

We apply the modular dialog system framework to combine open-domain question answering with a task-oriented dialog system. This meta dialog system can answer questions from Wikipedia and at the same time act as a personal assistant. The aim of this system is to combine the strength of an open-domain question answering system with the conversational power of task-oriented dialog systems. After explaining the technical details of the system, we combined a new dataset out of standard datasets to evaluate the system. We further introduce an evaluation method for this system. Using this method, we compare the performance of the non-modular system with the performance of the modular system and show that the modular dialog system framework is very suitable for this combination of conversational agents and that the performance of each agent decreases only marginally through the modular setting.

1 Introduction

Nehring and Ahmed (2021) defined a modular dialog system (MDS) as a dialog system that consists of multiple modules. In this paper, we want to use this framework to combine a task-oriented dialog system (TODS) with an open-domain question answering system (ODQA). For our experiments, we construct the TODS using the Frankenbot framework trained on the CLINC150 dataset (Larson et al., 2019) to build a dialog system from the personal assistant domain. For the ODQA system, we use DrQA (Chen et al., 2017) which uses Wikipedia among other corpora as knowledge sources.

The resulting meta dialog system combines the strengths and evens out the weaknesses of both approaches. ODQA can answer a wide range of questions. Furthermore, one can easily extend the system with new information as it only requires unstructured text as a knowledge base. It is not trivial to fix the mistakes of ODQA, so we have little control over the system.

Creating the TODS on the other hand requires a lot of manual work. It is not feasible to cover the amount of questions an ODQA system can answer. Therefore, the amount of topics that the TODS can talk about is rather limited. The strength of the TODS approach is its fine-grained control. Errors can easily be corrected by adding a small amount of training data and retraining the model. A TODS cannot only answer questions, but it can also understand other user queries. For example, the TODS can understand greetings and respond with a greeting, a task that is not possible for ODQA systems. Another strength of TODS is the possibility to create complex dialogs spanning multiple turns using a dialog manager.

Question answering has been augmented with TODS before (Banchs et al., 2013; D'Haro et al., 2015; Coronado et al., 2015; Podgorny et al., 2019). In this work, we apply the MDS framework to the combination of an ODQA system and a TODS. The other works mentioned here usually performed a user-based evaluation showing that the meta dialog system works. We present a method to evaluate such a system automatically. The method inspects module selection, ODQA and TODS individually and measures the performance change of those from the non-modular to the modular scenario. We show that the performance drop is very low because the module selection performs very well in our setup with an f1-measure of 0.964.

The remainder of this paper is structured as follows: Section 2 gives an overview over the background related to conversational agents, DrQA, Frankenbot, the MDS framework, evaluation measures and datasets. Next, section 3 explains how we created our dataset out of existing datasets. The setup of our MDS and implementation details are

Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering, pages 38–45

covered in section 4. Section 5 introduces our evaluation methodology, followed by the results and their discussion in 6. The following sections discuss conclusions (7) and future work (8).

2 Background

2.1 Conversational Agents

Zhu et al. (2021) define ODQA as the task of identifying answers to natural questions from a large corpus of documents. A typical system works in two steps: First, it selects a document from the corpus that contains the answer. Second, they generate the answer from this document, either in natural language (generative QA) or as the span of text containing the answer (extractive QA) (Zhu et al., 2021). Examples of such systems are DrQA (Chen et al., 2017), QuASE (Sun et al., 2015), YodaQA (Baudiš and Šedivý, 2015) and DeepQA (Ferrucci et al., 2010).

There are many ways to create a TODS. In this paper, we limit ourselves to TODS that build on the GUS architecture (Bobrow et al., 1977). Many modern chatbot frameworks like Amazon Alexa[1], Google Dialogflow[2] and others build on this surprisingly old architecture (Jurafsky and Martin, 2020). In GUS, each user utterance is processed by the Natural Language Understanding (NLU) unit. The NLU first performs intent classification which is the task of assigning one of many pre-defined user intents to the utterance. An important concept of GUS is the semantic frame which defines a set of slots. These slots represent information that the dialog system needs to understand and fill in from the user utterances in order to fulfill a task. For example, the semantic frame "restaurant reservation" consists of the slots "number of persons", "date and time". When a user utters "I want to book a table for three persons" the TODS can detect the intent "table reservation" and fill the slot "number of persons". The output of the NLU is fed into the Dialog Manager (DM). The DM keeps track of the dialog state and can be either rule-based or machine-learned. The dialog manager can, for example, decide to ask about the date and time of the reservation if the intent "table reservation" is detected, the slot "number of persons" is filled out but the slot "date and time" is still missing. Answers are usually based on the dialog state. In this paper, we limit ourselves to rule-based DMs and answers written manually by the chatbot designers. For a more detailed discussion of TODS, we refer to Jurafsky and Martin (2020).

