ADVISER: A Toolkit for Developing *Multi-modal*, *Multi-domain* and *Socially-engaged* Conversational Agents

Chia-Yu Li, Daniel Ortega, Dirk Väth, Florian Lux,
Lindsey Vanderlyn, Maximilian Schmidt, Michael Neumann, Moritz Völkel,
Pavel Denisov, Sabrina Jenne, Zorica Kacarevic and Ngoc Thang Vu*
Institute for Natural Language Processing (IMS), University of Stuttgart
`thangvu@ims.uni-stuttgart.de`

Abstract

We present ADVISER[1] - an open-source, multi-domain dialog system toolkit that enables the development of multi-modal (incorporating speech, text and vision), socially-engaged (e.g. emotion recognition, engagement level prediction and backchanneling) conversational agents. The final Python-based implementation of our toolkit is flexible, easy to use, and easy to extend not only for technically experienced users, such as machine learning researchers, but also for less technically experienced users, such as linguists or cognitive scientists, thereby providing a flexible platform for collaborative research.

1 Introduction

Dialog systems or chatbots, both text-based and multi-modal, have received much attention in recent years, with an increasing number of dialog systems in both industrial contexts such as Amazon *Alexa*, Apple *Siri*, Microsoft *Cortana*, Google *Duplex*, *XiaoIce* (Zhou et al., 2018) and Furhat[2], as well as academia such as *MuMMER* (Foster et al., 2016) and *Alana* (Curry et al., 2018). However, open-source toolkits and frameworks for developing such systems are rare, especially for developing multi-modal systems comprised of speech, text, and vision. Most of the existing toolkits are designed for developing dialog systems focused only on core dialog components, with or without the option to access external speech processing services (Bohus and Rudnicky, 2009; Baumann and Schlangen, 2012; Lison and Kennington, 2016; Ultes et al., 2017; Ortega et al., 2019; Lee et al., 2019).

To the best of our knowledge, there are only two

---

* All authors contributed equally.
[1] Link to open-source code: `https://github.com/DigitalPhonetics/adviser`
[2] https://docs.furhat.io

toolkits, proposed in (Foster et al., 2016) and (Bohus et al., 2017), that support developing dialog agents using multi-modal processing and social signals (Wagner et al., 2013). Both provide a decent platform for building systems, however, to the best of our knowledge, the former is not open-source, and the latter is based on the .NET platform, which could be less convenient for non-technical users such as linguists and cognitive scientists, who play an important role in dialog research.

In this paper, we introduce a new version of ADVISER - previously a text-based, multi-domain dialog system toolkit (Ortega et al., 2019) - that supports multi-modal dialogs, including speech, text and vision information processing. This provides a new option for building dialog systems that is open-source and Python-based for easy use and fast prototyping. The toolkit is designed in such a way that it is modular, flexible, transparent, and user-friendly for both technically experienced and less technically experienced users.

Furthermore, we add novel features to ADVISER, allowing it to process social signals and to incorporate them into the dialog flow. We believe that these features will be key to developing human-like dialog systems because it is well-known that social signals, such as emotional states and engagement levels, play an important role in human computer interaction (McTear et al., 2016). However in contrast to open-ended dialog systems (Weizenbaum, 1966), our toolkit focuses on task-oriented applications (Bobrow et al., 1977), such as searching for a lecturer at the university (Ortega et al., 2019). The purpose we envision for dialog systems developed using our toolkit is not the same as the objective of a social chatbot such as *XiaoIce* (Zhou et al., 2018). Rather than promoting "an AI companion with an emotional connection to satisfy the human need for communication, affection, and social belonging" (Zhou et al., 2018), ADVISER

helps develop dialog systems that support users in efficiently fulfilling concrete goals, while at the same time considering social signals such as emotional states and engagement levels so as to remain friendly and likeable.

2 Objectives

The main objective of this work is to develop a multi-domain dialog system toolkit that allows for multi-modal information processing and that provides different modules for extracting social signals such as emotional states and for integrating them into the decision making process. The toolkit should be easy to use and extend for users of all levels of technical experience, providing a flexible collaborative research platform.