A MDS, as defined by Nehring and Ahmed (2021), combines multiple dialog systems to form a meta dialog system. Each of these dialog systems is called a module. For each incoming user utterance, the module selection component chooses the module that produces the answer. The MDS framework does not define how to implement the module selection component, the actual implementation can vary from MDS to MDS. Another characteristic of the MDS is that modules are independent from each other, do not share a common state and do not share models or parameters. The MDS architecture consists of multiple subsequent models, in contrast to a joint architecture that uses one joint module for all tasks. This allows the combination of different, usually incompatible technologies under one framework.

2.2 DrQA

Chen et al. (2017) introduced the extractive question answering system DrQA. To answer a question, DrQA starts with an information retrieval step to detect the relevant article from the Wikipedia corpus. The information retrieval is based on bigram hashing and TF-IDF metrics. In the second step DrQA uses a recurrent neural network to identify the answer span in the retrieved document used as context. DrQA is trained on multiple datasets, including the Stanford Question Answering Dataset (Rajpurkar et al., 2016).

2.3 Frankenbot

Frankenbot is a framework for TODS that is capable of holding longer conversations spanning multiple turns by mapping the conversation onto pre-defined dialog trees. It consists of a set of nodes, each containing a possible answer. Furthermore, each node contains a pre-defined condition. Conditions can be e.g. "The detected intent is X", "The detected intent is Y and the slot Z is not filled out" and similar.

Frankenbot uses the Dual Intent and Entity Transformer (Bunk et al., 2020) as NLU for intent classification and slot filling. First, each utterance is processed by the NLU. Next, the dialog manager determines which nodes are active, meaning that they are candidates for the answer generation. Top-level nodes are always active. Nested nodes are

[1] https://developer.amazon.com/en-US/alexa

[2] https://cloud.google.com/dialogflow

only active if their parent node produced the answer in the previous turn. After the active nodes have been determined, the DM evaluates the conditions of all active nodes in a certain order (depth-first). The first node whose condition matches produces the answer.

2.4 Evaluation measures

In extractive QA, F1 is the standard measure (Chen et al., 2019). It is computed over tokens in the candidate and the reference. Automatic evaluation measures in QA are flawed because they rely on a gold standard of correct answers (Chen et al., 2019). When the tested QA system gives a correct answer which is not defined in the gold standard, the correct answer will be counted as an error. We note that there is still no consensus about an automated metric for question answering that correlates well with human judgment (Chen et al., 2019).

Evaluating a TODS in an automated manner suffers from similar problems but we can evaluate individual components of the TODS. Intent classification is the task of labeling a user utterance with one of a set of predefined intents. F1 scores are usually used to evaluate the performance of intent classification of the NLU component (Deriu et al., 2019; Liu et al., 2019).

2.5 Datasets

The CLINC150 dataset (Larson et al., 2019) is a dataset to evaluate the intent classification performance of a TODS for the personal assistant domain. Crowd workers wrote 22,500 user utterances for 150 intents. It contains the same amount of utterances for each intent. The dataset contains 1,200 out-of-scope utterances that belong to none of the intents which we did not use.

The Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) is a standard dataset for question answering which contains over 100.000+ questions. It contains a list of contexts, and each of them is a paragraph of text. The questions are always related to a context. Furthermore, it contains answers to the questions that are annotated as a span of text from the context. It is split into a training (80%), a development (10%), and a test (10%) dataset.