2.1 Toolkit Design

We extend and substantially modify our previous, text-based dialog system toolkit (Ortega et al., 2019) while following the same design choices. This means that our toolkit is meant to optimize the following four criteria: *Modularity*, *Flexibility*, *Transparency* and *User-friendliness at different levels*. This is accomplished by decomposing the dialog system into independent modules (services), which in turn are either rule-based, machine learning-based or both. These services can easily be combined in different orders/architectures, providing users with flexible options to design new dialog architectures.

2.2 Challenges & Proposed Solutions

Multi-modality The main challenges in handling multi-modality are a) the design of a synchronization infrastructure and b) the large range of different latencies from different modalities. To alleviate the former, we use the publisher/subscriber software pattern presented in section 4 to synchronize signals coming from different sources. This software pattern also allows for services to run in a distributed manner. By assigning computationally heavy tasks such as speech recognition and speech synthesis to a more powerful computing node, it is possible to reduce differences in latency when processing different modalities, therefore achieving more natural interactions.

Socially-Engaged Systems Determining the ideal scope of a socially-engaged dialog system is a complex issue, that is which information should be extracted from users and how the system can best
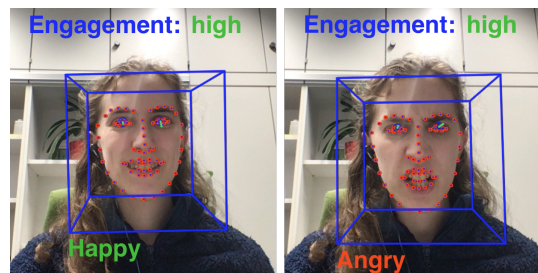

Figure 1: Tracking emotion states and engagement levels using multi-modal information.

react to these signals. Here we focus on two major social signals: emotional states and engagement levels (see section 3.1), and maintain an internal user state to track them over the course of a dialog. Note that the toolkit is designed in such a way that any social signal could be extracted and leveraged in the dialog manager. In order to react to social signals extracted from the user, we provide an initial affective policy module (see section 3.5) and an initial affective NLG module (see section 3.7), which could be easily extended to more sophisticated behavior. Furthermore, we provide a backchanneling module that enables the dialog system to give feedback to users during conversations. Utilizing these features could lead to increased trust and enhance the impression of an empathetic system.

3 Functionalities

3.1 Social Signal Processing

We present the three modules of ADVISER for processing social signals: (a) emotion recognition, (b) engagement level prediction, and (c) backchanneling. Figure 1 illustrates an example of our system tracking emotion states and engagement levels.

Multi-modal Emotion Recognition For recognizing a user's emotional state, all three available modalities – text, audio, and vision – can potentially be exploited, as they can deliver complementary information (Zeng et al., 2009). Therefore, the emotion recognition module can subscribe to the particular input streams of interest (see section 4 for details) and apply emotion prediction either in a time-continuous fashion or discretely per turn.

In our example implementation in the toolkit, we integrate speech emotion recognition, i.e. using the acoustic signal as features. Based on the work presented in (Neumann and Vu, 2017) we use log Mel filterbank coefficients as input to convolutional neural networks (CNNs). For the sake of

modularity, three separate models are employed for predicting different types of labels: (a) basic emotions {angry, happy, neutral, sad}, (b) arousal levels {low, medium, high}, and (c) valence levels {negative, neutral, positive}. The models are trained on the IEMOCAP dataset (Busso et al., 2008). The output of the emotion recognition module consists of three predictions per user turn, which can then be used by the user state tracker (see section 3.4). For future releases, we plan to incorporate multiple training datasets as well as visual features.

Engagement Level Prediction User engagement is closely related to states such as boredom and level of interest, with implications for user satisfaction and task success (Forbes-Riley et al., 2012; Schuller et al., 2009). In ADVISER, we assume that eye activity serves as an indicator of various mental states (Schuller et al., 2009; Niu et al., 2018) and implement a gaze tracker that monitors the user's direction of focus via webcam.