2.6 Hybrid Dialog Systems

There are many approaches how to combine several dialog tasks in one dialog system. One approach is a hierarchical architecture composed of several agents. They use a classifier to select the right agent for each utterance (Coronado et al., 2015; Banchs et al., 2013; Planells et al., 2013; Pichl et al., 2018) or a ranking approach that generates answers by each DS and then selects the best answer (Song et al., 2018; Tanaka et al., 2019; Paranjape et al., 2020). Other approaches use a joint architecture to solve multiple dialog tasks (Lewis et al., 2020; Shuster et al., 2020).

3 Creating a combined dataset for question answering and intent recognition

To our knowledge, no dataset exists to evaluate a TODS and a QA system at the same time. Therefore, we combined the SQuAD and CLINC150 datasets to form a single dataset for the evaluation of the combined system. Beyond the original labels from CLINC150 (intents) and SQuAD (answers), each sample has an additional label for the module selection which we call the true module. One must note that due to the nature of the MDS, we need to train each module on its own. DrQA is already pre-trained on the SQuAD training dataset and we did not retrain it. We kept 50% of the CLINC150 samples to train the Frankenbot and call this dataset $train_F$. We then train the module selection on the $train_MS$ part of the dataset. For parameter selection, we reserve $valid_MS$ samples. Finally, we use the dataset $test_{full}$ for the evaluation of the full system.

dataset	number of samples
all	32,390
$\text{all}_{\text{CLINC150}}$	21,820
$\text{all}_{\text{SQuAD}}$	10,570
train_{F}	11,250
train_{MS}	7,900
test_{MS}	2,634
$\text{test}_{\text{full}}$	10,606

Table 1: Dataset statistics

Table 1 shows statistics about this dataset. We used 11,250 samples to train the Frankenbot TODS ($train_F$). We do not need a training set for DrQA in our dataset and use the development subset of SQuAD only for $train_F$, $valid_MS$, and $test_{full}$.

We reserved 10,534 samples to train and validate the module selection. With a 75-25 split, it results in 7,900 samples for training ($train_{MS}$) and 2,634 samples for testing ($valid_{MS}$). For the

evaluation of the full system, we reserved 10,606 samples ($test_{full}$). We aimed for an equal amount of samples from CLINC150 and SQuAD in the train_{MS}, test_{MS}, and $\text{test}_{\text{full}}$ sections of the dataset and therefore randomly subsampled the CLINC150 dataset.

The SQuAD dataset contains multiple questions for each context, e.g. it contains 810 questions related to a short paragraph about a Super Bowl game. If we split the SQuAD dataset randomly, the module selection might overfit on such statistical cues, learning that the word Super Bowl is a hint that this utterance is aimed at the ODQA system. Therefore, we did not assign the SQuAD questions to the datasets at random, but split it along those contexts so that each context appears in either $train_{MS}$ or $test_{full}$, but not in both.

While the input questions of SQuAD use correct casing, the user utterances of CLINC150 use a mix of correct casing and lower casing. Furthermore, CLINC150 uses punctuation marks only sometimes while SQuAD always uses punctuation marks, mostly question marks. This makes our dataset unrealistic because it leads to distinguishing criteria which will not occur in real world data. We removed these differences to make the utterances more uniform and therefore more realistic by lowercasing all user utterances and removing all punctuation marks.

Many questions from SQuAD can be answered only when the context of the question is known. For example the question "Who approved of this plan?" is only answerable with the context paragraph at hand. DrQA retrieves the context from Wikipedia and therefore cannot answer this question. To get a more realistic impression of the performance of the DrQA module, we manually annotated 100 questions from $test_{full}$ that can be answered in the ODQA scenario.

We published the dataset under the Creative Commons Attribution Share Alike license CC-BY-SA 4.0 under this link[3].

4 Modular Dialog System

Figure 1 shows the architecture of the MDS that we used in our experiments. It contains two modules: The ODQA system DrQA and the TODS Frankenbot.

The module selection component decides for each user utterance which module will answer the utterance. We formulate this as a text classification task with the candidate modules as target classes, i.e., the module selection predicts each incoming user utterance as either DrQA or Frankenbot. We used BERT with a sequence classification head (Devlin et al., 2019) for this classification task. It is trained on dataset train_{MS} and evaluated on test_{MS}.