Using OpenFace 2.2.0, a toolkit for facial behavior analysis (Baltrusaitis et al., 2018), we extract the features *gaze_angle_x* and *gaze_angle_y*, which capture left-right and up-down eye movement, for each frame and compute the deviation from the central point of the screen. If the deviation exceeds a certain threshold for a certain number of seconds, the user is assumed to look away from the screen, thereby disengaging. Thus, the output of our engagement level prediction module is the binary decision {looking, not looking}. Both the spatial and temporal sensitivity can be adjusted, such that developers have the option to decide *how far* and *how long* the user's gaze can stray from the central point until they are considered to be disengaged. In an adaptive system, this information could be used to select re-engagement strategies, e.g. using an affective template (see section 3.7).

Backchanneling In a conversation, a backchannel (BC) is a soft interjection from the listener to the speaker, with the purpose of signaling acknowledgment or reacting to what was just uttered. Backchannels contribute to a successful conversation flow (Clark and Krych, 2004). Therefore, we add an acoustic backchannel module to create a more human-like dialog experience. For backchannel prediction, we extract 13 Mel-frequency-cepstral coefficients from the user's speech signal, which form the input to

the convolutional neural network based on Ortega et al. (2020). The model assigns one of three categories from the proactive backchanneling theory (Goodwin, 1986) to each user utterance {no-backchannel, backchannel-continuer and backchannel-assessment}. The predicted category is used to add the backchannel realization, such as *Right* or *Uh-huh*, to the next system response.

3.2 Speech Processing

Automatic Speech Recognition (ASR) The speech recognition module receives a speech signal as input, which can come from an internal or external microphone, and outputs decoded text. The specific realization of ASR can be interchanged or adapted, for example for new languages or different ASR methods. We provide an end-to-end ASR model for English based on the Transformer neural network architecture. We use the end-to-end speech processing toolkit ESPnet (Watanabe et al., 2018) and the IMS-speech English multi-dataset recipe (Denisov and Vu, 2019), updated to match the LibriSpeech Transformer-based system in ESPnet (Karita et al., 2019) and to include more training data. Training data comprises the LibriSpeech, Switchboard, TED-LIUM 3, AMI, WSJ, Common Voice 3, SWC, VoxForge and M-AILABS datasets with a total amount of 3249 hours. As input features, 80-dimensional log Mel filterbank coefficients are used. Output of the ASR model is a sequence of subword units, which include single characters as well as combinations of several characters, making the model lexicon independent.

Speech Synthesis For ADVISER's voice output, we use the ESPnet-TTS toolkit (Hayashi et al., 2019), which is an extension of the ESPnet toolkit mentioned above. We use FastSpeech as the synthesis model speeding up mel-spectrogram generation by a factor of 270 and voice generation by a factor of 38 compared to autoregressive Transformer TTS (Ren et al., 2019). We use a Parallel WaveGAN (Yamamoto et al., 2020) to generate waveforms that is computationally efficient and achieves a high mean opinion score of 4.16. The FastSpeech and WaveGAN models were trained with 24 hours of the LJSpeech dataset from a single speaker (Ito, 2017) and are capable of generating voice output in real-time when using a GPU. The synthesis can run on any device in a distributed system. Additionally, we optimize the synthesizer for abbreviations, such as *Prof., Univ., IMS, NLP, ECTS* and *PhD*, as well

as for German proper names, such as street names. These optimizations can be easily extended.

Turn Taking   To make interacting with the system more natural, we use a naive end-of-utterance detection. Users indicate the start of their turn by pressing a hotkey, so they can choose to pause the interaction. The highest absolute peak of each recording chunk is then compared with a predefined threshold. If a certain number of sequential chunks do not peak above the threshold, the recording stops. We are currenlty in the process of planning more sophisticated turn taking models, such as Skantze et al. (2015).

3.3   Natural Language Understanding

The natural language understanding (NLU) unit parses the textual user input (De Mori et al., 2008) - or the output from the speech recognition system - and extracts the user action type, generally referred to as intent in goal-oriented dialog systems (e.g. *Inform* and *Request*), as well as the corresponding slots and values. The domain-independent, rule-based NLU presented in Ortega et al. (2019) is integrated into ADVISER and adapted to the new domains presented in section 5.