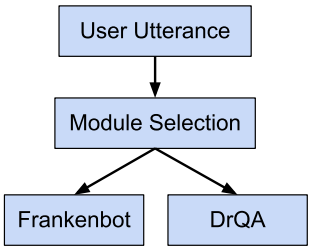

Figure 1: Architecture of the MDS

4.1 Modules

Our MDS consists of the two modules DrQA and Frankenbot. The DrQA module uses the implementation published by its authors[4] without further modification.

The Frankenbot module uses the Frankenbot framework as the technical backend for a TODS. We used the CLINC150 dataset to train the NLU. The CLINC150 dataset contains single turn utterances only so our dialog system does not use a deep dialog management either. The user makes his query, for example "Put the lights on". Using this dataset the dialog system cannot ask back, for example "In which room?".

Frankenbot can also predict samples as out of scope (oos) when the confidence of the intent classification is below a certain manually defined threshold. The oos class indicates that Frankenbot cannot answer this utterance.

5 Evaluation

5.1 Evaluation of the single modules

Following the approach of the SQuAD dataset (Rajpurkar et al., 2016), we measure the quality of the DrQA module using token-based F1 scores. Following the approach of the CoQA challenge (Reddy et al., 2019), we did not take the exact match measure into account.

Next we want to evaluate the quality of Frankenbot in the MDS. One can evaluate many aspects of

[3] link will be available in camera ready version

[4] https://github.com/facebookresearch/DrQA

Frankenbot. Since Frankenbot's answers are manually predefined, we make the assumption that they are correct and we do not want to evaluate them here. Another factor in the evaluation of dialog systems is the dialog management. In this work, we do not want to evaluate the quality of the dialog management but the quality of the modular dialog system. We argue that Frankenbot can produces the correct answer as long when the intent classification produced the correct intent. Therefore, we evaluate the quality of the intent classification to estimate the quality of the TODS. We evaluate the intent classification using F1 scores. It is a multi-class classification setting and we use the micro-average to calculate a weighted final score out of the F1 scores for each intent.

5.2 Evaluation of module selection

Module selection is a classification task and therefore its evaluation is straightforward. We use F1 scores to calculate the performance of the module selection. We use the $test_{full}$ dataset to calculate the scores of module selection. We repeated this evaluation ten times and averaged the results.

5.3 Evaluation of the full system

Here, we present a framework that can evaluate both systems jointly. The evaluation framework can then compare the performance change from a non-modular dialog system to a modular system. We also repeated this evaluation ten times and averaged the results.

For a fine-grained evaluation of the full system, we calculate scores for the non-modular and for the modular scenario. The *non-modular* scenario evaluates how the system would perform if it was not modular. Bypassing the module selection, it evaluates each module on its own data. In our case, it uses the CLINC150 part of the test dataset to evaluate the Frankenbot and the SQuAD part of the test dataset to evaluate the SQuAD.

The *modular scenario* evaluates the MDS. In this scenario, we evaluate each module on its own data again, but this time including the module selection. In this setting, it is possible that the module selection makes a mistake and incorrectly assigns a sample to the other module.

In case we are evaluating Frankenbot and a Frankenbot sample gets confused as DrQA, we label it with the intent class "oos" for out-of-scope. This will lower the F1 score of Frankenbot, but it will not affect the score of DrQA.

In case we misclassify a sample during DrQA's evaluation as Frankenbot, we assume that the dialog system answered with an empty string and continue the evaluation. We could use the actual answer of the dialog system instead of this arbitrary string, but we believe that the empty string provides a more stable error because it does not produce random matches on the token basis and has the same length always. Again, this misclassified sample only produces an error in the DrQA module and not in the Frankenbot module.

To get a joint score for the whole system, we take the macro-average between the F1 scores of intent recognition and QA and name it *joint F1*. This makes sense for the modular scenario only.

5.4 Questions that are answerable in ODQA

As stated earlier, we found that many questions from SQuAD are not answerable in the ODQA scenario. Therefore, we calculate the evaluation once for the full dataset and once only for questions that are answerable in the ODQA scenario. This evaluation includes all samples from CLINC150 and 100 questions from SQuAD that we manually annotated for being answerable without knowledge of the context document.

This is an additional error analysis of our specific system and not part of the evaluation method that we suggest for this kind of MDS.

6 Results and Discussion

Table 2 shows the results of the non-modular and modular evaluation across the evaluation using the full dataset and the subset of the dataset containing only questions that DrQA can answer.

	F1 Frankenbot	**F1 DrQA**	**joint F1**
Full evaluation data			
non-modular	0.897	0.340	-
modular	0.895	0.326	0.611
Questions that DrQA can answer			
non-modular	0.897	0.451	-
modular	0.892	0.442	0.670

Table 2: Results

6.1 Results of module evaluation

The evaluation of Frankenbot's NLU reports an F1 score of 0.897. The numbers are the same for the evaluation on the full data and on questions that

DrQA can answer, because it is an evaluation on the same data.

The evaluation of DrQA reports an F1 score of 0.36 in the non-modular setting which is comparable to the results reported in the original paper (Chen et al., 2017). The F1 score rises to 0.451 in the subset of SQuAD questions that DrQA can answer.

6.2 Results of Module Selection

The F1 score of the module selection is 0.964. This shows that the module selection does not introduce a large error source. This is different from the findings of Nehring and Ahmed (2021) where the modular setting introduced a large error. We believe that the high quality of module selection is partly a result of the different natures of the datasets. CLINC150 mostly contains commands like "Switch on the light" or "Play the next song in the radio" while SQuAD contains only questions. Our BERT-based classifier can easily distinguish between the two. We conclude from this result that the MDS framework is suitable for the combination of ODQA and TODS.

6.3 Results of the full system evaluation

The very high performance of module selection reflects itself in the results of the evaluation of the modular setting. Since the module selection is almost always correct, it does not introduce a significant additional error.

It is obvious that the quality is lower in the modular scenario compared to the non-modular scenario: The module selection is an additional source of error only and the performance of a module cannot improve through the module selection. This low performance drop is an indicator that the MDS framework is very suitable for this combination of dialog systems.

6.4 Error analysis of module selection

Table 3 shows the confusion matrix of the module selection over dataset $test_{full}$. In the former results sections, we repeated each experiment 10 times and averaged the results. In this section we show the results of one of these 10 module selections.

The amount of DrQA samples being misclassified as Frankenbot is 26x higher than the amount of Frankenbot samples being misclassified as DrQA. We assume that this is due to the nature of the datasets. Each of the intents from the CLINC150 dataset describe a narrow topic and therefore more suitable for the text classification of the module selection. The questions of DrQA do not share a common topic and are therefore harder to detect.

	Frankenbot	**DrQA**
Frankenbot	5,268	16
DrQA	421	4,903

Table 3: Confusion Matrix of module selection with the predicted module in the rows and true label in the columns and the true module in the columns.

7 Conclusion

We used the MDS framework to combine TODS and ODQA using the example of DrQA and Frankenbot. Using this framework, one can extend the capabilities of ODQA with the conversational capabilities of a TODS. Further, we introduced an evaluation method that a) evaluates the performance of module selection and b) compares the performance of the underlying ODQA and TODS systems in the modular and in the non-modular setting.

The evaluation showed that DrQA and Frankenbot work very well together as a MDS. The MDS introduces only a minimal additional error.

We believe that the MDS framework is especially suitable for practical applications because one can extend an existing TODS with a ODQA system or vice versa easier using MDS compared to a framework that performs TODS and ODQA together in a joint model. Although we did not prove it, we expect that when we exchange one of the modules, e.g. the Frankenbot system with Rasa[5], Google Dialogflow or IBM Watson Assistant[6], the performance of the MDS will change only in that module.

We present this evaluation framework for our specific use case, but we expect it to generalize to other settings as well such as using more than two modules. It can also work with other performance measures than F1 scores, although one needs to think about how to calculate a joint score out of different scores and how to deal with errors of the module selection in the modular scenario.

[5] https://rasa.com

[6] https://www.ibm.com/cloud/watson-assistant

8 Future Work

The module selection showed very good results. In future work, we want to try the module selection with other or smaller datasets to find out if this high performance is stable across datasets.

We showed that the combination of a single-turn ODQA with a single-turn TODS works very well. An interesting extension of this paper would be to use a multi-turn ODQA as in the CoQA challenge and a multi-turn TODS and find a way to automatically evaluate both together.