3.4   State Tracking

Belief State Tracking (BST):   The BST tracks the history of user informs and the user action types, requests, with one BST entry per turn. This information is stored in a dictionary structure that is built up, as the user provides more details and the system has a better understanding of user intent.

User State Tracking (UST):   Similar to the BST, the UST tracks the history of the user's state over the course of a dialog, with one entry per turn. In the current implementation, the user state consists of the user's engagement level, valence, arousal, and emotion category (details in section 3.1).

3.5   Dialog Policies

Policies   To determine the correct system action, we provide three types of policy services: a handcrafted and a reinforcement learning policy for finding entities from a database (Ortega et al., 2019), as well as a handcrafted policy for looking up information through an API call. Both handcrafted policies use a series of rules to help the user find a single entity or, once an entity has been found (or directly provided by the user), find information about that entity. The reinforcement learning (RL)

policy's action-value function is approximated by a neural network which outputs a value for each possible system action, given the vectorized representation of a turn's belief state as input. The neural network is constructed as proposed in Väth and Vu (2019) following a duelling architecture (Wang et al., 2016). It consists of two separate calculation streams, each with its own layers, where the final layer yields the action-value function. For off-policy batch-training, we make use of prioritized experience replay (Schaul et al., 2015).

Affective Policy   In addition, we have also implemented a rule-based affective policy service that can be used to determine the system's emotional response. As this policy is domain-agnostic, predicting the next system *emotion* output rather than the next system *action*, it can be used alongside any of the previously mentioned policies.

User Simulator   To support automatic evaluation and to train the RL policy, we provide a user simulator service outputting at the user acts level. As we are concerned with task-oriented dialogs here, the user simulator has an agenda-based (Schatzmann et al., 2007) architecture and is randomly assigned a goal at the beginning of the dialog. Each turn, it then works to first respond to the system utterance, and then after to fulfill its own goal. When the system utterance also works toward fulfilling the user goal, the RL policy is rewarded by achieving a shorter total dialog turn count (Ortega et al., 2019).

3.6   External Information Resources

ADVISER supports three options to access information from external information sources. In addition to being able to query information from SQL-based databases, we add two new options that includes querying information via APIs and from knowledge bases (e.g. Wikidata (Vrandečić and Krötzsch, 2014)). For example, when a user asks a simple question - *Where was Dirk Nowitzki born?*, our pretrained neural network predicts the topic entity - *Dirk Nowitzki* - and the relation - *place of birth*. Then, the answer is automatically looked up using Wikidata's SPARQL endpoint.

3.7   Natural Language Generation (NLG)

In the NLG service, the semantic representation of the system act is transformed into natural language. ADVISER currently uses a template-based approach to NLG in which each possible system act is mapped to exactly one utterance. A special
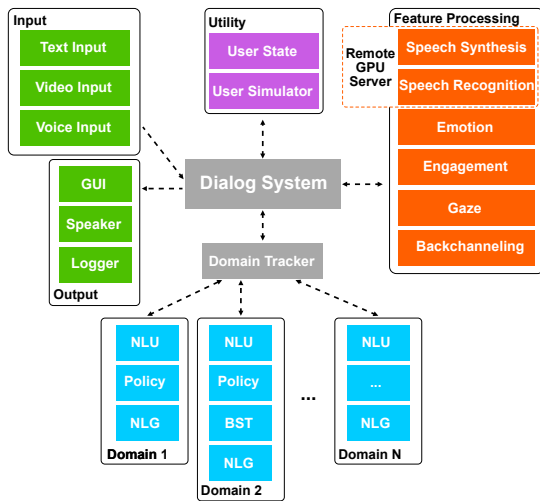
Figure 2: Example ADVISER toolkit configuration: Grey represents backend components, blue represents domain-specific services, and all other colors represent domain-agnostic services. Two components are run remotely.

syntax using placeholders reduces the number of templates needed and accounts for correct morphological inflections (Ortega et al., 2019). Additionally, we developed an affective NLG service, which allows for different templates to be used depending on the user's emotional state. This enables a more sensitive/adaptive system. For example, if the user is sad and the system does not understand the user's input, it might try to establish common ground to prevent their mood from getting worse due to the bad news. An example response would be "As much as I would love to help, I am a bit confused" rather than the more neutral "Sorry I am a bit confused". One set of NLG templates can be specified for each possible emotional state. At runtime, the utterance is then generated from the template associated with the current system emotion and system action.

4 Software Architecture

4.1 Dialog as a Collection of Services

To allow for maximum flexibility in combining and reusing components, we consider a dialog system as a group of services which communicate asynchronously by publishing/subscribing to certain topics. A service is called as soon as at least one message for all its subscribed topics is received and may additionally publish to one or more topics. Services can elect to receive the most recent message for a topic (e.g. up-to-date belief state) or

a list of all messages for that topic since the last service call (e.g. a list of video frames). Constructing a dialog system in this way allows us to break free from a pipeline architecture. Each step in the dialog process is represented by one or more services which can operate in parallel or sequentially. For example, tasks like video and speech capture may be performed and processed in parallel before being synchronized by a user state tracking module subscribing to input from both sources. Figure 2 illustrates the system architecture. For debugging purposes, we provide a utility to draw the dialog graph, showing the information flow between services, including remote services, and any inconsistencies in publish/subscribe connections.

4.2 Support for Distributed Systems

Services are location-transparent and may thus be distributed across multiple machines. A central dialog system discovers local and remote services and provides synchronization guarantees for dialog initialization and termination. Distribution of services enables, for instance, a more powerful computer to handle tasks such as real-time text-to-speech generation (see Figure 2). This is particularly helpful when multiple resource-heavy tasks are combined into a single dialog system.

4.3 Support for Multi-Domain Systems

In addition to providing multi-modal support, the publish/subscribe framework also allows for multi-domain support by providing a structure which enables arbitrary branching and rejoining of graph structures. When a service is created, users simply specify which domain(s) it should publish/subscribe to. This, in combination with a domain tracking service, allows for seamless integration of domain-agnostic services (such as speech input/output) and domain-specific services (such as NLU/NLG for the lecturers domain).

5 Example Use Cases

5.1 Example Domains

We provide several example domains to demonstrate ADVISER's functionalities. Databases for lecturers and courses at the Institute for Natural Language Processing (IMS), which we used in the previous version of ADVISER, were adapted to the new system architecture. As example APIs, we implemented a weather domain that makes calls to

the OpenWeatherMap API[3] and a mensa domain for gathering information from the dining hall at the university of Stuttgart. Note that affective templates were only added to the lecturers and mensa domain. All domains can be used within the same dialog, simply by switching the topic.

5.2 An Example Implementation

Our toolkit allows for easy creation of a dialog system within a few lines of code as follows.

```
# domains
weather = WeatherDomain()
mensa = MensaDomain()
# create system from desired services
ds = DialogSystem(services=[
    DomainTracker(domains=[mensa,
    weather]),
    SpeechRecorder(),
    VideoInput(),
    EngagementTracker(),
    EmotionTracker(),
    WeatherNLU(domain=weather),
    MensaNLU(domain=mensa),
    ...,
    SpeechOutputPlayer()])

ds.run_dialog(...)
```

As a first step, a dialog system object is initialized, which is responsible for coordinating the initialization and graceful termination of all dialog services. Talking about multiple domains in one dialog is enabled by creating a simple keyword-based domain tracker which is introduced as the first argument to the dialog system. To make the dialog multi-modal, speech and vision modules are introduced next, along with modules to extract engagement and emotion. So far, all of these modules are domain-agnostic and can be used as shared resources between all domains. Next, domain-dependent services such as NLUs, BSTs and NLGs for weather and mensa, are added. The following shows an example dialog.

SYSTEM: Hello, please let me know how I can help you, I can discuss the following domains: Mensa Food and Weather.
USER: I could have something to eat. What does the mensa offer today?
SYSTEM: What type of dish are you looking for?
USER: I would like a main dish.
SYSTEM: Should the meal be vegan?
USER: Yes.
SYSTEM: The meal mediterranean Ebly wheat is served today, is a main dish and is vegan.
USER: Okay, cool, I will go there now! What is the weather like?

---

[3] https://openweathermap.org/api

SYSTEM: The weather in Stuttgart on January 28 at 3 PM is 3 degrees celsius with light snow.
USER: Thank you, ADVISER, good bye!
SYSTEM: Thank you, good bye.

6 Related Work

Other tools for building dialog systems include *ConvLab* (Lee et al., 2019), an open-source, text-based dialog system platform that supports both pipelined architectures and an end-to-end neural architecture. *ConvLab* also provides reusable components and supports multi-domain settings. Other systems are largely text-based, but offer the incorporation of external speech components. *InproTK* (Baumann and Schlangen, 2012), for instance, in which modules communicate by networks via configuration files, uses ASR based on Sphinx-4 and synthesis based on MaryTTS. Similarly, *RavenClaw* (Bohus and Rudnicky, 2009) provides a framework for creating dialog managers; ASR and synthesis components can be supplied, for example, by connecting to Sphinx and Kalliope. *OpenDial* (Lison and Kennington, 2016) relies on probabilistic rules and provides options to connect to speech components such as Sphinx. Multi-domain dialog toolkit - *PyDial* (Ultes et al., 2017) supports connection to DialPort.

As mentioned in the introduction, Microsoft Research's \psi is an open and extensible platform that supports the development of multi-modal AI systems (Bohus et al., 2017). It further offers audio and visual processing, such as speech recognition and face tracking, as well as output, such as synthesis and avatar rendering. And the *MuMMER* (multi-modal Mall Entertainment Robot) project (Foster et al., 2016) is based on the SoftBank Robotics *Pepper* platform, and thereby comprises processing of audio-, visual- and social signals, with the aim to develop a socially engaging robot that can be deployed in public spaces.

7 Conclusions

We introduce ADVISER – an open-source, multi-domain dialog system toolkit that allows users to easily develop multi-modal and socially-engaged conversational agents. We provide a large variety of functionalities, ranging from speech processing to core dialog system capabilities and social signal processing. With this toolkit, we hope to provide a flexible platform for collaborative research in multi-domain, multi-modal, socially-engaged conversational agents.

References

Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. 2018. OpenFace 2.0: Facial Behavior Analysis Toolkit.

Timo Baumann and David Schlangen. 2012. The InproTK 2012 Release. In *NAACL-HLT Workshop on Future Directions and Needs in the Spoken Dialog Community: Tools and Data*.

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry Thompson, and Terry Winograd. 1977. Gus, a frame-driven dialog system. *Artificial Intelligence*.

Dan Bohus, Sean Andrist, and Mihai Jalobeanu. 2017. Rapid Development of Multimodal Interactive Systems: A Demonstration of Platform for Situated Intelligence. In *ICMI '17: Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pages 493–494.

Dan Bohus and Alexander I Rudnicky. 2009. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4).

H. H Clark and M. A Krych. 2004. Speaking while monitoring addressees for understanding. *Journal of Memory and Language*.

Amanda Cercas Curry, Ioannis Papaioannou, Alessandro Suglia, Shubham Agarwal, Igor Shalyminov, Xinnuo Xu, Ondřej Dušek, Arash Eshghi, Ioannis Konstas, Verena Rieser, and Oliver Lemon. 2018. Alana v2: Entertaining and Informative Opendomain Social Dialogue using Ontologies and Entity Linking. In *1st Proceedings of Alexa Prize*.

Renato De Mori, Frédéric Bechet, Dilek Hakkani-Tur, Michael McTear, Giuseppe Riccardi, and Gokhan Tur. 2008. Spoken language understanding. *IEEE Signal Processing Magazine*.

Pavel Denisov and Ngoc Thang Vu. 2019. Imsspeech: A speech to text tool. *Studientexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2019*, pages 170–177.

Kate Forbes-Riley, Diane Litman, Heather Friedberg, and Joanna Drummond. 2012. Intrinsic and Extrinsic Evaluation of an Automatic User Disengagement Detector for an Uncertainty-Adaptive Spoken Dialogue System. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 91–102, Montréal.

Mary Ellen Foster, Rachid Alami, Olli Gestranius, Oliver Lemon, Marketta Niemelä, Jean Marc Odobez, and Amit Kumar Pandey. 2016. The MuMMER project: Engaging human-robot interaction in real-world public spaces. In *Processings of the Eighth International Conference on Social Robotics (ICSR 2016)*, pages 753–763. Springer.

Charles Goodwin. 1986. Between and within: Alternative sequential treatments of continuers and assessments. *Journal of Human Studies*.

Tomoki Hayashi, Ryuichi Yamamoto, Katsuki Inoue, Takenori Yoshimura, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Yu Zhang, and Xu Tan. 2019. Espnet-tts: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit.

Keith Ito. 2017. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. 2019. A comparative study on transformer vs rnn in speech applications. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

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 Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69.

Pierre Lison and Casey Kennington. 2016. Opendial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of ACL-2016 system demonstrations*, pages 67–72.

Michael Frederick McTear, Zoraida Callejas, and David Griol. 2016. *The conversational interface*, volume 6. Springer.

Michael Neumann and Ngoc Thang Vu. 2017. Attentive convolutional neural network based speech emotion recognition: A study on the impact of input features, signal length, and acted speech. In *Proceedings of Interspeech*.

Xuesong Niu, Hu Han, Jiabei Zeng, Xuran Sun, Shiguang Shan, Yan Huang, Songfan Yang, and Xilin Chen. 2018. Automatic Engagement Prediction with GAP Feature. In *ICMI*, pages 599–603, Boulder. Association for Computing Machinery.

Daniel Ortega, Chia-Yu Li, and Thang Vu. 2020. Oh, Jeez! or uh-huh? A listener-aware Backchannel predictor on ASR transcriptions. In *ICASSP 2020 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8064–8068.

Daniel Ortega, Dirk Väth, Gianna Weber, Lindsey Vanderlyn, Maximilian Schmidt, Moritz Völkel, Zorica Karacevic, and Ngoc Thang Vu. 2019. Adviser: A dialog system framework for education & research. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–98.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*, pages 3165–3174.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *Proceedings of NAACL*.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. In *Proceedings of ICLR*.

Björn Schuller, Ronald Müller, Florian Eyben, Jürgen Gast, Benedikt Hörnler, Martin Wöllmer, Gerhard Rigoll, Anja Höthker, and Hitoshi Konosu. 2009. Being bored? Recognising natural interest by extensive audiovisual integration for real-life application. *Image and Vision Computing*, 27:1760–1774.

Gabriel Skantze, Martin Johansson, and Jonas Beskow. 2015. Exploring turn-taking cues in multi-party human-robot discussions about objects. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 67–74.

Stefan Ultes, Lina M. Rojas Barahona, Pei-Hao Su, David Vandyke, Dongho Kim, Iñigo Casanueva, Paweł Budzianowski, Nikola Mrkšić, Tsung-Hsien Wen, Milica Gasic, and Steve Young. 2017. PyDial: A Multi-domain Statistical Dialogue System Toolkit. In *Proceedings of ACL*.

Dirk Väth and Ngoc Thang Vu. 2019. To combine or not to combine? a rainbow deep reinforcement learning agent for dialog policies. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 62–67.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Johannes Wagner, Florian Lingenfelser, Tobias Baur, Ionut Damian, Felix Kistler, and Elisabeth André. 2013. The social signal interpretation (ssi) framework: multimodal signal processing and recognition in real-time. In *Proceedings of the 21st ACM international conference on Multimedia*.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *Proceedings of ICML*.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. Espnet: End-to-end speech processing toolkit. In *Interspeech*, pages 2207–2211.

Joseph Weizenbaum. 1966. ELIZA: A Computer Program for the Study of Natural Language Communication Between Man and Machine. *Communications of the ACM*, 9(1).

Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. 2020. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6199–6203.

Zhihong Zeng, Maja Pantic, Glenn I Roisman, and Thomas S Huang. 2009. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE transactions on pattern analysis and machine intelligence*, 31(1):39–58.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2018. The Design and Implementation of XiaoIce, an Empathetic Social Chatbot. *Computational Linguistics*, pages 1–62.