9 Acknowledgment

This research was partially supported by the German Federal Ministry of Education and Research (BMBF) through the project SIM3S (19F2058A).

References

Rafael E. Banchs, Ridong Jiang, Seokhwan Kim, Arthur Niswar, and Kheng Hui Yeo. 2013. AIDA: Artificial Intelligent Dialogue Agent. In *Proceedings of the SIGDIAL 2013 Conference*, pages 145–147, Metz, France. Association for Computational Linguistics.

Petr Baudiš and Jan Šedivý. 2015. Modeling of the Question Answering Task in the YodaQA System. In *Proceedings of the 6th International Conference on Experimental IR Meets Multilinguality, Multimodality, and Interaction - Volume 9283*, CLEF'15, page 222–228, Berlin, Heidelberg. Springer-Verlag.

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry Thompson, and Terry Winograd. 1977. GUS, a Frame-Driven Dialog System. *Artif. Intell.*, 8(2):155–173.

Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. 2020. DIET: Lightweight Language Understanding for Dialogue Systems.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Evaluating Question Answering Evaluation. In *MRQA@EMNLP*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Miguel Coronado, Carlos A. Iglesias, and Alberto Mardomingo. 2015. A Personal Agents hybrid architecture for question answering featuring social dialog. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–8.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2019. Survey on Evaluation Methods for Dialogue Systems. *Artificial Intelligence Review*, 54(1):755–810.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Luis Fernando D'Haro, Seokhwan Kim, Kheng Hui Yeo, Ridong Jiang, Andreea I. Niculescu, Rafael E. Banchs, and Haizhou Li. 2015. CLARA: A Multifunctional Virtual Agent for Conference Support and Touristic Information. In *Natural Language Dialog Systems and Intelligent Assistants*, pages 233–239. Springer International Publishing.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer, and Chris Welty. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79.

Dan Jurafsky and James H. Martin. 2020. *Speech and Language Processing : An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Prentice Hall, Upper Saddle River, N.J.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

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

Jiao Liu, Yanling Li, and Min Lin. 2019. Review of Intent Detection Methods in the Human-Machine Dialogue System. *Journal of Physics: Conference Series*, 1267:12059.

Jan Nehring and Akhyar Ahmed. 2021. Normalisierungsmethoden für Intent Erkennung Modularer Dialogsysteme. In *Tagungsband der 32. Konferenz. Elektronische Sprachsignalverarbeitung (ESSV-2021), March 3-5, Berlin, Germany*. TUDpress.

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

Jan Pichl, Petr Marek, Jakub Konrád, Martin Matulík, Hoang Long Nguyen, and Jan Šedivý. 2018. Alquist: The alexa prize socialbot.

Joaquin Planells, Lluís F. Hurtado, Encarna Segarra, and Emilio Sanchis. 2013. A multi-domain dialog system to integrate heterogeneous spoken dialog systems. In *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013*, pages 1891–1895. ISCA.

Igor Podgorny, Yason Khaburzaniya, and Jeff Geisler. 2019. Conversational Agents and Community Question Answering. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*. ACM.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Kurt Shuster, Da Ju, Stephen Roller, Emily Dinan, Y-Lan Boureau, and Jason Weston. 2020. The dialogue dodecathlon: Open-domain knowledge and image grounded conversational agents. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2453–2470, Online. Association for Computational Linguistics.

Yiping Song, Cheng-Te Li, Jian-Yun Nie, Ming Zhang, Dongyan Zhao, and Rui Yan. 2018. An Ensemble of Retrieval-Based and Generation-Based Human-Computer Conversation Systems. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, {IJCAI-18}*, pages 4382–4388. International Joint Conferences on Artificial Intelligence Organization.

Huan Sun, Hao Ma, Wen-tau Yih, Chen-Tse Tsai, Jingjing Liu, and Ming-Wei Chang. 2015. Open Domain Question Answering via Semantic Enrichment. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 1045–1055, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Ryota Tanaka, Akihide Ozeki, Shugo Kato, and Akinobu Lee. 2019. An Ensemble Dialogue System for Facts-Based Sentence Generation. *CoRR*, abs/1902.01529.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering.