End-to-End Multimodal Learning for Situated Dialogue Systems

Submitted in partial fulfillment of the requirements for
the degree of

Doctor of Philsophy
in
Department of Electrical and Computer Engineering

Guan-Lin Chao

B.Sc Electrical Engineering, National Taiwan University

Carnegie Mellon University
Pittsburgh, PA

August, 2021

ACKNOWLEDGEMENT

First, I'd like to give thanks to my advisors Prof. Ian Lane and Prof. John Shen from CMU-ECE for in depth research guidance, funding support, and career advice. I'm also grateful for my thesis committee, Prof. Alexander Rudnicky from CMU-LTI, Dr. Dilek Hakkani-Tür from Amazon Alexa AI and Dr. William Chan from Google Brain for the much constructive feedback in my thesis research directions and writing. I would like to acknowledge the financial support from various organizations including the College of Engineering at Carnegie Mellon University, Ford Motor Company and Sense of Wonder Group. I was very fortunate to learn from and discuss research with many professors and PhD students in the ECE community, William Chan, David Cohen, Akshay Chandrashekaran, Bing Liu, Wonkyum Lee, Nagasrikanth Kallakuri, Suyoun Kim, Benjamin Elizalde, Naoki Tsuda, Roshan Sharma, Muqiao Yang and Yifan Peng from Ian's group and Chih Chi Hu from John's group, as well as Ming Zeng, Aniruddha Basak, Tong Yu, Yong Zhuang, Yingrui Zhang, Rashad Eletreby, Prof. Pei Zhang, Prof. Hae Young Noh, Shijia Pan, Frank Mokaya, Xinlei Chen, Carlos Ruiz Dominguez, Adeola Bannis, Amelie Bonde, Prof. Patrick Tague, Thanh Le Nguyen, Yuan Tian, Jun Han, Xiao Wang, Madhumitha Harishankar, Prof. Carlee Joe-Wong, Prof. Richard Stern, Prof. Bhiksha Raj, Yang Gao, and Wenbo Zhao. During my summer internships at Google, I learned a lot from my mentors and collaborators: Heng-Tze Cheng, Jindong Chen, Abhinav Rastogi, Dilek Hakkani-Tür and Semih Yavuz. In my six years residing in the Bay Area, I also enjoyed the fellowship with many brothers and sisters from the Canaan Taiwanese Christian Church and the Grace Plus fellowship. I'm also very grateful to my parents and younger brother Ching-Ming Chao, Huey-Mann Sheu and Chih-Lin Chao who consistently send their love and support from Taiwan. Finally biggest thanks to my dear husband Sander Valstar who moved to the US for me. It is the fondest memory that we love and support each other throughout "our" PhD years.

ABSTRACT

Virtual assistants have become an essential part in many people's lives today. These dialogue systems perform services given users' voice commands, such as controlling devices, searching for information, or performing conversational tasks such as booking of events or navigation instruction. However, today's dialogue systems face challenges, because (1) they are implemented using a pipeline of multiple independently optimized modules which do not necessarily provide the best performance when integrated together and (2) they are limited to utilizing only unimodal input, i.e. speech input from the user. The modularized system design induces a disconnect between each module's and the quality of the overall dialogue system, and it also makes it difficult to update the entire system for a new task as every module will need to be changed. While the multimodal context contains rich information of the users and their surrounding environments, many dialogue systems in today's virtual assistants interact with the users utilizing only their language input via a speech interface. As dialogue systems only utilize speech input, they are unable to provide services which require understanding the user or environmental context, for example conversing with a user regarding their physical surroundings.

In this thesis, we mitigate the limitations of prior dialogue systems in two ways: (1) we propose an end-to-end model which fuses the separate components in a standard spoken dialogue system together and (2) we leverage multimoal contextual cues from the user and the environment, to enable a dialogue system that can interact with the user based on their physical surroundings. We introduce end-to-end learning for scalable dialogue state tracking, where the model directly predicts dialogue states from natural language input and can handle unseen slot values. We enhance our speech recognition system using multimodal input with the target speaker's mouth movements and learned speaker embedding to improve robustness in noisy cocktail party environments. Finally, we apply end-to-end and multimodal learning on two situated dialogue tasks: vision-grounded instruction following and video question answering. The situated dialogue model directly takes as input the multimodal language and visual context from the user and the environment, and outputs system actions or natural language responses. Compared to prior methods, our proposed situated dialogue systems showed improved speech recognition accuracy, dialog state tracking accuracy, task success rate and response generation quality.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation and Research Problem

Virtual assistants, such as Amazon Alexa [153] and Google Assistant [154], are becoming increasingly intelligent and prevalent in our lives [94, 30, 84]. Instructed via the speech interface, they are able to perform tasks of various complexities, from setting up a reminder, answering factoid questions, to buying a product online and making a restaurant reservation.

Conventional spoken dialogue systems are generally a pipeline of the following modules: (1) speech recognition interface to transcribe user's speech into text, (2) spoken language understanding to extract task-related information from a user response, (3) dialogue state tracking to track the user's goals in the course of multiple dialogue turns, (4) dialogue policy to decide the dialogue system's action, (5) response generation to convert the dialogue action to a natural language response, and (6) text-to-speech engine to convert the system response into auditory form.

There are several limitations coming from the modularized design of the conventional systems:

1. Because these modules are trained independently with their own objectives, instead of

towards the final output quality (i.e., quality of system response), there is a disconnect between the performance of the intermediate modules and the quality of the overall system.

2. In the production system, when one module is updated, all of its subsequent modules need to be re-trained with the updated inputs to ensure the performance during inference. This reduces the flexibility for intermediate module update.

Besides the limitations from the complex modularized architecture, another challenge that the conventional dialogue systems face is the scenario of noisy background with overlapping speech. This results in limited speech recognition accuracy in the dialogue systems, and the speech recognition errors can propagate to the downstream modules. Furthermore, using only the user's speech as input, conventional spoken dialogue systems are usually limited to perform software tasks, or at most triggering pre-programmed smart hardware devices. Several multimodal dialogue applications can not be realized without taking into account the rich multimodal context information of the user and their surrounding environment. For example, in visual question answering, a user can have a conversation with the system regarding what happens in their physical world. In vision-grounded instruction following, a user can ask the agent to perform an action in the physical space. Both of these applications require the system to have visual perception of the environment.

We can use multimodal context from the user and the environment to mitigate these limitations. To handle the problem of noisy speech input, we can use the visual cues from the user to improve speech recognition, because visual cures are invariant to acoustic interference. We can also incorporate the contextual visual cues from the environment for the multimodal situated dialogue tasks.

1.2 Thesis Outline

In this thesis, we aim to improve the dialogue systems via (1) learning an end-to-end model which fuses separate modules in the pipeline to directly optimize the objective of interest, and (2) leveraging multimodal context from the user and the environment for situated dialogue interaction.

In Chapter 2, we review the related literature and state-of-the-art approaches on the situated dialogue tasks.

In Chapter 3, we study end-to-end dialogue state tracking. Without a separate language understanding module, the dialogue state tracker directly predicts the dialogue states from natural language dialogue text input. Based on the state-of-the-art pretrained BERT language model, our state tracker does not require a predefined ontology for slot values, and thus is scalable for unseen slot values.

In Chapter 4, we extend the conventional audio-only Speech Recognizer with the multimodal context from the user, in order to address speech recognition in adverse acoustic environments where background speech signals and noise drastically deteriorate recognition accuracy. We propose to use the visual information and machine learned speaker embedding from the speech to adapt the acoustic models. They provide correlated and supplementary information on top of the acoustic signal and are robust against environmental acoustic noise.

In Chapter 5, we explore policy learning strategies for vision-grounded instruction following agents. The agent policy is an end-to-end model that takes the dialogue text from the human and contextual visual cues from the environment as input and predicts an action. The agent's goal is to understand the natural language instruction given by the user, execute a series of actions to navigate the indoor environment based on its visual perception and eventually complete the instruction. We experiment with curriculum learning and human-agent dialogue to boost the agent's policy training efficiency.

In Chapter 6, we consider response generation for video question answering (VideoQA). The end-to-end VideoQA model receives multimodal inputs: a textual question from the user and a video. The goal is to generate natural language answers to the question grounded on the video content. We propose a question-guided video feature extraction mechanism to extract, summarize and filter the video features based on the question to provide more relevant information for answer generation.

In Chapter 7, we conclude the thesis and discuss future research directions.

Chapter 2

Background

2.1 Scalable Dialogue State Tracking

Dialogue state tracking (DST), a core component in today's task-oriented dialogue systems, maintains user's intentional states through the course of a dialogue. The dialogue states predicted by DST are used by the downstream dialogue management component to produce API calls to a backend database and generate responses to the user [20]. A dialogue state is often expressed as a collection of slot-value pairs. The set of slots and their possible values are often domain-specific, defined in a *domain ontology.* Many state-of-the-art approaches operate on a fixed ontology, by performing classification over a predefined set of slot values or iteratively scoring slot-value pairs from the ontology [92, 134, 127]. However, such models can be inefficient or infeasible when the ontology is dynamic (*e.g.*, *movie, restaurant*), innumerable (*e.g.*, *time*), or simply not exposed by an external database [96, 131].

In Chapter 3 of this thesis, we study one practical problem in DST – scalability with unknown ontology and unseen slot values, with a specific condition: the target slot value (except for *none* and *dontcare*) always appears as word segment in the dialogue context. Previous approaches often require a *candidate list*, which can be an exhaustive list of n-grams in the dialogue context or slot tagging outputs from a separate language understanding (LU) module [96, 126, 118]. Using n-gram candidate generation might be inefficient because the number of candidates the DST scorer needs to iterate through is proportional to the length of the dialogue context. Although the LU-generated candidate list can be shorter, the DST scorer cannot recover from missing target candidates incurred by LU errors [96, 131].

2.2 Bidirectional Encoder Representations from Transformer (BERT)

BERT [140] is a multi-layer bidirectional Transformer encoder [103], which is a stack of multiple identical layers each containing a multi-head self-attention and a fully-connected sub-layer with residual connections [61]. The input to BERT is a sequence of tokens, which can be concatenation of a pair of sentences. The input sequence is prepended by a special `[CLS]` token whose final hidden state is used as the aggregate sequence representation. The final hidden states of the other tokens are used as token-level representations. Besides word embedding and positional embedding used in the original Transformer model, BERT's input layer adds an additional segment embedding to differentiate tokens from the pair of sentences.

To learn bidirectional contextualized representations and inter-sentence relationship, BERT model is pre-trained on two unsupervised language modeling tasks: masked language modeling [1] and next sentence prediction, using the BooksCorpus [54] and the English Wikipedia corpora. The procedures of language model pre-training are detailed in [140]. With extra projection layers and fine-tuning the deep structure, BERT has been successfully applied to various tasks such as reading comprehension, named entity recognition, sentiment analysis, *etc.*

2.3 Robust Speech Recognition

Speech recognition is an essential front-end of a dialogue system, where the speech of the user of interest needs to be extracted and transcribed from the acoustic signal input. Robust Speech Recognition in cocktail-party environments aims to recognize the speech of an individual speaker from a background containing many concurrent voices, and has attracted researchers for decades [5, 18]. Current ASR systems can decode clear speech well in relatively noiseless environments. However, in a cocktail-party environment, their performance is severely degraded in the presence of loud noise or interfering speech signals, especially when the acoustic signal of the speaker of interest and the background share similar frequency and temporal characteristics [7].

Some previous approaches to this problem can be: *multimodal robust features* and *blind signal separation*, or a hybrid of both. In ASR systems, it is common to adapt a well-trained, general acoustic model to new users or environmental conditions. [29] proposed to supply speaker identity vectors, *i-vectors* as input features to a deep neural network (DNN) along with acoustic features. [33] extended [29] by factorizing i-vectors to represent speaker as well as acoustic environment. [4] trained speaker-specific parameters jointly with acoustic features in an adaptive DNN-hidden Markov model (DNN-HMM) for word recognition. [25, 38, 31] proposed training speaker-specific discriminant features (referred to as *speaker codes* and *bottleneck features*) for fast DNN-HMM speaker adaptation in speech recognition. [26] extended the speaker codes approach to convolutional neural network-HMM (CNN-HMM) systems. [42] investigated different NN architectures of learning i-vectors for input feature mapping.

Audio-Visual Speech Recognition

Inspired by humans' ability to use other sensory information like visual cues and knowledge about the environment to recognize speech, research in audio-visual ASR has also demonstrated the advantage of using audio-visual features over audio-only features in robust speech recognition. The McGurk effect was introduced in [2], which illustrates that visual information can affect human's interpretation of audio signals. In [5], low dimensional lip movement vectors, *eigenlips*, were used to complement acoustic features for ASR. In [9], generalized versions of HMMs, factorial HMM and the coupled HMM, were used to fuse auditory and visual information, in which the HMM parameters were able to be trained with dynamic Bayesian networks. In [23], the authors proposed a DNN-based approach to learning multimodal features and a shared representation between modalities. In [43], the authors presented a deep neural network that used a bilinear softmax layer to account for class specific correlations between modalities. In [44], a deep learning architecture with multi-stream HMM model was proposed. Using noise-robust acoustic features extracted by autoencoders and mouth region of interest (ROI) image features extracted by CNNs, this approach achieved higher word recognition rate than the use of non-denoised features or normal HMMs. [56] proposed an active appearance model-based approach to extracting visual features of jaw and lip ROI on four image streams which were then combined with acoustic features for in-car audio-visual ASR.

Blind Signal Separation

Traditional cocktail-party ASR methods suggest performing blind signal separation prior to auditory speech recognition of individual signals. Blind signal separation aims at estimating multiple unknown sources from the sensor signals. When there is only a single-channel signal available, source separation on the cocktail-party problem becomes even more difficult [14]. A main assumption in the signal separation is that speech signals from different sources are statistically independent [7]. Another common assumption of signal separation is that all the sources have zero-mean and unit variance for the convenience of performing Independent Component Analysis [6, 8]. However, these two assumptions are not always correct in practice. Therefore, we try to lift these assumptions by directly recognizing single-channel signals of overlapping speech in this work.

2.4 Speaker Embedding Models

Speaker embedding models are widely used as the front-end of speaker verification and identification systems. The models are trained to map variable length speech utterances into a feature space. And the back-end classifiers use the projected embedding as input features to make decisions. Traditionally, Gaussian Mixture Model-Universal Background Models (GMM-UBMs) are used to extract i-vectors [22], which are useful speaker embedding to model speaker and channel variability at the same time. i-vectors are generated by computing and optimizing sufficient statistics to fit the acoustic features of speech utterances. There have been many previous approaches to replacing the GMM-UBM with Deep Neural Networks (DNNs) for speaker embedding extraction [80, 114, 115, 88, 90, 64, 125, 70, 99, 102, 108, 132, 110]. Speaker embedding is also useful for adapting acoustic models in speech recognition [39, 80, 114, 88, 64, 125, 102, 110]. [64] and [29] adapt DNN acoustic models to a target speaker by concatenating the input acoustic features with GMM-UBM-based and Hidden Markov Model-based i-vectors respectively. In [62], DNN acoustic models are adapted to noisy and reverberant acoustic environments for robust speech recognition, by concatenating acoustic features with environment embeddings, which are bottleneck features extracted from an environment recognition DNN. However, the aforementioned methods require the embedding estimation to be performed on the fly. In other words, after an utterance is received, the speech recognizer has to wait until the embedding is extracted from the current input signal, before transcribing starts.

2.5 Instruction Following

Instructions following has attracted extensive research attention. [17] trains a RL agent to map natural language instructions in troubleshooting manuals and game tutorials to sequence of actions. [19] trains a RL agent to interpret instructions to draw a path on a map from natural language dialogue input.

In recent years, more research focuses on learning to follow instructions which are grounded or embodied in the physical world. Many simulation platforms are introduced for training and testing vision-grounded instruction following agents. For example, MiniGrid [113] and BabyAI [139] offer 2D maze-like environments, and DeepMind Lab [55] is a 3D environment simulator. [82] trains a RL agent to follow multi-word instructions, including moving around and manipulating objects in a multi-room 3D environment based on DeepMind Lab using curriculum learning and multi-task learning techniques.

Compared with the Vision-and-Language Navigation task [112, 116], such as Room-to-Room dataset [79], in which visual observations are real building scenes, instruction-following agents deal with simulated and simplistic visual input. On the other hand, the simulation environments provide high flexibility and randomness to create complex room layouts and agent and object settings. The highly diversified training data can help the agent generalize to unseen room environments.

2.6 Visual Question Answering

In recent years, research on visual question answering has accelerated following the release of multiple publicly available datasets. These datasets include COCO-QA [46], VQA [77], and Visual Madlibs [52] for image question answering and MovieQA [71], TGIF-QA [83], and TVQA [120] for video question answering.

Image Question Answering

The goal of image question answering is to infer the correct answer, given a natural language question related to the visual content of an image. It assesses the system's capability of multimodal understanding and reasoning regarding multiple aspects of humans and objects, such as their appearance, counting, relationships and interactions [120]. State-of-the-art image question answering models make use of spatial attention to obtain a fixed length question-dependent embedded representation of the image, which is then combined with the question feature to predict the answer [75, 74, 86, 111]. Dynamic memory [63, 73] and co-attention mechanism [65, 121] are also adopted to model sophisticated cross-modality interactions.

Video Question Answering

VideoQA is a more complex task. As a video is a sequence of images, it contains not only appearance information but also motion and transitions. Therefore, VideoQA requires spatial and temporal aggregation of image features to encode the video into a question-relevant representation. Hence, temporal frame-level attention is utilized to model the temporal dynamics, where frame-level attribute detection and unified video representation are learned jointly [107, 106, 93]. Similarly, [120] uses Faster R-CNN [47] trained with the Visual Genome [89] dataset to detect object and attribute regions in each frame, which are used as input features to the question answering model. Previous works also adopt various forms of external memory [49, 63, 59] to store question information, which allows multiple iterations of question-conditioned inference on the video features [95, 87, 109, 117, 141].

Video Question Answering Dialogue

Recently in DSTC7, [135] introduces the Audio-Visual Scene-aware Dialog (AVSD) dataset for multi-turn VideoQA. In addition to the challenge of integrating the questions and the dynamic scene information, the dialogue system also needs to effectively incorporate the dialogue context for coreference resolution to fully understand the user's questions across turns. To this end, [136] uses two-stream inflated 3D ConvNet (I3D) model [78] to extract spatiotemporal visual frame features (I3D-RGB features for RGB input and I3D-flow features for optical flow input), and propose the Naïve Fusion method to combine multimodal inputs based on the hierarchical recurrent encoder (HRE) architecture [81]. [119] extends the Naïve Fusion approach and propose the Attentional Fusion method which learns multimodal attention weights to fuse features from different modalities. [151] modify the

Attentional Fusion method and propose to use Maximum Mutual Information (MMI) [3] as the training objective. Besides the HRE architecture, the multi-source sequence-to-sequence (Multi-Source Seq2Seq) architecture with attention [76, 58] is also commonly applied [148, 143, 150]. Previous works [149, 144, 148] also explore various attention mechanisms to incorporate the different modal inputs, such as hierarchical attention [91] and cross attention [97]. For modeling visual features, [145] proposes to use Dynamic memory networks [63] and [147] proposes to use feature-wise linear modulation layers [124].

Chapter 3

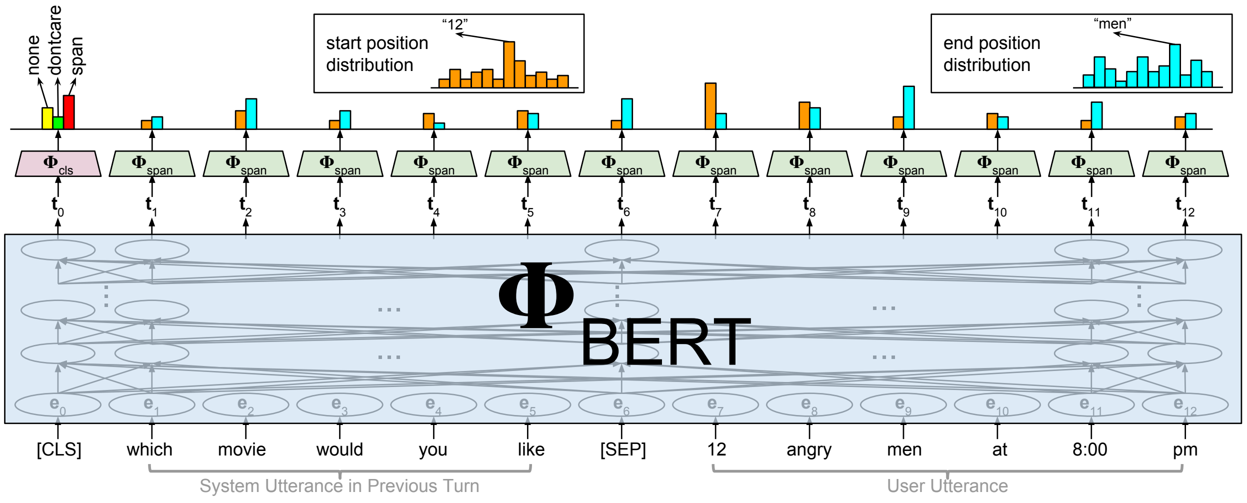
End-to-End Dialogue State Tracking

We propose BERT-DST, an end-to-end dialogue state tracker based on the pretrained BERT language model. It directly predicts dialogue states from natural language utterances input, without using a separate language understanding model.

Our proposed application of BERT to scalable DST is in spirit similar to the Stanford Question Answering Dataset (SQuAD) task [67]. In SQuAD, the input is a question and a reading passage. If the reading paragraph contains the answer to the question, the output is a segment of text from the paragraph, represented by its span (start and end positions). Otherwise, the model should output *unanswerable*. Similarly, in our targeted case of scalable DST, a slot's value can be *none*, *dontcare*, or a word segment from the dialogue context. Our proposed framework uses BERT's contextualized sentence-level and token-level representations to determine the type of slot value (*none*, *dontcare*, or *span*), and the span of the specified slot value from the dialogue context. Using BERT as dialogue context encoder provides the following advantages. The contextualized word representations are suitable for extracting slot values from contextual patterns. With large-scale language model pretraining, BERT's word representations are good initialization to be fine-tuned to our DST problem.

3.1 Model

The BERT-DST model architecture is shown in Figure 3.1. For each user turn, the model takes the recent dialogue context as input and outputs the turn-level dialogue state. First, the dialogue context input is encoded by the BERT-based encoding module to produce contextualized sentence-level and token-level representations. The sentence-level representation is then used by the classification module to generate a categorical distribution over three types of slot values: *none*, *dontcare* or a *span* from the input. The span prediction module gathers the token-level representations and outputs the slot value's start and end positions. Finally, an update mechanism is used to track dialogue states across turns.



none
dontcare
span
start position distribution
"12"
end position distribution
"men"
Φ_{cls}
Φ_{span}
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12}
Φ_{BERT}
e_0 e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12}
[CLS] which movie would you like [SEP] 12 angry men at 8:00 pm
System Utterance in Previous Turn
User Utterance

Figure 3.1: Architecture of the proposed BERT-DST framework. The diagram is color-coded such that modules with the same color share the same parameters. For each user turn, BERT-DST takes as input the recent dialogue context (system utterance in previous turn and the user utterance), and outputs turn-level dialogue state. BERT dialogue context encoding module Φ_{BERT} (blue) produces contextualized sentence-level and token-level representations of the dialogue context. The per-slot classification module Φ_{cls} (red) uses the sentence-level representation to generate a categorical distribution over three types of slot values {*none*, *dontcare* and *span*}. The per-slot span prediction module Φ_{span} (green) gathers the token-level representations and output the start and end positions (span) of the slot value. Note that the dialogue context encoding module Φ_{BERT} allows parameter sharing across all slots.

Dialogue Context Encoding

The dialogue context encoding module is based on BERT. We use the the previous turn's system utterance and the current turn's user utterance as dialogue context input, and pass it to BERT's bidirectional Transformer encoder, which outputs a sentence-level and token-level representation of the dialogue context input.

The first token is `[CLS]`, followed by the tokenized system utterance, `[SEP]`, and tokenized user utterance. Let $[x_0, x_1, \cdots, x_n]$ denote the input token sequence. BERT's input layer embeds each token x_i into an embedding $\mathbf{e}_i$. , which is the sum of three embeddings:

$$\begin{aligned} \text{BERTinput}(x_i) &= E_{\text{tok}}(x_i) + E_{\text{seg}}(i) + E_{\text{pos}}(i) \\ &= \mathbf{e}_i \in \mathbb{R}^d, \quad \forall\ 0 \le i \le n \end{aligned} \tag{3.1}$$

where $E_{\text{tok}}(x_i)$ is WordPiece embedding [72] for token x_i, $E_{\text{seg}}(i) \in \{\mathbf{e}_{\text{first}}, \mathbf{e}_{\text{second}}\}$ is segment embedding whose value is determined by whether the token belongs to the first or second sentence, and $E_{\text{pos}}(i)$ is positional embedding [130] for the i-th token.

The embedded input sequence $[\mathbf{e}_0, \cdots, \mathbf{e}_n]$ is then passed to BERT's bidirectional Transformer encoder, whose final hidden states are denoted by $[\mathbf{t}_0, \cdots, \mathbf{t}_n]$.

$$\begin{aligned} &[\mathbf{t}_0, \cdots, \mathbf{t}_n] = \text{BiTransformer}([\mathbf{e}_0, \cdots, \mathbf{e}_n]) \\ &\mathbf{t}_i \in \mathbb{R}^d, \quad \forall\ 0 \le i \le n \end{aligned} \tag{3.2}$$

The contextualized sentence-level representation $\mathbf{t}_0$, *i.e.*, the final state corresponding to the `[CLS]` token, is passed to the classification module. The contextualized token-level representations $[\mathbf{t}_1, \cdots, \mathbf{t}_n]$ are used by the span prediction module.

The parameters in the dialogue context encoding module, denoted by Φ_{BERT}, are initialized from a pre-trained BERT checkpoint and then fine-tuned on our DST dataset.

Slot State Classification

The classification module's input is the sentence-level representation $\mathbf{t}_0$ from the dialogue context encoding module. For each slot $s \in S$ in the collection of all informable slots, S the classification module's prediction the value of s is one of the three classes {*none*, *dontcare*, *span*}.

$$\mathbf{a}^s = \mathbf{W}^s_{cls}\mathbf{t}_0 + \mathbf{b}^s_{\text{cls}} = [a^s_{\text{none}}, a^s_{\text{dontcare}}, a^s_{\text{span}}] \in \mathbb{R}^3 \tag{3.3}$$

$$\mathbf{p}^s = \text{softmax}(\mathbf{a}^s) = [p^s_{\text{none}}, p^s_{\text{dontcare}}, p^s_{\text{span}}] \tag{3.4}$$

$$\text{slot\_value}^s = \text{argmax}_{c \in \{\text{none, dontcare, span}\}}(p^s_c) \tag{3.5}$$

The per-slot classification parameters, denoted by $\Phi^s_{\text{cls}} = \{\mathbf{W}^s_{\text{cls}}, \mathbf{b}_{\text{cls}}\}$, are trained from scratch on our DST dataset.

Slot Value Span Prediction

For each informable slot, $s \in S$ the span prediction module takes as input the token-level representations $[\mathbf{t}_1, \cdots, \mathbf{t}_n]$ from the dialogue context encoding module. Each token representation $\mathbf{t}_i$ is linearly projected through a common layer whose output values α^s_i and β^s_i correspond to start and end positions respectively. Softmax is then applied to the position values to produce a probability distribution over all tokens, by which the slot value span (start and end positions) of the slot can be determined.

$$[\alpha^s_i, \beta^s_i] = \mathbf{W}^s_{\text{span}}\mathbf{t}_i + \mathbf{b}^s_{\text{span}} \in \mathbb{R}^2, \forall\ 1 \leq i \leq n \tag{3.6}$$

$$\mathbf{p}^s_\alpha = \text{softmax}(\alpha^s) \tag{3.7}$$

$$\mathbf{p}^s_\beta = \text{softmax}(\beta^s) \tag{3.8}$$

$$\text{start\_pos}^s = \text{argmax}_i(p^s_{\alpha,i}) \tag{3.9}$$

$$\text{end\_pos}^s = \text{argmax}_i(p^s_{\beta,i}) \tag{3.10}$$

The per-slot span prediction parameters, denoted by $\Phi^s_{\text{span}} = \{\mathbf{W}^s_{\text{span}}, \mathbf{b}^s_{\text{span}}\}$, are trained from scratch on our DST dataset.

Dialogue State Update Mechanism

To track dialogue states across turns, we employ a rule-based update mechanism. In each turn, if the model's turn prediction for a slot is *dontcare* or a specified value (*i.e.*, any value other than *none*), it will be used to update the dialogue state. Otherwise, the dialogue state of the slot remains the same as the previous turn.

Parameter Sharing

Although our classification and span prediction modules are slot-specific, we notice that the contextualized representations generated by the dialogue context encoding module can be shared among slots; *i.e.*, we can apply parameter sharing in the dialogue context encoding module across all slots. Sharing dialogue context encoder parameters Φ_{BERT} across all slots not only drastically reduces the number of model parameters. It also allows knowledge transfer among slots, which may potentially benefit contextual relation understanding. In the following sections, we call the joint architecture of slot-specific BERT-DST models as **BERT-DST_SS** and the BERT-DST model with encoding module parameter sharing as **BERT-DST_PS**.

Slot Value Dropout

Slot value dropout, or targeted feature dropout, was originally proposed to address the under-training problem of contextual features in slot-filling [37, 131]. The problem happens when models tend to overfit to frequent slot values in training data instead of learning contextual patterns, which adversely harms the performance on Out-of-Vocabulary (OOV) slot values. To improve the robustness for unseen slot values, in the training phase, we replace each of the target slot value tokens by a special `[UNK]` token at a certain probability.

3.2 Experiments

We evaluate our models using *joint goal accuracy* [32], a standard metric for DST. The model's prediction has to jointly match all the informable slot labels to be considered correct.

Datasets

Table 3.1: Statistics of Sim-M, Sim-R, DSTC2 and WOZ 2.0 datasets. The number of dialogues is given for train, dev and test sets respectively. The slots containing OOV values are marked in bold. Parentheses represent
(# unique OOV values in dev set / # unique values in dev set;
# unique OOV values in test set / # unique values in test set).

Datasets	# Dialogues			Slots
	Train	Validation	Test	
Sim-M	384	120	264	date, time, num_tickets, theatre_name, **movie** (5/5; 26/26)
Sim-R	1116	349	775	date, time, category, price_range, rating, num_people, location, meal, **restaurant_name** (5/19; 9/23)
DSTC2	1612	506	1117	area, price range, **food** (1/73; 0/74)
WOZ 2.0	600	200	400	**area** (0/6; 1/7), price range, **food** (1/65; 2/72)

We evaluate our models on four benchmark datasets: Sim-M, Sim-R [128], DSTC2 [32] and WOZ 2.0 [104]. The statistics of the datasets are shown in Table 3.1.

Sim-M and Sim-R are specialized for scalable DST, which contain human-paraphrased simulated dialogues in the movie and restaurant domains. They have span annotations for all specified slot values (*i.e.* values other than *none* and *dontcare*) in the system and user utterances. In the event that the target slot value has multiple spans in the dialogue context, we use the span of the last occurrence as reference. The prevalence of out-of-vocabulary (OOV) values in Sim-M's *movie* and Sim-R's *restaurant_name* slots makes them particularly challenging and suitable for scalable DST evaluation.

DSTC2 and WOZ 2.0 are standard benchmarks for task-oriented dialogue systems, which are both in the restaurant domain and share the same ontology. In DSTC2, automatic speech recognition (ASR) hypotheses of user utterances are provided to assess DST models' robustness against ASR errors, so we use the top ASR hypothesis for validation and testing. In WOZ 2.0, the user interface is typing and collection of user utterances exhibit higher degree of lexical variation. ASR errors and flexible language use can cause problems in defining slot value spans, which is basis for our targeted scalable DST condition. The problem arises when erroneous ASR hypotheses do not contain a user's intended specified value or the user uses a *creative* expression in which a clear boundary of a value's span can be hard to define. For example, "My wife thinks she likes international but I don't want to take out a loan." is annotated with `price_range=cheap`. Such challenging instances in DSTC2 and WOZ 2.0 set the performance upper bound for our proposed scalable DST framework.

Note that in the evaluation, we do not apply an output canonicalization step to handle other possible valid variations of span. Therefore our model's predicted slot value span has to exactly match the label span to be considered correct.

Training Details

We use the pre-trained *[BERT-Base, Uncased]* model which has 12 hidden layers of 768 units and 12 self-attention heads for lower-cased input text. The span prediction loss for *{none, dontcar}* slots is set to zero. The total loss is defined as $(0.8\mathcal{L}^{\text{xent}}_{\text{cls}} + 0.1\mathcal{L}^{\text{xent}}_{\text{span\_start}} + 0.1\mathcal{L}^{\text{xent}}_{\text{span\_end}})$, where $\mathcal{L}^{\text{xent}}$ denotes the cross entropy loss for the corresponding prediction target. We update all layers in the model using ADAM optimization [34] with an initial learning rate $2e^{-5}$ and early stopping on the validation set. During training, we use 30% dropout rate [36] on the dialogue context encoder outputs. We also experiment with various rates of slot value dropout.

Table 3.2: Joint goal accuracy comparisons with prior approaches on Sim-M and Sim-R datasets. * indicates statistically significant improvement over BERT-DST model (paired sample t-test; $p < 0.01$). † indicates the corresponding model should be considered as a kind of oracle because the candidates are ground truth slot-tagging labels, *i.e.* the targeted slot value is guaranteed to be in the candidate list and considered by DST.

DST Models	Sim-M	Sim-R
DST + LU Candidates [126]	50.4%	87.1%
DST + Oracle Candidates† [96]	96.8%	94.4%
BERT-DST_SS	71.6%	87.4%
+ slot value dropout	76.3%*	87.6%
BERT-DST_PS	72.3%	88.6%*
+ slot value dropout	**80.1%***	**89.6%***

Results

Table 3.2 presents the performance of the proposed BERT-DST models compared to prior work on the scalable DST datasets Sim-M and Sim-R. In [126, 96], the DST component scores slot values from a candidate list, which is slot tagging predictions of a jointly-trained language understanding component (*DST + LU Candidates*), or the ground truth slot tagging labels (*DST + Oracle Candidates*). We compare our proposed model with the (*DST + LU Candidates*) baseline because in practice an oracle candidate list that always contains the target slot label is rarely available. On both Sim-M and Sim-R, BERT-DST_SS outperforms the baseline model. We attribute the performance gain to the effective contextualized representations obtained from the BERT dialogue encoding module. BERT-DST_PS with slot value dropout achieves further statistically significant improvement over BERT-DST_SS. The comparison of per-slot and joint goal accuracy of the different BERT-DST models is shown in Figure 3.2. We observe that it is mainly the slots with OOV values (*movie* for Sim-M and *restaurant_name* for Sim-R) that benefit from the encoder parameter sharing and slot value dropout techniques. The accuracy improvement on these bottleneck slots eventually leads to gain in the joint goal accuracy.

To investigate the effect of slot value dropout, we compare the performance with different

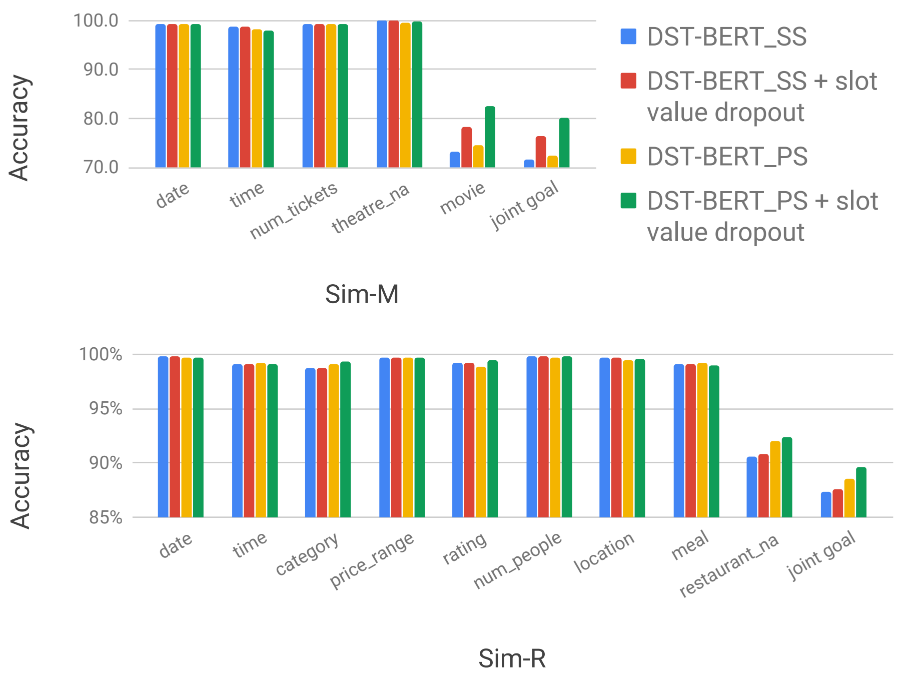


Figure 3.2: Per-slot and joint goal accuracy of the proposed models on Sim-M and Sim-R datasets

slot value dropout probabilities of BERT-DST_PS on Sim-M and Sim-R datasets, as shown in Figure 3.3. While a proper selection of slot value dropout rate can result in slight improvement on Sim-R, the effect of slot value dropout is more pronounced on Sim-M. Because of the high OOV value rate of the *movie* slot (100% OOV in test set), higher slot value dropout rate can be helpful for extracting unseen slot values from contextual patterns.

Table 3.3 presents the performance of BERT-DST with prior approaches on the standard DSTC2 and WOZ 2.0 datasets. Our work is more comparable with the top group frameworks, which are also designed for scalable DST to handle unknown ontology. The middle group of models require a predefined ontology to perform classification or scoring over a predefined set of possible slot values. On DSTC2, BERT-DST_PS shows comparable performance with prior scalable DST models, although not as high as the state-of-the-art models. On WOZ 2.0, BERT-DST_PS achieves competitive results with state-of-the-art models, which demonstrates BERT-DST's capability in understanding sophisticated language. Note that

Table 3.3: Joint goal accuracy comparison with prior approaches on DSTC2 and WOZ 2.0 datasets. We report the average and standard deviation of test set accuracy of 5 model runs with random training data shuffling and normal initialization on classification and span prediction weights.

DST Models	DSTC2	WOZ 2.0
DST + LU Candidates [126]	67.0%	-
DST + n-gram Candidates [118]	68.2±1.8%	-
DST + Oracle Candidates [96]	70.3%	-
Pointer Network [131]	72.1%	-
Delex.-Based Model [92]	69.1%	70.8%
Delex. + Semantic Dict. [92]	72.9%	83.7%
Neural Belief Tracker [92]	73.4%	84.2%
GLAD [134]	74.5±0.2%	88.1±0.4%
StateNet [127]	**75.5%**	**88.9%**
BERT-DST_PS	69.3±0.4%	87.7±1.1%

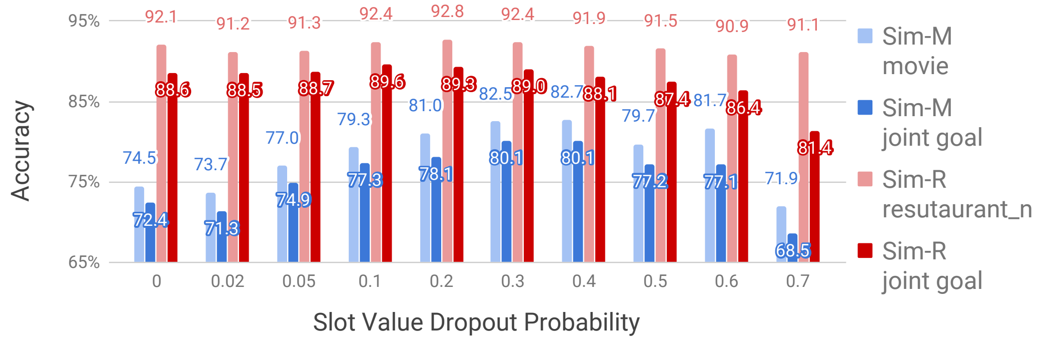


Figure 3.3: Comparison of different slot value dropout probabilities of the BERT-DST_PS model on Sim-M and Sim-R datasets.

it is not our goal to achieve state-of-the-art performance on the standard datasets. Instead, BERT-DST is tasked to handle unknown ontology and unseen slot values and does not require a separate candidate generation module.

3.3 Conclusion

We introduce BERT-DST, a scalable end-to-end dialogue state tracker that directly predicts slot values from the dialogue context with no dependency on candidate generation. In our framework, BERT is adopted to produce contextualized representations of dialogue context which are used to by the classification and span prediction modules to predict the slot value as *none*, *dontcare* or a text span in the dialogue context. The advantages of using BERT as dialogue context encoder include: (1) The contextualized word representations are suitable for extracting slot values from semantic context. (2) Pre-trained on large-scale language modeling datasets, BERT's word representations are good initialization to be fine-tuned to our DST problem. Moreover, we employ parameter sharing in the BERT dialogue encoder across all slots, which reduces the number of model parameters. Contextualized language representation can also benefit from more training examples of other slots. To prevent overfitting, we apply the slot value dropout technique. This step is critical for extracting unseen slot values from their contextual patterns. Empirical evaluation shows our model with cross-slot parameter sharing outperforms prior work on the benchmark scalable DST datasets Sim-M and Sim-R, and achieves competitive performance on the standard DSTC2 and WOZ 2.0 datasets.

Chapter 4

Robust Speech Recognition Using Multi-Modal Input

Robust speech recognition in cocktail party environments remains a challenging task in Automatic Speech Recognition (ASR). While current ASR systems have competitive performance in a low noise environment, their performance deteriorates when background noise or background speech is present.

In many real-world applications, before a speaker starts talking, a speech recognizer (such as a social robot) may already have access to the speaker's acoustic or visual data via one's mobile phones and social media, and could extract relevant speaker characteristics, ready to be used as additional features for speech recognition even before a speech signal is uttered by the target user. Moreover, the recognizer may know the speaker's identity by face recognition or various biometric identification methods and could use the identity to retrieve the speaker's information from the database of pre-extracted speaker embedding.

In this chapter, we tackle multi-modal speaker-targeted ASR of multi-speaker acoustic input signals in cocktail-party environments without the use of blind signal separation [57, 137]. With the term *speaker-targeted* model, we refer to a speaker-independent model with speaker

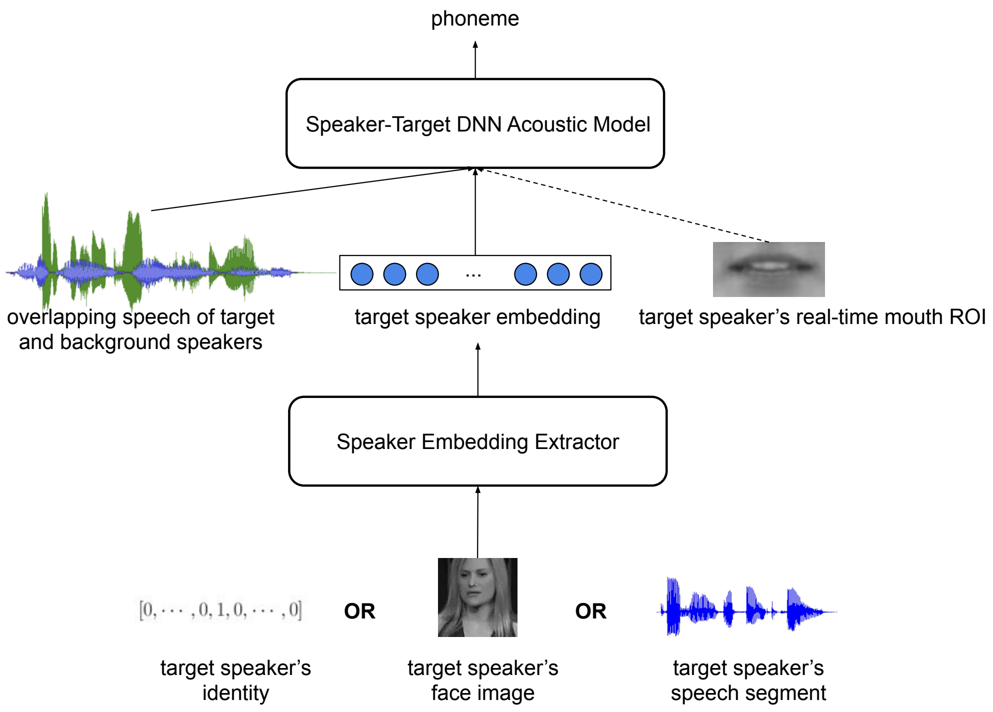

phoneme
Speaker-Target DNN Acoustic Model
overlapping speech of target and background speakers
target speaker embedding
target speaker's real-time mouth ROI
Speaker Embedding Extractor
[0, ⋯ , 0, 1, 0, ⋯ , 0]
OR
OR
target speaker's identity
target speaker's face image
target speaker's speech segment

Figure 4.1: Speaker-targetd acoustic modeling pipeline. The arrow connecting the target speaker embedding and the acoustic model is a dashed arrow. The audio-only model is illustrated without the dashed arrow. The speaker-targeted model is illustrated with the dashed arrow, where the target speaker embedding supplied as additional input feature to the acoustic model. The target speaker embedding is represented as a one-hot vector when the target speaker's identity is known and exists in training data; otherwise it can be extracted by a speaker embedding model given the available target speaker data (a speech segment or face image).

identity information input.

Speaker-Targeted Acoustic Model Pipeline We propose a pipeline of speaker-targeted acoustic model to recognize the speech of a target speaker from a mixture of speech signals [114, 88, 125, 110], as shown in Figure 4.1.

We complement the acoustic features with information of the target speaker's identity in embeddings similar to i-vectors in [29], along with raw pixels of the target speaker's mouth ROI images, to supply multimodal input features to a hybrid DNN-HMM for speech recognition in cocktail-party environments. In particular, we focus on recognizing a target speaker's speech from overlapping speech signals of two speakers: the target and the background speaker.

4.1 Audio-Visual Speaker-Targeted Speech Recognition

We approach this problem using DNN acoustic models with different combinations of additional modalities: visual features and speaker embedding information. The acoustic features are filterbank features extracted from the audio signals where two speakers' speech is mixed on a single acoustic channel. The visual features are raw pixel values of the mouth ROI images of the target speaker whose speech the system is expected to recognize. The speaker identity information is represented by the target speaker's ID-embedding.

4.1.1 Model

DNN acoustic models have been widely and successfully used in ASR [24]. Let $\mathbf{x}$ be a window of acoustic frames (i.e., context of filterbanks), the standard DNN acoustic models model the posterior probability:

$$p(y|\mathbf{x}) = \mathrm{DNN}(\mathbf{x}) \tag{4.1}$$

where y is a phoneme label or alignment (i.e., from GMM-HMM) and DNN is a deep neural network with softmax outputs. The DNN is typically trained to maximize the log probability of the phoneme alignment or minimize the cross-entropy error. However, this optimization problem is difficult when $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ is a superposition of two signals $\mathbf{x}_1$ and $\mathbf{x}_2$ (i.e., cocktail party).

we extend the previous traditional DNN acoustic models to leverage additional information in order to model our phonemes. By leveraging combinations of the visual features and speaker identity information, the standard DNN acoustic model is extended to have multimodal inputs.

We train the DNN acoustic models with four possible combinations of input features: (1) audio-only, (2) audio-visual, (3) speaker-targeted audio-only and (4) speaker-targeted audio-visual in two steps: *speaker-independent models training* followed by *speaker-targeted models training.* The details of the two steps are described in the following sub-sections.

Two-Speaker Speaker-Independent Models First, we leverage the visual information of the speaker's mouth region in conjunction with the acoustic features. The standard DNN acoustic model:

$$p(y|\mathbf{x}) = \mathrm{DNN_A}(\mathbf{x}) \tag{4.2}$$

with additional input of visual features becomes:

$$p(y|\mathbf{x}, \mathbf{w}) = \mathrm{DNN_{AV}}(\mathbf{x}, \mathbf{w}) \tag{4.3}$$

where $\mathbf{w}$ are the visual features. In this step, speaker-independent audio-only model $\mathrm{DNN_A}$ and audio-visual model $\mathrm{DNN_{AV}}$ are trained for the two-speaker cocktail-party problems. The acoustic and visual features are concatenated directly as DNN inputs for the audio-visual model. The speaker-independent models are illustrated in Figure 4.2, where the figure without the dashed arrow represents the audio-only model, and the figure with the dashed arrow represents the audio-visual model.

Two-Speaker Speaker-Targeted Models Secondly, we try to leverage the speaker identity information to extend the previous models, $\mathrm{DNN_A}$ and $\mathrm{DNN_{AV}}$. $\mathrm{DNN_A}$ is extended to:

$$p(y|\mathbf{x}, \mathbf{z}) = \mathrm{DNN_{AI}}(\mathbf{x}, \mathbf{z}) \tag{4.4}$$

and $\mathrm{DNN_{AV}}$ is extended to:

$$p(y|\mathbf{x}, \mathbf{w}, \mathbf{z}) = \mathrm{DNN_{AVI}}(\mathbf{x}, \mathbf{w}, \mathbf{z}) \tag{4.5}$$

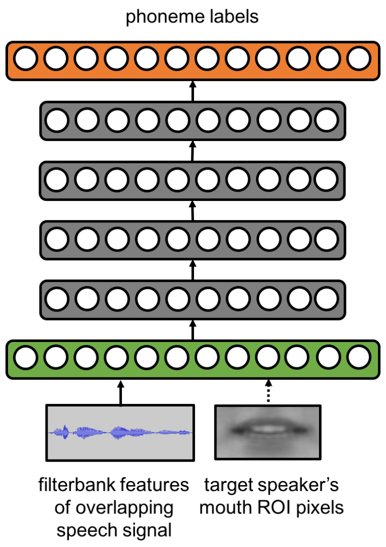

phoneme labels
filterbank features of overlapping speech signal
target speaker's mouth ROI pixels

Figure 4.2: Speaker-independent models. This figure illustrates a DNN architecture, where the phoneme labels are modeled in the output layer. The arrow connecting the visual features (mouth ROI pixels) and the input layer is a dashed arrow. The speaker-independent audio-only model is illustrated without the dashed arrow. The speaker-independent audio-visual model is illustrated with the dashed arrow, where the acoustic features (filterbank features) and video features are concatenated as DNN inputs.

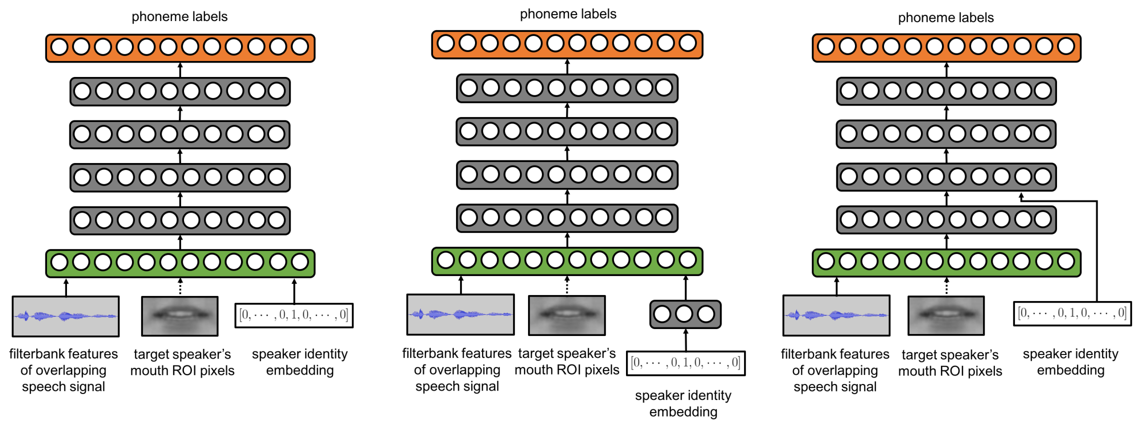
where $\mathbf{z}$ are the speaker identity information. In this step, we adapt the audio-only and audio-visual speaker-independent models to speaker-targeted models respectively (i.e., from DNN_A to DNN_AI and from DNN_AV to DNN_AVI) by hinting the network which target speaker to attend to by supplying speaker identity information as input. The speaker identity information is represented by an embedding that corresponds to the target speaker's ID. We investigate three ways to fuse the audio-visual features with the speaker identity information:

(A) Concatenating the speaker identity directly with audio-only and audio-visual features.

(B) Mapping speaker identity into a compact but presumably more discriminative embedding and then concatenating the compact embedding with audio-only and audio-visual features.

(C) Connecting the speaker identity to a later layer than audio-only and audio-visual features.

The three fusion techniques introduce the three variants (A), (B) and (C) of both the speaker-targeted models DNN_{AI} and DNN_{AVI}. The speaker-targeted models of the three invariants are shown in Figure 4.3, where the figures without the dashed arrow represent the audio-only models, and the figures with the dashed arrow represent the audio-visual models.


phoneme labels
filterbank features of overlapping speech signal
target speaker's mouth ROI pixels
speaker identity embedding
[0, ··· , 0, 1, 0, ··· , 0]
phoneme labels
filterbank features of overlapping speech signal
target speaker's mouth ROI pixels
[0, ··· , 0, 1, 0, ··· , 0]
speaker identity embedding
phoneme labels
filterbank features of overlapping speech signal
target speaker's mouth ROI pixels
speaker identity embedding
[0, ··· , 0, 1, 0, ··· , 0]

(A) concatenating the speaker identity directly with audio-only and audio-visual features

(B) mapping speaker identity into a compact but presumably more discriminative embedding and then concatenating the compact embedding with audio-only and audio-visual features

(C) connecting the speaker identity to a later layer than audio-only and audio-visual features

Figure 4.3: Three variants of speaker-targeted models. These figures illustrate three fusion techniques of audio-visual features with speaker identity information in a DNN architecture, where the phoneme labels are modeled in the output layer. The arrows connecting the visual features and the input layers are dashed arrows. The speaker-targeted audio-only models are illustrated without the dashed arrows. The speaker-targeted audio-visual models are illustrated with the dashed arrows, where the acoustic and video features are concatenated as DNN inputs.

Moreover, we train single-speaker speaker-independent models in comparison with the two-speaker speaker-independent models. We also train 6 randomly-selected speaker's speaker-dependent models (adapted from speaker-independent models as well) to compare with the speaker-targeted models.

4.1.2 Experiments

Dataset

The GRID corpus [13] is a multi-speaker audio-visual corpus. This corpus consists of high-quality audio and video recordings of 34 speakers in quiet and low-noise conditions. Each of the speakers read 1000 sentences which are simple six-word commands obeying the following syntax:

$command $color $preposition $letter $digit $adverb

We use the utterances of 31 speakers (16 males and 15 females) from the GRID corpus, excluding speaker 2, 21 and 28 and part of the utterances of the remaining 31 speakers due to the availability of mouth ROI image data. In the one-speaker datasets, there are 15395 utterances in the training set, 548 in the validation set, and 540 in the testing set, following the convention of CHiME Challenge [27]. The GRID corpus utterances that don't belong to the one-speaker datasets are termed background utterance set. To simulate the overlapping speech audios for the two-speaker datasets, we mix the target speaker and a background speaker's utterances with equal weights on a single acoustic channel using SoX software [155]. The background speaker's utterances are randomly selected from the background utterance set excluding the utterances of the target speaker. The resulting mixed audio's length is as long as the length of the target speaker's utterance. Since we also train speaker-dependent speech recognizers for individual speakers, for each target speaker's utterance, we mix it with more background utterances of other speakers in order to generate enough data for speaker-dependent training. There are in total of 523430 utterances in the training set, 548 in the validation set, and 540 in the testing set in the two-speaker datasets.

Feature Extraction

Audio Features Log-mel filterbank features with 40 bins are extracted, and a context of ± 5 frames was used for audio input features (i.e., $440 = 40 * (5 + 1 + 5)$ dimensions per

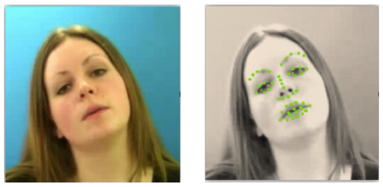


Figure 4.4: Facial landmarks extracted by IntraFace

acoustic feature $\mathbf{x}$).

Visual Features We use target speaker's mouth ROI images' pixel values as visual features. The facial landmarks are first extracted by IntraFace software [50] as shown in Figure 4.4, and each video frame is cropped into a 60 pixel * 30 pixel mouth ROI image [53] according to the mouth region landmarks (i.e., $1800 = 60 * 30$ dimensions per visual feature $\mathbf{w}$). The gray-scale pixel values are then concatenated with audio features to form audio-visual features.

Speaker Identity Information Speaker identity information is represented by the target speaker's ID-embedding, which is simply a one-hot vector of thirty-three 0s and a single 1, $[0, \cdots, 0, 1, 0, \cdots, 0]$, in which the entry of 1 corresponds to the target speaker's ID (i.e., 34 dimensions per speaker identity embedding $\mathbf{z}$).

Training Details

Here we describe the architecture of our DNNs. The number of hidden layers for audio-only models and speaker-independent audio-visual models is 4, while it is 5 for speaker-targeted and speaker-dependent audio-visual models. Each hidden layer contains 2048 nodes. Rectified linear function (ReLU) is used for activation in each hidden layer. The output layer has a softmax of 2371 phoneme labels. We use stochastic gradient descent with a batch size of 128 frames and a learning rate of 0.01.

Results

The aforementioned single-speaker models are used to decode the single-speaker testing dataset, while the two-speaker models are used to decode the two-speaker testing dataset.

Table 4.1: WER comparisons of single-speaker models on GRID corpus

	audio-only	audio-visual
speaker-independent	0.3%	0.4%

Table 4.2: WER comparisons of two-Speaker models on GRID corpus

	audio-only	audio-visual
speaker-independent	26.3%	4.4%
speaker-targeted A	4.0%	3.6%
speaker-targeted B	3.6%	3.9%
speaker-targeted C	4.4%	4.4%
speaker-dependent	3.9%	3.4%

Table 4.1 shows the WER of single-speaker models. Table 4.2 shows the WER of two-speaker models. The audio-only baseline for two-speaker cocktail-party problem is 26.3%. The results of speaker-independent models for single-speaker and two-speaker suggest that automatic speech recognizers' performance degrades severely in cocktail-party environments compared to low-noise conditions. It is also demonstrated that the introduction of visual information to acoustic features can reduce WER significantly in cocktail-party environments, improving the WER to 4.4%, although it may not help when the environmental noise is low. WER comparisons between two-speaker's audio-only speaker-independent and speaker-targeted models suggest that using speaker identity information in conjunction with acoustic features achieves a better improvement on WER, reducing WER up to 3.6%.

The results of two-speaker's speaker-targeted models A, B, and C suggest a weak tendency that providing speaker information in earlier layers of the network seems to have advantage. WER comparisons between two-speaker speaker-dependent and speaker-targeted models suggest an intuitive result that a speaker-dependent ASR system which is optimized for one specific speaker performs better than a speaker-targeted ASR system which is optimized for multiple speakers simultaneously. We also find the introduction of visual information improves the WER of speaker-dependent acoustic models while it doesn't improve the speaker-

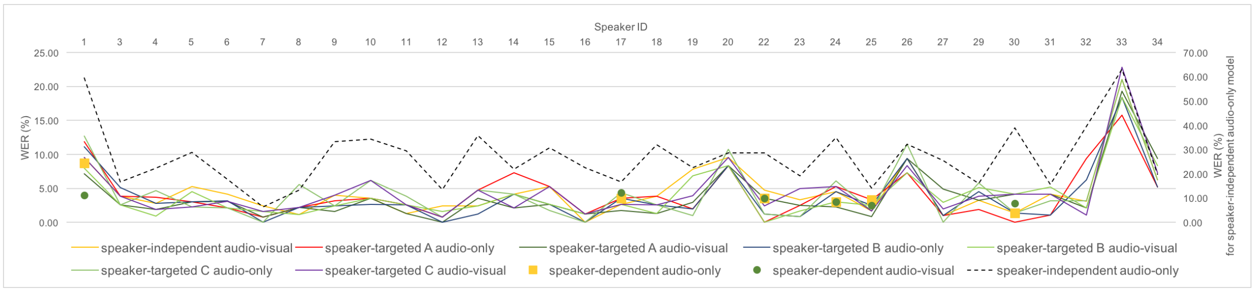


Figure 4.5: WER comparisons of two-speaker models for individual speakers on GRID corpus. WER of two-speaker models for individual speakers are illustrated. The dashed line is plotted on the right vertical axis which represents the speaker-independent audio-only model. The solid lines and markers are plotted on the left vertical axis. Speaker-dependent models for speaker 1, 17, 22, 24, 25 and 30 are plotted in markers. The chart demonstrates a similar trend between different models' performance on individual speakers.

targeted acoustic models. We subscribe this finding to the limitation of the capacity of the neural network architecture that we use for both models, that it is able to optimize for one specific speaker's visual information in a speaker-dependent model, but not powerful enough to learn a unified optimization for all 31 speakers' visual information in a single speaker-targeted model. Figure 4.5 illustrates the WER of the individual speakers. A similar trend between different models' performance on individual speakers is demonstrated.

4.2 Offline Speaker Embedding Estimation

Since real-time visual information used in audio-visual speech recognition may not always be available, we further propose an offline speaker embedding estimation model that only requires very low resource of speaker data [138]. We investigate three types of text-independent speaker embedding models (i-vector, x-vector and f-vector), using a speech segment or face image as input to generate speaker embedding and compare their performance in different speaker and environment conditions.

4.2.1 Speaker Embedding Models

We train and compare three types of text-independent speaker embedding models. i-vector and x-vector models are audio-based, taking a speech segment of the target speaker as input,

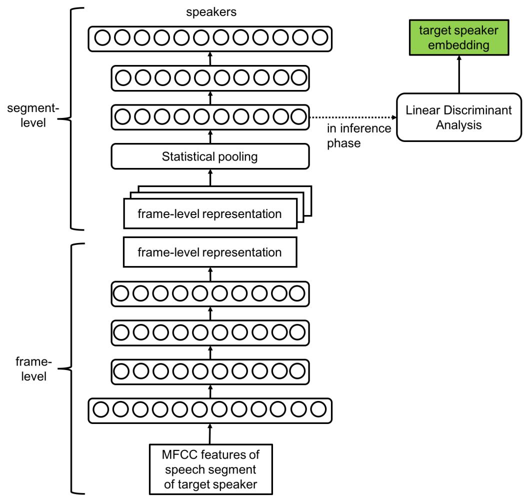

speakers
target speaker embedding
segment-level
Linear Discriminant Analysis
in inference phase
Statistical pooling
frame-level representation
frame-level representation
frame-level
MFCC features of speech segment of target speaker

Figure 4.6: TDNN based x-vectors extractor. TDNN-based model extracts x-vectors from bottleneck features given the target speaker's speech segment as input. The TDNN model is pre-trained for speaker recognition.

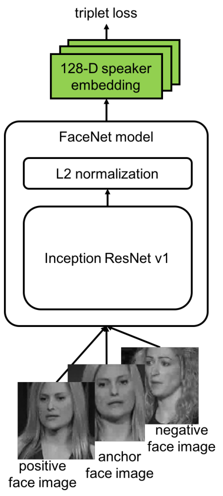


Figure 4.7: Facenet: CNN-based face vectors extractor. FaceNet extracts f-vectors from an input face image of the target speaker. FaceNet is a deep Convolutional Neural Network of Inception-ResNet architecture pre-trained on triplet loss.

and the f-vector model is vision-based, taking a face image of the target speaker as input.

GMM-UBM i-vector Model Firstly, we consider the conventional i-vector extractor [22]. We use a 512-component diagonal Gaussian Mixture Model-Universal Background Models (GMM-UBMs). Iteratively, we update the UBM components' means (factor loading sub-matrices), and collect sufficient statistics to estimate i-vectors, using the EM algorithm. Then Linear Discriminant Analysis (LDA) is used to reduce the dimension of i-vectors to 128. i-vector extractor training is performed at frame-level MFCC features. In the inference phase, given a speech segment of the target speaker, we extract a frame-level i-vector every 10 frames, and the *i-vector* speaker embedding is extracted by taking the average of all the frame-level i-vectors across the entire segment.

TDNN-based x-vector Model The second speaker embedding model is a speaker recognition model, with the Time-Delay Neural Network (TDNN) architecture as in [99]. The input of the model is the MFCC features of all frames across a speech segment, and the output of the model is a speaker label. It is trained on multiclass cross entropy loss, to differentiate the 1228 speakers in the training set. The architecture of the TDNN model for speaker embedding is shown in Figure 4.6. The first four layers are time-delay and fully-connected layers which take a context of MFCC features as input to generate a frame-level representation. The middle statistical pooling layer aggregates all frame-level representations of the entire speech segment, and concatenates the mean and standard deviation as statistical features. The last three layers are fully-connected layers and a softmax output layer for speaker classification. In the inference phase, we use the bottleneck features from the second to last hidden layer, projected through LDA to 128-dimensional, as speaker embedding. We call it *x-vector*.

CNN-based f-vector Model The third speaker embedding model we consider is FaceNet [48]. It is a deep Convolutional Neural Network (CNN) which takes face images as input and outputs 128 dimensional embeddings which are L2 normalized (rescaled to a unit hypersphere). It is trained on triplet loss, where each triplet training example consists of three face images: an anchor, a positive, and a negative image. The anchor and positive images belong to the same speaker, and the negative image belongs to another speaker. Triplet loss of a triplet training example is given by:

$$\left[\|e^a - e^p\|_2^2 - \|e^a - e^n\|_2^2 + \alpha \right]_+ \tag{4.6}$$

where e^a, e^p, e^n represent the face embeddings of the anchor, positive and negative face images, α is the distance margin, and the operator $[x]_+ = \max(x, 0)$. Training on triplet loss essentially forces the network to project the face images such that face image embeddings of the same speaker have small Euclidean distances and the face image embeddings of distinct speakers have large distances. We use a FaceNet model with the Inception ResNet v1 architecture, described in [100], which was pre-trained on the MS-Celeb-1M dataset [60], as shown in Figure 4.7. In the inference phase, we supply a face image of the target speaker as input to the FaceNet model and use the 128-D vector output as speaker embedding. If more than one face image of the target speaker is provided, we use the average of all the 128-D vector outputs as speaker embedding. We call it *f-vector*.

4.2.2 Experiments

Table 4.3: Statistics of TED-LIUM2 corpus

	Train	Validation	Test
Number of videos	1444	8	11
Number of utterances	97876	507	1155
Duration of videos (hours)	341.6	1.7	2.9
Number of speakers	1228	8	11

Dataset

For the dataset of single-speaker scenario, we directly use the real-world acoustic data from the TED-LIUM corpus release 2 [35]. We augment the dataset with the speaker face images. The speaker face images are obtained by applying face detection and tracking on the video frames of TED talks, which are downloaded from TED.com. Table 4.3 lists the statistics of the TED-LIUM2 corpus dataset. To simulate the overlapping speech audios for the two-speaker scenario, we mix two utterances with equal energy on a single acoustic channel using SoX software, one from the target speaker and the other from a background speaker. The same mixture process is applied for each split of the dataset, which we describe as follows. Take the training split for example. Each of the utterances in the split will be used as the target speaker's speech. For each target speech utterance, we randomly select a different speaker's utterance from the same split as the background speech, so that there is no overlap in background speech utterances between different datasets. The length of the resulted mixed audio is as long as that of the target speaker's utterance. Both datasets of the single-speaker and two-speaker scenarios have the same split as the TED-LIUM corpus release 2.

Speaker Embedding Robustness against Environment Variation

To evaluate the speaker embedding robustness against environment variation, we compare two sources of the target speaker's information (i.e. a segment of speech or a face image)

1. **Same Env:** information from the same environment as the input speech to be recognized, achieved by randomly selecting another utterance in the same TED talk of the target speaker's speech.

2. **Diff Env:** information from a different environment as the input speech to be recognized, achieved by randomly selecting an utterance in another TED talk of the target speaker.

as input to speaker embedding models. The former condition signifies the effect of speaker embedding if we know how the target speaker sounds or looks like in the exact environment for speech recognition. The latter condition represents the situation when we only know what the target speaker sounds or looks like in a different environment from speech recognition, which is more likely to happen in the real world and helps us validate the robustness of target speaker embeddings with respect to different environments.

Training Details

Here we describe the architectures of our DNN models. Our acoustic model is a TDNN-based chain model [66]. Let t denote the current timestep. At the input layer, we splice together the 40-bin MFCC features of frames at $\{t-1, t, t+1\}$ and the 128-dimensional speaker embedding. The first hidden layer is fully connected of 450 nodes. Each of the second to sixth hidden layers has 450 nodes and splices the output of the previous layer at times $\{t-1, t, t+1\}, \{t-1, 0, t+1, t+2\}, \{t-3, t, t+3\}, \{t-3, t, t+3\}, \{t-6, t-3, t\}$ respectively. The output layer is a softmax function of 3683 phoneme classes. We train with stochastic gradient descent (SGD) with a batch size of 128 frames and an initial learning rate of 0.001. The input layer of our TDNN-based x-vector model is 40-bin MFCC features. Each of the first 3 hidden layers has 512 nodes and splices the output of the previous layer at times $\{t-2, t-1, t, t+1, t+2\}, \{t-2, t, t+2\}, \{t-3, t, t+3\}$ respectively. The fourth and fifth layers are fully connected of 512 and 1500 nodes. After the statistical pooling layer are two fully-connected layers of 512 nodes. We use SGD with a batch size 64 frames and an initial learning rate of 0.001. All the nonlinearities for the hidden fully-connected and time-delay layers are rectified linear units followed by batch normalization.

Results

Table 4.4: WER comparisons of single-speaker models on TED-LIUM2 corpus

Audio-only	14.8%	
Speaker Embedding	Same Env	Diff Env
i-vector	**14.2%**	15.4%
x-vector	**14.2%**	**14.3%**
f-vector	14.4%	14.4%

Table 4.5: WER comparisons of two-speaker models on TED-LIUM2 corpus

Audio-only	65.7%	
Speaker Embedding	Same Env	Diff Env
i-vector	**29.5%**	52.4%
x-vector	34.6%	**41.0%**
f-vector	54.5%	51.2%

Table 4.4 shows the Word Error Rate (WER) of the single-speaker models. Table 4.5 shows the WER of the two-speaker models. Comparing the results of the single-speaker and two-speaker models, we observe that ASR performance degrades severely in cocktail-party environments compared to single-speaker conditions which have lower background interfering speech. The audio-only baseline without using a static target speaker embedding input for two-speaker cocktail-party problem achieves 65.7% WER. It is demonstrated that using all three types of speaker embedding as an additional feature can reduce the WER significantly, improving WER to as low as 29.5%. Even in relatively lower noise environments, the results of single-speaker show that using any of the three types of speaker embedding as an additional feature still provides a small gain, reducing the WER from 14.8% to as low as 14.2%.

Moreover, we compare the performance of the three types of speaker embedding. When the target speaker embedding is extracted using the speaker data from the same environment as speech recognition, i-vector shows better performance than x-vector and f-vector in both single-speaker and cocktail-party conditions. However, when the target speaker embedding is extracted using the speaker data from a different environment, x-vector and f-vector outperform i-vector. This suggest i-vector may tend to overfit the exact speaker and environment condition, while x-vector and f-vector show robustness against environment changes. This is expected because i-vector accounts for both speaker and environment acoustic variations at the same time, while x-vector and f-vector are trained to discriminate speaker identities, regardless of environments. We also observe that x-vector outperforms f-vector in both conditions. While f-vector characterizes the target speaker's identity (or even gender) from the speaker's face image, it does not have acoustic information about the target speaker's voice attributes. On the other hand, x-vector, using speech segment as input to the speaker embedding model, not only represents the target speaker's identity, but also retains one's voice characteristics.

4.3 Conclusions

In this chapter, we first propose a speaker-targeted audio-visual DNN-HMM model for speech recognition in cocktail-party environment. Different combinations of acoustic and visual features and speaker identity information as DNN inputs are presented. Experimental results suggest that the audio-visual model achieves significant improvement over the audio-only model. Introducing speaker identity information introduces an even more pronounced improvement. Combining both approaches, however, does not significantly improve performance further.

Secondly, we present using a static target speaker embedding as additional acoustic modeling input feature for speaker-targeted speech recognition in cocktail party environments. We investigate three types of text-independent speaker embedding extraction models, using the target speaker's data as simple as a short speech segment or a face image. i-vector is extracted by the GMM-UBM with speech segment of the target speaker. x-vector is extracted from the bottleneck features of a TDNN-based speaker recognizer given a speech segment of the target speaker. f-vector is extracted by a CNN of Inception-ResNet architecture trained on triplet loss given an input face image of the target speaker. Empirical evaluation is performed on the TED-LIUM corpus release 2 by overlapping a target speaker and a background speaker's speech signals. We show that using the target speaker embedding reduces WER over the acoustic feature only model from 65.7% to 29.5% in two-speaker settings. Comparing the three types of speaker embedding, i-vectors tend to overfit to specific speaker and environment conditions and show the best performance when the environment of speaker data matches that of speech recognition, while x-vectors and f-vectors are more robust with respect to environment variations, as they are trained to differentiate speakers.

Chapter 5

Policy for Vision-Grounded Instruction Following

In this chapter, we focus on dialogue policy learning on the task of vision-grounded instruction following [122, 139, 152]. In our setting, the agent receives an instruction (switch an appliance or fetch an object) in text form and learns to navigate and manipulate objects to complete the instruction in a simulated indoor multi-room environment, as shown in Figure 5.1. Vision-grounded instruction following involves understanding both the language and visual context (instruction description, the objects and the environment surrounding the agent) and decision making (a policy that outputs a series of actions for the agent to carry out in order to complete the instruction).

Unlike studies on Vision-and-Language Navigation which focus on understanding the complex and photo-realistic visual scenes from collections of real buildings scanned data, we train our instruction following agent using a simulator which maximizes the randomness of the environment layout, object positions and appearances to enable Domain Randomization [101]. Despite simplifying the visual understanding aspect, our goal is to train an instruction following agent that is able to generalize to act in diversified unseen environments. We create a series of subtasks which are simplified from the target task along two complexity dimensions: instruction variety and spatial layout.

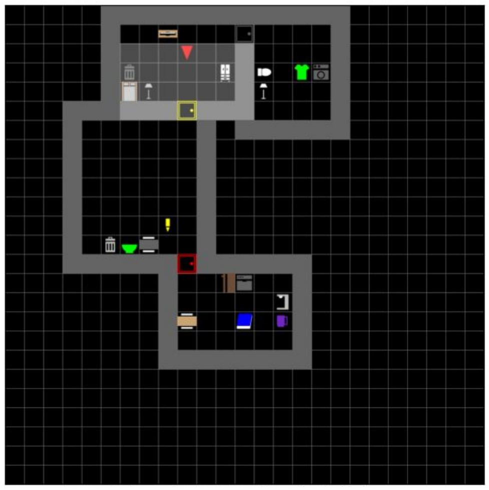

Example Instruction 1: You must fetch the purple cup.
Example Instruction 2: Go turn on the tv in the living room.

Figure 5.1: Example illustration of the vision-grounded instruction following task. The goal of the agent (represented by the red arrow) is to complete the given instruction by navigating the environment.

5.1 Vision-Grounded Instruction Following Framework

Reinforcement Learning Environment Setup

MiniGrid Our training environment is based on the Minimalistic Gridworld Environment (MiniGrid) [113] package's `Multiroom` task and `Fetch` task with several modifications. Within the 25 by 25 grid area, in each episode, the number of rooms is randomly chosen between 2 to 4, the width and height of each room is randomly chosen from 6 to 9 grid cells, and the room layout and door locations are also randomly generated. The rooms are randomly assigned to have different functions and the furniture and appliances are chosen and placed randomly. Household objects randomly chosen from 5 types and 5 colors are also placed at randomly locations. The number of objects is defined to be the number of rooms plus one. The details of furniture and appliances for different room types and the types and colors of objects are listed in Table 5.1. The agent's starting position and orientation are

Room type	Furniture and Appliances
living room	sofa, table, TV, light, trash bin
kitchen	refrigerator, coffeemaker, dishwasher, table, light, trash bin
bedroom	bed, table, wardrobe, TV, light, trash bin
bathroom	washer, toilet, light, trash bin
Object types	pen, book, cup, bowl, shirt
Object colors	red, green, blue, yellow, purple

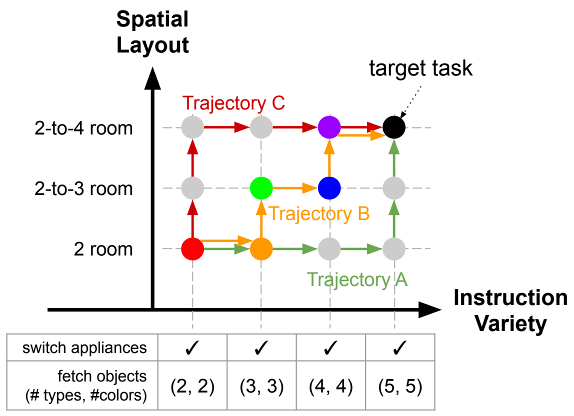
Table 5.1: Household furniture, appliances and objects defined in our vision-grounded instruction following environment.

also randomly determined in the beginning of each episode.

We choose to extend the MiniGrid training framework for its strength in Domain Randomization [101, 123]. MiniGrid provides high flexibility to create complex room layouts and its object, furniture and appliance positions are completely random. By training the model that works across the highly diversified simulated environments, we expect the agent to generalize better to unseen room environments.

Instruction We define two types of instructions to instruct the agents: (1) *fetch* the specified object, *e.g.* "get the orange cup in the kitchen", and (2) *switch* the specified appliance (light or TV), *e.g.* "please switch on the light in the bedroom". For the *fetch* instruction, task success is achieved when the agent picks up the specified object. For the *switch* instruction, task success is achieved when the agent operates the specified appliance to be on. When the agent takes more than *max_step* steps before completing the instructed task, it is counted as task failure. Because in our environment the objects have more variety than appliances, which makes the *fetch* instruction more challenging, we set each episode to have 75% probability to have the *fetch* instruction and 25% for the *switch* instruction. For both instructions, we use multiple forms of natural language expressions to mimic real human's instructions.

Subtasks In the target task, there are two kinds of instructions: switch appliances (light and TV) and fetch objects (5 types in 5 colors) in a 2-to-4-room environment. We define a series of subtasks as simplified versions of the target task along two complexity dimensions: *instruction variety* and *spatial layout*, as shown in Figure 5.2, based on heuristic.



switch appliances	✓	✓	✓	✓
fetch objects (# types, #colors)	(2, 2)	(3, 3)	(4, 4)	(5, 5)

Figure 5.2: The target task (represented by the black dot) can be simplied to easier subtasks along two complexity dimensions: instruction variety and spatial layout. The subtasks are the grid points. The spatial layout dimension can be divided to three levels of complexity: `2` room, `2-to-3` room, and `2-to-4` room. The instruction variety dimension can be divided to four levels of complexity: switch appliances plus fetch objects of four levels of varieties.

Reinforcement Learning Agent

Agent Model

Action The agent's actions include six basic motions: *turn left*, *turn right*, *move forward*, *operate furniture or appliance*, *pick up object*, and *put down object*, with three additional dialogue actions: *inquire best motion action*, *inquire navigation hint*, and *inquire recognition hint*.

Observation In every step, the agent's observations include the agent's partially observable *view image* (defined to be 7 by 7 grid area) and an *instruction description* text received in the beginning of an episode. The image input feature dimension is 7x7x3; the first two dimensions are 7*7 grid cells of the agent's partially observable front view (with occlusion by walls and closed doors), and the third dimension is a grid cell's 3-tuple state: (object_type, object_color, object_status). We define two additional observation inputs to support human-agent dialogue communication: *response* from the human collaborator to answer the agent's query and the number of *remaining_dialogue_turns*. We define *max_dialogue*, the number of dialogue exchanges allowed in each episode. In the beginning of an episode, *remaining_dialogue_turns* will be set to *max_dialogue*. When the agent takes one of the dialogue actions at step t, it will receive a *response* from the human collaborator in its observation at step $t + 1$, and the value *remaining_dialogue_turns* will be decremented by 1. When the number of dialogue exchanges exceeds *max_dialogue* dialogue actions (*i.e.*, the agent's observation of *remaining_dialogue_turns* is 0), even when the agent takes one of the dialogue actions to inquire the human collaborator, it will not receive a *response*.

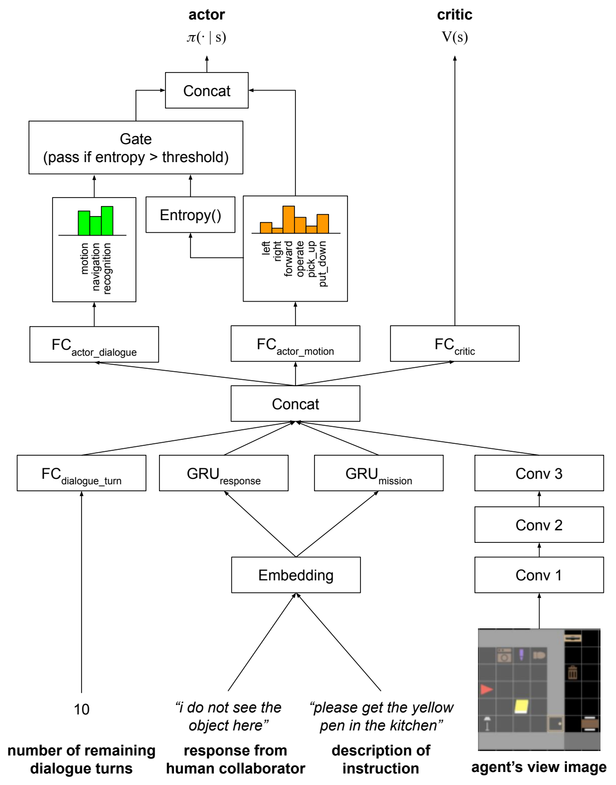


Figure 5.3: Instruction following agent model architecture. The response from human collaborator input is optional.

Model Architecture The model architecture for our instruction following agent is shown in Figure 5.3. The agent has four sources of observations as input: *instruction* description, *view image*, *response* from the human collaborator, and number of *remaining dialogue turns*. The text of instruction description and response from the human collaborator are both tokenized and mapped by the shared word embedding. Then they are encoded by their respective GRU network to produce the utterance-level encoding vectors (final hidden states of GRU). The view image is encoded by a Convolutional Neural Network (CNN), composed of a convolutional layer, a maxpooling layer, and then two convolutional layers, which produces the image encoding. The integer value of the number of remaining dialogue turns is projected through a fully connected layer which generates the encoding for remaining dialogue turns. All the encodings (image, instruction, response, and dialogue turn) are concatenated, and then fed as input to three feedforward networks of one fully-connected hidden layer. One network is the actor network for motion actions, predicting a distribution over the 6 motion actions. The second network is actor for dialogue actions, predicting a distribution over the 3 dialogue actions. And the third network's output is the critic's estimated value of the current state. To encourage the agent to take the dialogue actions only when it is confused about which motion actions to take, we introduce a gating mechanism to control the flow of the dialogue action predictions, and use the entropy of the motion actions to approximate the agent's confusion level. When the entropy of the motion action distribution is above the *entropy_threshold*, the gate lets the dialogue action distribution pass; otherwise the dialogue actions will be set to zero likelihood. Finally the distributions of the motion actions and the dialogue actions are concatenated and normalized as the the overall actor network's prediction.

Policy Learning

The agent's model network is trained using the Advantage Actor-Critic (A2C) [105] algorithm. The critic network learns to estimate the value function V, which is then used to calculate the advantage function:

$$\begin{aligned} A(s_t, a_t) &= Q(s_t, a_t) - V(s_t) && (5.1) \\ &= r_{t+1} + \gamma V(s_{t+1}) - V(s_t) && (5.2) \end{aligned}$$

The actor network parameterizes the policy, and is updated in the direction suggested by the critic network, similar to other policy gradient algorithms. The policy gradient can be written as:

$$\begin{aligned}\nabla_\theta J(\theta) &\sim \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)(r_{t+1} + \gamma V(s_{t+1}) - V(s_t)) \qquad (5.3)\\ &= \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) A(s_t, a_t) \qquad (5.4)\end{aligned}$$

5.2 Curriculum Learning

The principle of curriculum learning is starting with simpler training samples and gradually increasing the complexity of training data, such that training on such curriculum will help the performance on the target task [16, 45].

A series of subtasks is selected to form the learning curriculum. Then the instruction following agent will be trained for a number of steps in each subtask according to the curriculum. This type of curriculum learning strategy can be categorized in the task-specific curriculum learning family [16]. We explore two types of curriculum learning strategies to boost RL agent's training: manually designed curriculum and automatic curriculum.

5.2.1 Manual Curriculum

To manually design a learning curriculum, we consider the following hyperparameters: subtask trajectory, curriculum pacing and subtask ordering.

Subtask Trajectory Because the subtask complexity is two-dimensional, there exist multiple subtask trajectories from the simplest task to the target task in monotonically increasingly complexity, e.g. trajectory A, B, and C as shown in Figure 5.2.

Curriculum Pacing The pace of the curriculum determines the number of training steps to spend on each subtask in the curriculum. We compare different training steps in a fixed pacing function, where all subtasks are trained with the same number of steps.

Subtask Ordering Does ordering of the subtasks in the learning curriculum affect learning effectiveness? Besides the regular increasing complexity curriculum, we also consider an "anti" curriculum where the subtasks appear in decreasing complexity, and a "random" curriculum with randomly shuffled subtask order.

5.2.2 Automatic Curriculum

While it is possible to manually design a reasonable learning curriculum by carefully choosing the subtask trajectories, ordering and pacing, we are also interested in generating the curriculum automatically. [146] proposed the Teacher-Student Curriculum Learning, where the agent model (Student) is trained according to the curriculum generated by the Teacher model. The Teacher model uses the Student's past episode returns as input, and at each timestep it predicts a subtask for the Student to practice for a few episodes and the episode returns are fed as input back to the Teacher model. The goal of the Teacher is to maximize the sum of performance for all the subtasks, with the assumption that the performance gain on subtasks will translate to improved performance in the target task.

Learning the Teacher model can be formulated as a non-stationary multi-armed bandit problem. Let's denote the subtasks as $\{u_1, \cdots, u_K\}$. The horizon is denoted as T. In each timestep t, the Teacher picks a subtask $u_t = u_k, k \in [K]$. The Student trains on u_k and returns the episode return x_t. The teacher's goal is to maximize $\sum_{t=1}^{T} x_t$.

We compare three types of mutli-armed bandit algorithms to train the Teacher model: Online Algorithm, Window Algorithm, and Sampling Algorithm [129].

Online Algorithm In Online Algorithm, we define the Teacher's reward r_t as the change in student's episode return $r_t = x_t - x_{t'}$, where t' is the previous timestep when u_k was trained on. And we approximate the expected reward Q for different subtasks using exponentially weighted moving average as $Q_{t+1}(u_t) = \alpha r_t + (1-\alpha)Q_t(u_t)$, where α is learning rate. The next subtask u_{t+1} is drawn from the Boltzmann distribution $p(u) = \frac{e^{Q_{t+1}(u)/\tau}}{\sum_{k=1}^{K} e^{Q_{t+1}(u_k)/\tau}}$, where τ is the temperature of Boltzmann distribution.

Window Algorithm The Window Algorithm shares the same definition and update rule of expected reward Q and subtask selection policy as Online Algorithm. The only difference is that the reward r_t is instead defined as the slope of the subtask u_k's last N episode returns $\{x_t, x_{t'}, \cdots, x_{t^{(N)}}\}$, calculated using linear regression.

Sampling Algorithm In Sample Algorithm, the Teacher's reward has the same definition as Online Algorithm. Inspired by Thompson Sampling [21], we store the last N episode returns $\{x_t, x_{t'}, \cdots, x_{t^{(N)}}\}$ in a buffer for each subtask u_k. Subtask selection is done by sampling a recent reward from each of the subtasks' buffer, and the subtask whose buffer yields the highest sampled reward is chosen as u_{t+1}.

5.2.3 Experiments

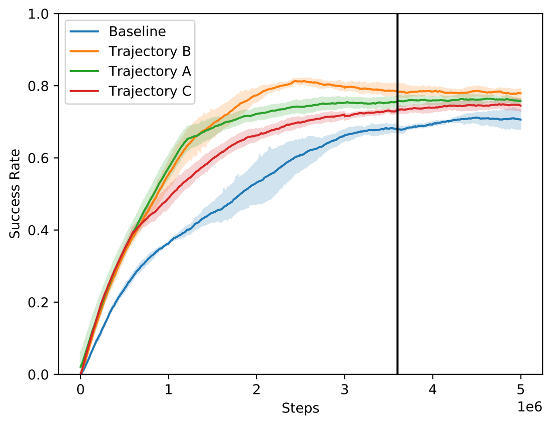


Figure 5.4: Learning curves comparing subtask trajectories. Training on the target task begins at the vertical line. We plot the mean and one standard deviation of four runs of random initialization for each agent variant. Trajectory B shows statistically significant improvement over Trajectory C and Baseline ($p < 0.05$ in Welch's t-test).

The learning curves of manual curriculum learning are shown in Figure 5.4 through 5.6. For subtask trajectory, we compare three trajectories (A: instruction-first, C: spatial-first, and B: alternating-axes) with increasing complexity, as shown in the Figure 5.2. Figure 5.4 shows that training with the curriculum following the trajectories A, B and C all outperform the baseline (only training on the target task from the beginning), and trajectory B has the best performance. In the following experiments, we use Trajectory B as the default manual

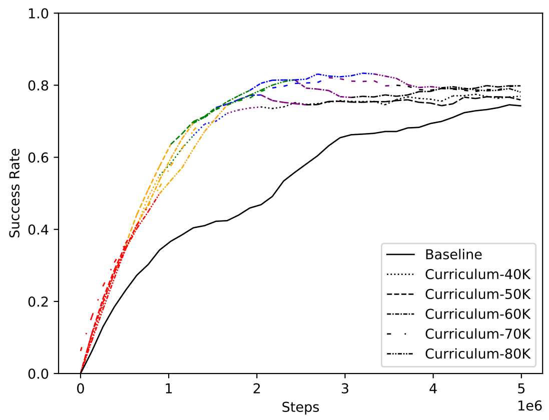


Figure 5.5: Learning curves comparing curriculum pacing step sizes. Each subtask is represented by one color, with the same color scheme as Figure 5.2. Training on the target task is represented using black color.

curriculum. In Figure 5.5, we compare training each subtask for 400K, 500K, 600K, 700K, or 800K steps respectively in a fixed pacing schedule. We observe that a moderate pacing which balances between under training and over training on subtasks shows the best performance and outperforms the baseline. In Figure 5.6, we observe that the "anti" curriculum provides a small performance gain compared with the baseline. It is noteworthy that a "random" curriculum's performance is comparable as the regular curriculum.

Find the best hyperparameters to manually design a learning curriculum requires expert knowledge or a great amount of search efforts. In Figure 5.7, we evaluate learning the curriculum automatically with different Teacher-Student Curriculum Learning algorithms. It's shown that when spending the same number of training steps, automatic curricula achieve matching performance compared with the carefully-designed manual curriculum and outperform the baseline.

5.3 Human-Agent Dialogue Collaboration

Inspired by the extensive studies on Human-Agent Collaboration [68] and Human-Robot Collaboration [142, 15], where the human and agent share the same goal and cooperate

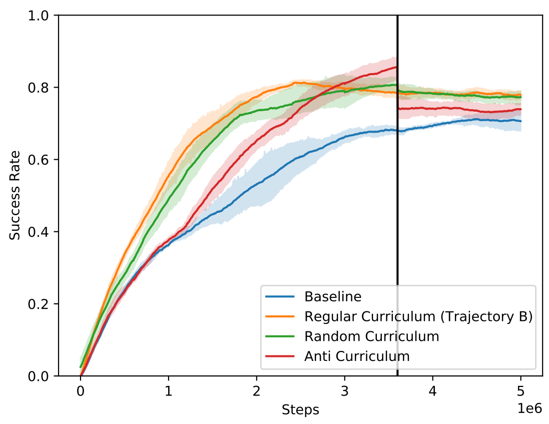


Figure 5.6: Learning curves comparing subtask ordering. Training on the target task begins at the vertical line. We plot the mean and one standard deviation of four runs of random initialization for each agent variant. Regular Curriculum, Random Curriculum and Anti Curriculum all show statistically significant improvement over Baseline ($p < 0.05$ in Welch's t-test).

to achieve the goal, we propose to incorporate a human collaborator via natural language dialogue to assist the agent's policy learning. In the human-agent dialogue, the agents receive the human collaborator's responses in text form, the same way it receives the instruction description. Since human guidance is more expensive to obtain, we specify a limit for the number of dialogue exchanges allowed per episode: *max_dialogue*. The dialogue is solicited by the agent, inquiring hints from the human collaborator to help it complete the instruction.

5.3.1 Dialogue Acts

We consider three types of dialogue exchanges: *inquire best motion action* for direct control over the agent actions, and two high level hints *inquire navigation hint* and *inquire recognition hint*.

Inquire best motion action (motion) Inquiry for the navigation hint asks the human collaborator what motion action is optimal in the current step. For example, the agent's query is "What is the optimal action now?", and the responses can be "Turn left.", "Turn right.", "Move forward.", "Open it.", "Pick it up.", or "Put it down."

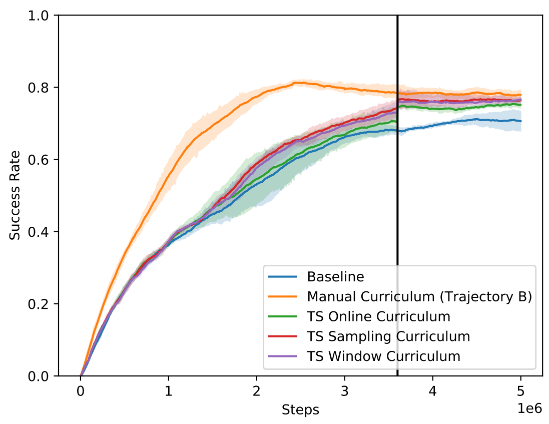


Figure 5.7: Learning curves comparing Teacher-Student Curriculum Learning algorithms. After the vertical line, the agent begins training on the target task. We plot the mean and one standard deviation of four runs of random initialization for each agent variant. Manual Curriculum, TS Sampling Curriculum, and TS Window Curriculum show statistically significant improvement over Baseline, and Manual Curriculum also shows statistically significant improvement TS Online Curriculum ($p < 0.05$ in Welch's t-test).

Inquire navigation hint (nav) Inquiry for the navigation hint asks the human collaborator whether the agent is in the right room where the goal object is located. The purpose of the hint is to help the agent navigate to the right position. For example, the agent's query is "Am I in the right room?", and the responses from the human collaborator can be "Yes, you are in the right place." or "I think you are in the wrong room."

Inquire recognition hint (rec) The goal of the recognition hint is to supplement the agent's object recognition ability. Inquiry for the recognition hint asks the human collaborator whether the goal object is in the agent's current point of view. For example, the agent's query is "Am I looking at the right target?", and the responses from the human collaborator can be "The target object is here." or "That is not what we are looking for."

We use multiple forms of natural language expressions as the human collaborator's responses in our rule-based user simulator.

5.3.2 Experiments

Agent Variants

Baseline Agent The baseline agent is directly trained on the target task from the beginning.

Dialogue Agent (Dial) The Dial agent is directly trained on the target task with human-agent dialogue.

Curriculum Learning Agent (CL) The CL agent is trained according to the curriculum designed by the human collaborator. It is trained for the same number of steps on each of the subtasks in the curriculum, and then trained on the target task.

Curriculum Learning Agent followed by Dialogue Finetuning (CL+DialFinetune) The CL+DialFinetune agent is trained with the same curriculum as the CL agent, except that after training on the target task, the agent's model is finetuned using dialogue with human collaborator in the last few steps.

Curriculum Learning Agent with Dialogue Subtasks (CL+Dial) The CL+Dial agent is trained with the same curriculum similar as the CL agent, except that in every training episode the agent is equipped with human-agent dialogue.

Training details

We set *max_step* to 800, meaning that the episode will be terminated and counted as failure when the agent takes more than 800 steps in an episode. We set *max_dialogue* to be 20.

The word embedding dimension is 32. The GRU networks for encoding instruction description and human response are both single-layer and have output dimension 64. The output channels of the three convolutional layers are 16, 32, and 64 respectively, and their kernels all have size 2x2, followed by ReLU activations. The maxpooling layer after the first convolutional layer's kernel size is 2x2 as well. The fully-connected layer for encoding the remaining dialogue turns has output dimension 20, and uses ReLU activation. The fully-connected

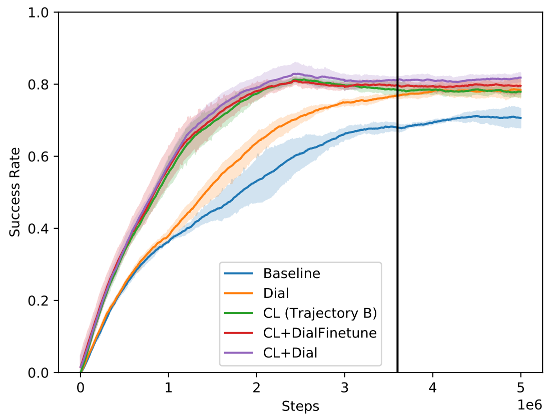


Figure 5.8: Learning curves comparing agent variants. We plot the mean and one standard deviation of four runs of random initialization for each agent variant. Dial, CL, CL+DialFinetune, and CL+Dial all show statistically significant improvement over Baseline; CL+DialFinetune and CL+Dial show statistically significant improvement over Dial; CL+Dial shows statistically significant improvement over CL; CL+Dial also shows statistically significant improvement over CL+DialFinetune ($p < 0.05$ in Welch's t-test).

hidden layer in the critic network and the actor networks for motion and dialogue actions all have dimension 64 with Tanh activations. We set the *entropy_threshold* to 1.5 (maximum entropy of a 6-class categorical distribution is $ln(6) \approx 1.8$).

In the A2C algorithm to learn the agent's policy, we use 0.99 discount factor and $\lambda = 0.95$ for the generalized advantage estimation, and we use RMSprop optimizer with 0.001 learning rate, smoothing factor $\alpha = 0.99$, and $\epsilon = 10^{-8}$.

For curriculum learning, we choose the default curriculum (Trajectory B), comprising five subtasks followed by the target task, starting from the simplest subtask with alternating complexity increase along the instruction variety and spatial layout dimensions, as shown as Figure 5.2. The CL, CL+DialFinetune, and CL+Dial agents are trained on each subtask for 600K steps (total of 3M steps for subtask training), and then trained on the target task. All variants of agents are trained until 5M steps. The finetuning stage of the CL+DialFinetune agent is the last 600K steps (from 4.4M to 5M steps).

Table 5.2: Evaluation results comparing agent variants on 150 evaluation episodes. Success Rate shows the portion of successful episodes. Motion Steps and Dialogue Steps show the average number of motion actions and dialogue actions taken per episode in the successful episodes. The results are computed from four runs of random initialization for each agent variant. * represents statistically significant improvement over Baseline, and † represents statistically significant improvement over Dial ($p < 0.05$ in McNemar's test).

	Baseline	Dial	CL	CL+DialFinetune	CL+Dial
Success Rate	77.2%	84.0%*	84.8%*	87.0%*	87.3%*†
Motion Steps (Success episodes)	99.2	78.4	79.8	73.0	67.4
Dialogue Steps (Success episodes)	-	3.3	-	5.4	2.8

Results

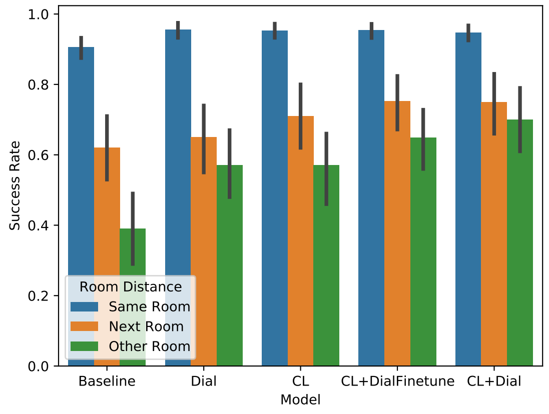
The learning curves comparing the different agent variants are shown in Figure 5.8. It shows that, within the same number of total training steps, the agents which are trained on human-designed curriculum (CL, CL+DialFinetune and CL+Dial) achieve faster performance convergence and higher success rate compared with the baseline.

For evaluation, we focus on the easier episodes where one or more agents succeed and exclude the episodes where none of the agents succeed. The evaluation results are presented in Table 5.2. We present three metrics: Success Rate, Average Motion Steps in successful episodes, and Average Dialogue Steps in successful episodes. The Success Rate metric shows the portion of successful episodes out of all evaluation episodes. The Motion Steps and Dialogue Steps metrics show the average number of motion actions and dialogue actions taken per episode among the successful episodes. We can see that both curriculum learning and human-agent dialogue improve the success rate compared with the baseline. Among the successful episodes, the agents with human-agent dialogue have fewer motion steps, which suggests that responses from human collaborator can help the agent learn policies which have better route efficiency.

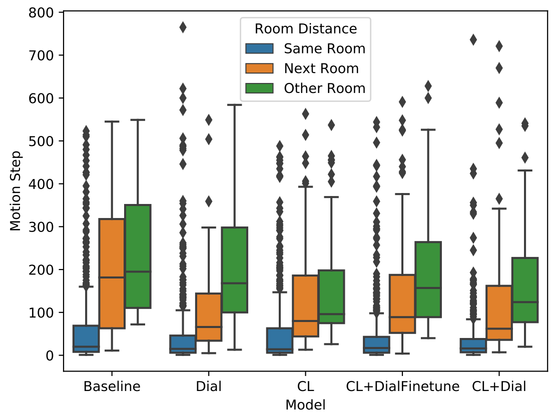
Figure 5.9 shows evaluation results breakdown across task difficulty. We divide the tasks into three levels of difficulties: *Same Room* (easy), *Next Room* (medium), and *Other Room* (hard), representing three types of topology of the agent's initial position and the goal object or appliance's position. Same Room represents the condition when the agent's initial position is in the same room as the goal object or appliance, Next Room signifies the two rooms are connected by a door, and Other Room represents that there are one or two rooms in between the agent's initial room and the goal room. We can observe the trend that the success rate consistently decreases as the task difficulty increases, and the number of motion and dialogue steps increase with the task difficulty.

5.4 Conclusion

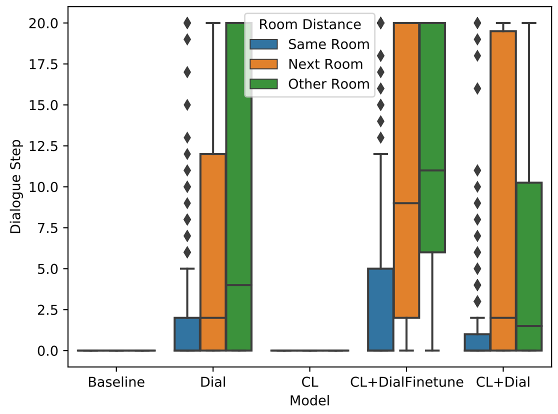
We develop a reinforcement learning framework for learning the vision-grounded instruction following agent which enables Domain Randomization. First, we propose to train a vision-grounded instruction following agent with curriculum learning, using manually-designed curricula and the automatic curricula learned by Teacher-Student Curriculum Learning. In the manual curricula, the subtasks' trajectory, ordering and pacing are carefully designed with human heuristic. We utilize Teacher-Student Curriculum Learning to automatically generate learning curricula. Secondly, the agent can solicit hints from human collaborators in the human-agent dialogue. Experiments show that training with carefully-designed manual curricula and automatically learned curricula is able to accelerate the learning speed on the target task. It's also shown through empirical evaluation that the human-agent dialogue boosts the agent task success rate and the resulted policies lead to more efficient routes.



(A) Success Rate



(B) Motion Steps in Successful Episodes



(C) Dialogue Steps in Successful Episodes

Figure 5.9: Evaluation results breakdown across task difficulty on 150 evaluation episodes. The task difficulty is divided into three levels: Same Room (easy), Next Room (medium), and Other Room (hard). In the subfigures (B) Motion Steps and (C) Dialogue Steps, the box and whisker plots show the min, 25% percentile, 50% percentile, 75% percentile and max of the steps. The results are computed from four runs of random initialization for each agent variant.

Chapter 6

Response Generation for Video Question Answering

In this chapter, we explore the application of a dialogue system for question answering, in particular to answer questions regarding content in videos. Video question answering (VideoQA) systems provide a convenient way for humans to acquire visual information about the environment. If a user wants to obtain information about a dynamic scene, one can simply ask the VideoQA system a question in natural language, and the system generates a natural language answer. The task of a VideoQA dialogue system is described as follows. Given a video as grounding evidence, in each dialogue turn, the system is presented a question and is required to generate an answer in natural language. Figure 6.1 shows an example of multi-turn VideoQA. It is composed of a video clip and a dialogue, where the dialogue contains open-ended question answer pairs regarding the scene in the video. In order to answer the questions correctly, the system needs to be effective at understanding the question, the video and the dialogue context altogether.

Recent work on VideoQA has shown promising performance using multimodal attention fusion for combination of features from different modalities [106, 109, 133, 117]. However,

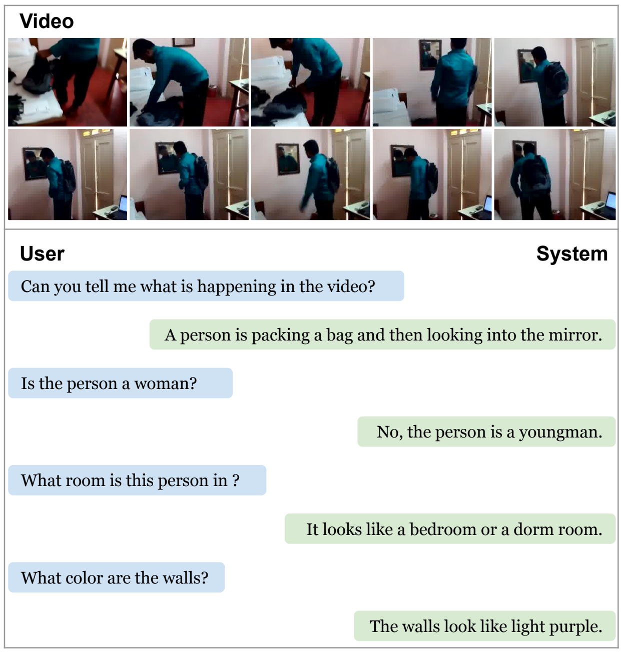

Video
User
System
Can you tell me what is happening in the video?
A person is packing a bag and then looking into the mirror.
Is the person a woman?
No, the person is a youngman.
What room is this person in ?
It looks like a bedroom or a dorm room.
What color are the walls?
The walls look like light purple.

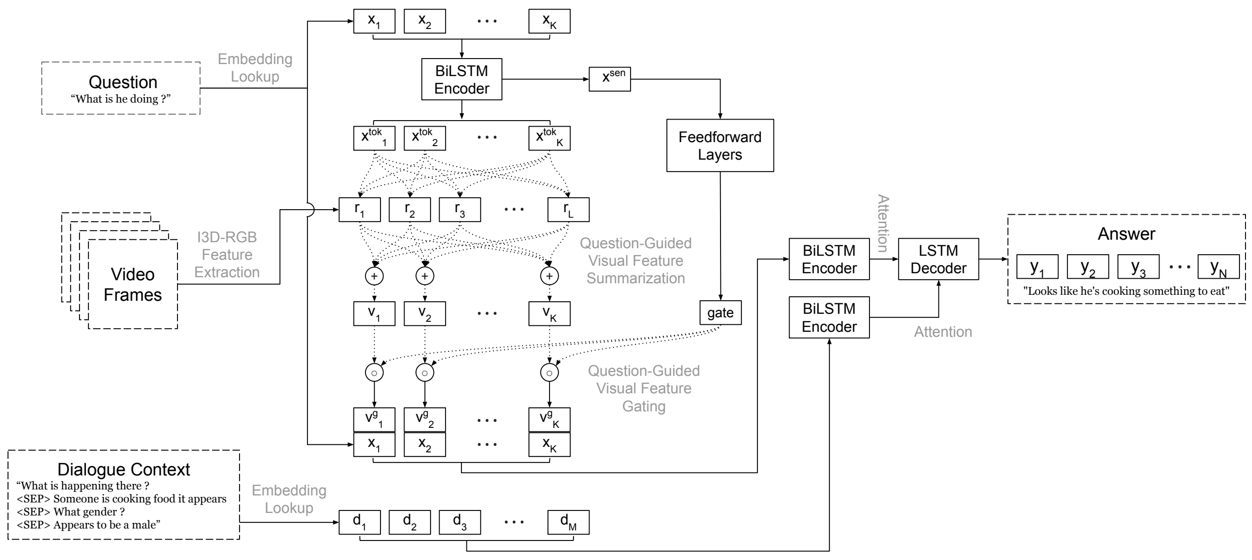
Figure 6.1: An example from the AVSD dataset. Each example contains a video and its associated question answering dialogue regarding the video scene.

one of the challenges is that the length of the video sequence can be very long and the question may concern only a small segment in the video. Therefore, it may be time inefficient to encode the entire video sequence using a recurrent neural network.

6.1 Model

We formulate the multi-turn VideoQA task as follows. Given a sequence of raw video frames $\mathbf{f}$, the embedded question sentence $\mathbf{x} = \{x_1, \ldots, x_K\}$ and the single concatenated embedded sentence of the dialogue context $\mathbf{d} = \{d_1, \ldots, d_M\}$, the output is an answer sentence $\mathbf{y} = \{y_1, \ldots, y_N\}$.

The architecture of our proposed model is illustrated in Figure 6.2. First the Video Frame Feature Extraction Module extracts the I3D-RGB frame features from the video frames. The Question-Guided Video Representation Module takes as input the embedded question sentence and the I3D-RGB features, and generates a compact video representation for each token in the question sentence. In the Video-Augmented Question Encoder, the question



Question
"What is he doing ?"
Embedding Lookup
Video Frames
I3D-RGB Feature Extraction
Dialogue Context
"What is happening there ?
<SEP> Someone is cooking food it appears
<SEP> What gender ?
<SEP> Appears to be a male"
Embedding Lookup
BiLSTM Encoder
Feedforward Layers
gate
Question-Guided Visual Feature Summarization
Question-Guided Visual Feature Gating
BiLSTM Encoder
BiLSTM Encoder
LSTM Decoder
Attention
Attention
Answer
"Looks like he's cooking something to eat"

Figure 6.2: Overview of the proposed multi-turn VideoQA model. First the I3D-RGB frame features are extracted. The question-guided video representation module takes as input the question sentence and the I3D-RGB features, generates a video representation for each token and applies gating using question as guidance. Then the question tokens are augmented by the per-token video representations and encoded by a bidirectional LSTM encoder. Similarly, the dialogue context is encoded by a bidirectional LSTM encoder. Finally, the LSTM answer decoder predicts the answer sequence.

tokens are first augmented by their corresponding per-token video representations and then encoded by a bidirectional LSTM. Similarly, in the Dialogue Context Encoder, the dialogue context is encoded by a bidirectional LSTM. Finally, in the Answer Decoder, the outputs from the Video-Augmented Question Encoder and the Dialogue Context Encoder are used as attention memory for the LSTM decoder to predict the answer sentence. Our encoders and decoder work in the same way as the multi-source sequence-to-sequence models with attention [76, 58].

Video Frame Feature Extraction

We use the I3D-RGB frame features as the visual modality input, which are pre-extracted and provided in the AVSD dataset [135]. Here we briefly describe the I3D-RGB feature extraction process, and we refer the readers to [78] for more details of the I3D model. Two-stream Inflated 3D ConvNet (I3D) is a state-of-the-art action recognition model which operates on video inputs. The I3D model takes as input two streams of video frames: RGB frames and optical flow frames. The two streams are separately passed to a respective 3D ConvNet, which is inflated from 2D ConvNets to incorporate the temporal dimension. Two sequences of spatiotemporal features are produced by the respective 3D ConvNet, which are jointly used to predict the action class. The I3D-RGB features provided in the AVSD dataset are intermediate spatiotemporal representations from the "Mixed_5c" layer of the RGB stream's 3D ConvNet. The AVSD dataset uses the I3D model parameters pre-trained on the Kinetics dataset [85]. To reduce the number of parameters in our model, we use a trainable linear projection layer to reduce the dimensionality of I3D-RGB features from 2048 to 256. Extracted from the video frames $\mathbf{f}$ and projected to a lower dimension, the sequence of dimension-reduced I3D-RGB frame features are denoted by $\mathbf{r} = \{r_1, \ldots, r_L\}$, where $r_i \in \mathrm{R}^{256}, \forall i$.

Question-Guided Video Representation

We use a bidirectional LSTM network to encode the sequence of question token embedding $\mathbf{x} = \{x_1, \ldots, x_K\}$. The token-level intermediate representations are denoted by $\mathbf{x}^{\text{tok}} = \{x_1^{\text{tok}}, \ldots, x_K^{\text{tok}}\}$, and the embedded representation of the entire question is denoted by x^{sen}. These outputs will be used to guide the video representation.

$$\vec{h}_0 = \overleftarrow{h}_{K+1} = \mathbf{0} \tag{6.1}$$

$$\vec{h}_k = \mathrm{LSTM}_{\text{guide}}^{\text{forw}}(x_k, \vec{h}_{k-1}) \tag{6.2}$$

$$\overleftarrow{h}_k = \mathrm{LSTM}_{\text{guide}}^{\text{back}}(x_k, \overleftarrow{h}_{k+1}) \tag{6.3}$$

$$x_k^{\text{tok}} = \vec{h}_k \oplus \overleftarrow{h}_k \tag{6.4}$$

$$\forall k \in \{1, \ldots, K\}$$

$$x^{\text{sen}} = \vec{h}_K \oplus \overleftarrow{h}_1 \tag{6.5}$$

where $\oplus$ denotes vector concatenation; $\overrightarrow{\mathbf{h}}$ and $\overleftarrow{\mathbf{h}}$ represent the local forward and backward LSTM hidden states.

Per-Token Visual Feature Summarization Generally the sequence length of the video frame features is quite large, as shown in Table 6.1. Therefore it is not computationally efficient to encode the video features using a recurrent neural network. We propose to use the attention mechanism to generate a context vector to efficiently summarize the I3D-RGB features. We use the trilinear function [97] as a similarity measure to identify the frames most similar to the question tokens. For each question token x_k, we compute the similarity scores of its encoded representation x_k^{tok} with each of the I3D-RGB features $\mathbf{r}$. The similarity scores $\mathbf{s}_k$ are converted to an attention distribution $\mathbf{w}_k^{\text{att}}$ over the I3D-RGB features by the softmax function. And the video summary v_k corresponding to the question token x_k is defined as the attention weighted linear combination of the I3D-RGB features. We also explored using dot product for computing similarity and empirically found out it yields suboptimal results.

$$
\begin{aligned}
s_{k,l} &= \text{trilinear}(x_k^{\text{tok}}, r_l) && (6.6)\\
&= W_{\text{sim}}[x_k^{\text{tok}} \oplus r_l \oplus (x_k^{\text{tok}} \odot r_l)] && (6.7)\\
&\quad \forall l \in \{1, \ldots, L\}\\
\mathbf{w}_k^{\text{att}} &= \text{softmax}(\mathbf{s}_k) && (6.8)\\
v_k &= \sum_{l=1}^{L} w_{k,l}^{\text{att}}\ r_l && (6.9)\\
\forall k &\in \{1, \ldots, K\}
\end{aligned}
$$

where $\odot$ denotes element-wise multiplication, and W_{sim} is a trainable variable.

Visual Feature Gating Not all details in the video are important for answering a question. Attention helps in discarding the unimportant frames in the time dimension. We propose a gating mechanism which enables us to perform feature selection within each frame. We project the sentence-level question representation through fully-connected layers to generate a gate vector, and perform element-wise multiplication of the video summary for each question token with the gate vector to generate a gated visual summary. We project the sentence-level question representation x^{sen} through fully-connected layers with ReLU non-linearity to generate a gate vector g. For each question token x_k, its corresponding video summary v_k is then multiplied element-wise with the gate vector g to generate a gated visual summary v_k^{g}. We also experimented applying gating on the dimension-reduced I3D-RGB features $\mathbf{r}$, prior to the per-token visual feature summarization step, but it resulted in an inferior performance.

$$
\begin{aligned}
g &= \text{sigmoid}(W_{\text{g},\,1}(\text{ReLU}(W_{\text{g},\,2}x^{\text{sen}} + b_{\text{g},\,2}) + b_{\text{g},\,1}) && (6.10)\\
v_k^g &= v_k \odot g && (6.11)\\
\forall k &\in \{1, \ldots, K\}
\end{aligned}
$$

where $W_{\text{g},\,1}$, $b_{\text{g},\,1}$, $W_{\text{g},\,2}$, $b_{\text{g},\,2}$ are trainable variables.

Video-Augmented Question Encoder

Given the sequence of per-token gated visual summary $\mathbf{v}^{\text{g}} = \{v_1^{\text{g}}, \ldots, v_K^{\text{g}}\}$, we augment the question features by concatenating the embedded question tokens $\mathbf{x} = \{x_1, \ldots, x_K\}$ with their associated per-token video summary. The augmented question features are then encoded using a bidirectional LSTM. The token-level video-augmented question features are denoted by $\mathbf{q}^{\text{tok}} = \{q_1^{\text{tok}}, \ldots, q_K^{\text{tok}}\}$, and the sentence-level feature is denoted by q^{sen}.

$$\vec{h}_0 = \overleftarrow{h}_{K+1} = \mathbf{0} \tag{6.12}$$

$$\vec{h}_k = \text{LSTM}_{\text{ques}}^{\text{forw}}(x_k \oplus v_k^{\text{g}}, \vec{h}_{k-1}) \tag{6.13}$$

$$\overleftarrow{h}_k = \text{LSTM}_{\text{ques}}^{\text{back}}(x_k \oplus v_k^{\text{g}}, \overleftarrow{h}_{k+1}) \tag{6.14}$$

$$q_k^{\text{tok}} = \vec{h}_k \oplus \overleftarrow{h}_k \tag{6.15}$$

$$\forall k \in \{1, \ldots, K\}$$

$$q^{\text{sen}} = \vec{h}_K \oplus \overleftarrow{h}_1 \tag{6.16}$$

where $\vec{\mathbf{h}}$ and $\overleftarrow{\mathbf{h}}$ represent the local forward and backward LSTM hidden states.

Dialogue Context Encoder

Similar to the video-augmented question encoder, we encode the embedded dialogue context tokens $\mathbf{d} = \{d_1, \ldots, d_M\}$ using a bidirectional LSTM. The embedded token-level representations are denoted by $\mathbf{d}^{\text{tok}} = \{d_1^{\text{tok}}, \ldots, d_M^{\text{tok}}\}$.

$$\vec{h}_0 = \overleftarrow{h}_{M+1} = \mathbf{0} \tag{6.17}$$

$$\vec{h}_m = \text{LSTM}_{\text{dial}}^{\text{forw}}(d_m, \vec{h}_{m-1}) \tag{6.18}$$

$$\overleftarrow{h}_m = \text{LSTM}_{\text{dial}}^{\text{back}}(d_m, \overleftarrow{h}_{m+1}) \tag{6.19}$$

$$d_m^{\text{tok}} = \vec{h}_m \oplus \overleftarrow{h}_m \tag{6.20}$$

$$\forall m \in \{1, \ldots, M\}$$

where $\vec{\mathbf{h}}$ and $\overleftarrow{\mathbf{h}}$ represent the local forward and backward LSTM hidden states.

Answer Decoder

The final states of the forward and backward LSTM units of the question encoder are used to initialize the state of answer decoder. Let y_n be the output of the decoder at step n, where $1 \leq n \leq N$, y_0 be the special start of sentence token and y_n^{emb} be the embedded representation of y_n. At a decoder step n, the previous decoder hidden state h_{n-1} is used to attend over $\mathbf{q}^{\text{tok}}$ and $\mathbf{d}^{\text{tok}}$ to get the attention vectors $h_n^{\text{att, q}}$ and $h_n^{\text{att, d}}$ respectively. These two vectors retrieve the relevant features from the intermediate representations of the video-augmented question encoder and the dialogue context encoder, both of which are useful for generating the next token of the answer. At each decoder step, the decoder hidden state h_n is used to generate a distribution over the vocabulary. The decoder output y_n^* is defined to be $\text{argmax}_{y_n}\ p(y_n|y_{\leq n-1})$.

$$h_0 = q^{\text{sen}} \tag{6.21}$$

$$s_{n,k}^{\text{q}} = v_{\text{ans, q}}^{\top} \tanh(W_{\text{ans, q}}[h_{n-1} \oplus q_k^{\text{tok}}]) \tag{6.22}$$

$$\forall k \in \{1, \ldots, K\}$$

$$\mathbf{w}_n^{\text{q}} = \text{softmax}(\mathbf{s}_n^{\text{q}}) \tag{6.23}$$

$$h_n^{\text{att, q}} = \sum_{k=1}^{K} w_{n,k}^{\text{q}}\ q_k^{\text{tok}} \tag{6.24}$$

$$s_{n,m}^{\text{d}} = v_{\text{ans, d}}^{\top} \tanh(W_{\text{ans, d}}[h_{n-1} \oplus d_m^{\text{tok}}]) \tag{6.25}$$

$$\forall m \in \{1, \ldots, M\}$$

$$\mathbf{w}_n^{\text{d}} = \text{softmax}(\mathbf{s}_n^{\text{d}}) \tag{6.26}$$

$$h_n^{\text{att, d}} = \sum_{m=1}^{M} w_{n,m}^{\text{d}}\ d_m^{\text{tok}} \tag{6.27}$$

$$h_n = \text{LSTM}_{\text{ans}}(y_{n-1}^{\text{emb}}, [h_n^{\text{att, q}} \oplus h_n^{\text{att, d}} \oplus h_{n-1}]) \tag{6.28}$$

$$p(y_n|y_{\leq n-1}) = \text{softmax}(W_{\text{ans}} h_n + b_{\text{ans}}) \tag{6.29}$$

$$\forall n \in \{1, \ldots, N\}$$

where $\mathbf{h}$ represents the local LSTM hidden states, and $W_{\text{ans, q}}$, $W_{\text{ans, d}}$, W_{ans}, b_{ans} are trainable variables.

Table 6.1: Statistics of AVSD dataset. We use the official training set, and the public (*i.e.,* prototype) validation and test sets. We also present the average length of the question token sequences and the I3D-RGB frame feature sequences to highlight the importance of time efficient video encoding without using a recurrent neural network. The sequence lengths of the questions and I3D-RGB frame features are denoted by K and L respectively in the model description.

	Training	Validation	Test
# of dialogues	7659	732	733
# of turns	153,180	14,680	14,660
# of words	1,450,754	138,314	138,790
Avg. length of question (K)	8.5	8.4	8.5
Avg. length of I3D-RGB (L)	179.2	173.0	171.3

6.2 Experiments

Dataset

We consider the Audio-Visual Scene-aware Dialog (AVSD) dataset [135] for evaluating our proposed model in single-turn and multi-turn VideoQA. We use the official release of train set for training, and the public (*i.e.,* prototype) validation and test sets for inference. The AVSD dataset is a collection of text-based human-human question answering dialogues based on the video clips from the CHARADES dataset [69]. The CHARADES dataset contains video clips of daily indoor human activities, originally purposed for research in video activity classification and localization. Along with the video clips and associated question answering dialogues, the AVSD dataset also provides the pre-extracted I3D-RGB visual frame features using a pre-trained two-stream inflated 3D ConvNet (I3D) model [78]. The pre-trained I3D model was trained on the Kinetics dataset [85] for human action recognition.

In Table 6.1, we present the statistics of the AVSD dataset. Given the fact that the lengths of the I3D-RGB frame feature sequences are more than 20 times longer than the questions, using a recurrent neural network to encode the visual feature sequences will be very time

Table 6.2: NLG evaluation metrics comparison with existing approaches on AVSD dataset: Naïve Fusion [136, 151], Attentional Fusion [119, 151], Multi-Source Sequence-to-Sequence model [148], Modified Attentional Fusion with Maximum Mutual Information objective [151] and Hierarchical Attention with pre-trained embedding [144], on the AVSD public test set. For each approach, we report its corpus-wide scores on BLEU-1 through BLEU-4, METEOR, ROUGE-L and CIDEr. We report the mean and standard deviation scores of 5 runs using random initialization and early stopping on the public (prototype) validation set.

Single-Turn VideoQA Models	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr
Naïve Fusion	27.7	17.5	11.8	8.3	11.7	28.8	74.0
Multi-source Seq2Seq	-	-	-	8.83	12.43	34.23	95.54
Ours	**29.56**±0.75	**18.60**±0.49	**13.16**±0.33	**9.77**±0.21	**13.19**±0.20	**34.29**±0.19	**101.75**±1.03
Multi-Turn VideoQA Models	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr
Naïve Fusion	27.7	17.6	12.0	8.5	11.8	29.0	76.5
Attentional Fusion	27.6	17.7	12.2	8.7	11.7	29.3	78.7
Modified Attn. Fusion	27.7	17.6	12.0	8.5	11.8	29.0	76.5
+MMI objective	28.3	18.1	12.4	8.9	12.1	29.6	80.5
Hierarchical Attention	29.1	18.6	12.6	9.0	12.7	30.1	82.4
+pre-trained embedding	**30.7**	**20.4**	14.4	10.6	13.6	32.0	99.5
Multi-Source Seq2Seq	-	-	-	10.58	**14.13**	36.54	105.39
Ours	30.52±0.34	20.00±0.20	**14.46**±0.14	**10.93**±0.11	13.87±0.10	**36.62**±0.23	**113.28**±1.37

consuming, as the visual frames are processed sequentially. Our proposed question-guided video representation module summarizes the video sequence efficiently - aggregating the visual features by question-guided attention and weighted summation and performing gating with a question-guided gate vector, both of which can be done in parallel across all frames.

Training Details

We implement our models using the Tensor2Tensor framework [130]. The question and dialogue context tokens are both embedded with the same randomly-initialized word embedding matrix, which is also shared with the answer decoder's output embedding. The dimension of the word embedding is 256, the same dimension to which the I3D-RGB features are transformed. All of our LSTM encoders and decoder have 1 hidden layer. Bahdanau attention mechanism [40] is used in the answer decoder. During training, we apply dropout rate 0.2 in the encoder and decoder cells. We use the ADAM optimizer [41] with $\alpha = 2 \times 10^{-4}, \beta_1 = 0.85, \beta_2 = 0.997, \epsilon = 10^{-6}$, and clip the gradient with L2 norm threshold 2.0 [28]. The models are trained up to 100K steps with early stopping on the validation BLEU-4 score using batch size 1024 on a single GPU. During inference, we use beam search decoding with beam width 3. We experimented with word embedding dimension {256, 512}, dropout rate {0, 0.2}, Luong and Bahdanau attention mechanisms, {1, 2} hidden layer(s) for both encoders and the decoder. We found the aforementioned setting worked best for most models.

Results

Comparison with Existing Methods We evaluate our proposed approach using the same natural language generation evaluation toolkit NLGEval [98] as the previous approaches. The corpus-wide scores of the following unsupervised automated metrics are reported, including BLEU-1 through BLEU-4 [10], METEOR [12], ROUGE-L [11] and CIDEr [51]. The results of our models in comparison with the previous approaches are shown in Table 6.2. We report the mean and standard deviation scores of 5 runs using random initialization and early stopping on the public (prototype) validation set. We apply our model in two scenarios: single-turn and multi-turn VideoQA. The only difference is that in single-turn VideoQA, the dialogue context encoder is excluded from the model.

First we observe that our proposed multi-turn VideoQA model significantly outperforms the single-turn VideoQA model. This suggests that the additional dialogue context input can provide supplementary information from the question and visual features, and thus is helpful for generating the correct answer. Secondly, comparing the single-turn VideoQA models, our approach outperforms the existing approaches across all automatic evaluation metrics. This suggests the effectiveness of our proposed question-guided video representations for VideoQA. When comparing with previous multi-turn VideoQA models, our approach that uses the dialogue context (questions and answers in previous turns) yields state-of-the-art performance on the BLEU-3, BLEU-4, ROUGE-L and CIDEr metrics and competitive results on BLEU-1, BLEU-2 and METEOR. It is worth mentioning that our model does not use pre-trained word embedding or audio features as in the previous hierarchical attention approach [144].

Table 6.3: Ablation study on the AVSD validation set. We observe that the performance degrades when either of both of the question-guided per-token visual feature summarization (TokSumm) and feature gating (Gating) techniques are removed.

Model	BLEU-4	METEOR	ROUGE-L	CIDEr
Ours	10.94	13.73	36.30	111.12
-TokSumm	10.46	13.49	35.81	110.08
-Gating	10.59	13.64	36.11	108.51
-TokSumm-Gating	10.06	13.20	35.35	104.01

Ablation Study and Weights Visualization We perform ablation experiments on the validation set in the multi-turn VideoQA scenario to analyze the effectiveness of the two techniques in the question-guided video representation module. The results are shown in Table 6.3.

Question-Guided Per-Token Visual Feature Summarization (TokSumm) Instead of using token-level question representations $\mathbf{x}^{\text{tok}} = \{x_1^{\text{tok}}, \ldots, x_K^{\text{tok}}\}$ to generate per-token video summary $\mathbf{v} = \{v_1, \ldots, v_K\}$, we experiment with using the sentence-level representation of the question x^{sen} as the query vector to attend over the I3D-RGB visual features to create a visual summary v, and use v to augment each of the question tokens in the video-augmented question encoder.

$$s_l = \text{trilinear}(x^{\text{sen}}, r_l) \tag{6.30}$$

$$\forall l \in \{1, \ldots, L\}$$

$$\mathbf{w}^{\text{att}} = \text{softmax}(\mathbf{s}) \tag{6.31}$$

$$v = \sum_{l=1}^{L} w_l^{\text{att}} r_l \tag{6.32}$$

We observe the performance degrades when the sentence-level video summary is used instead of the token-level video summary.

Figure 6.3 shows an example of the attention weights in the question-guided per-token visual feature summarization. We can see that for different question tokens, the attention weights

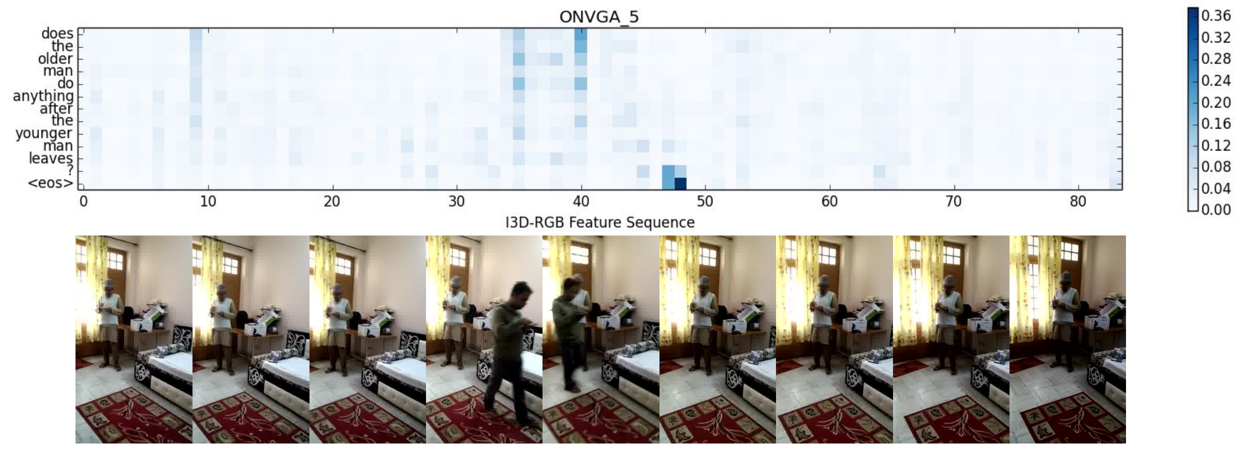


Figure 6.3: Visualization of question-guided per-token visual feature summary weights on a question. Each row represents the attention weights $\mathbf{w}_k^{\text{att}}$ of the corresponding encoded question token x_k^{tok} over the I3D-RGB visual features. We can observe that the attention weights are shifted to focus on the relevant segment of the visual frame features for the question tokens “after the younger man leaves <eos>?"

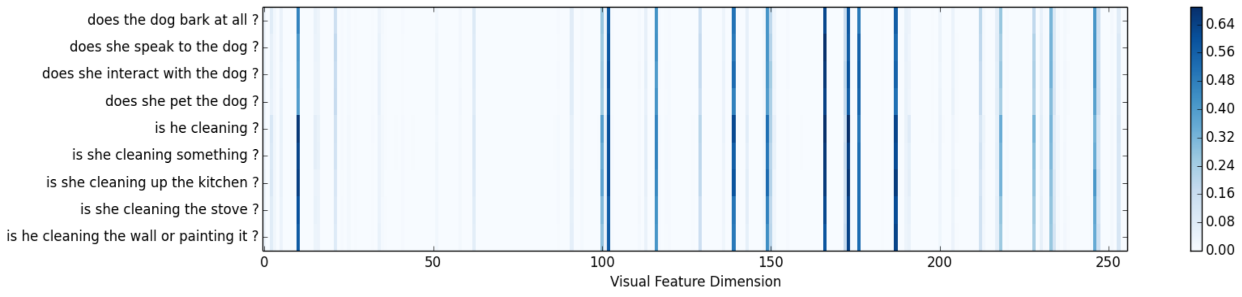


Figure 6.4: Visualization of question-guided gate weights g for some example questions. Across the questions about similar subjects, we observe a similar trend of weight distribution over visual feature dimensions. Conversely, questions about different topics show different gate weights patterns.

are shifted to focus on the different segment in the sequence of the video frame features.

Question-Guided Visual Feature Gating (Gating) We also experiment with using the non-gated token-level video summary $\mathbf{v} = \{v_1, \ldots, v_K\}$ to augment the question information in the video-augmented question encoder. We observe the model's performance declines when the question-guided gating is not applied on the video summary feature. Removing both the per-token visual feature summarization and the gating mechanism results in further degradation in the model performance.

Figure 6.4 illustrates the question-guided gate weights g of several example questions. We observe that the gate vectors corresponding to the questions regarding similar subjects assign weights on similar dimensions of the visual feature. Although many of the visual feature dimensions have low weights across different questions, the feature dimensions of higher gate weights still exhibit certain topic-specific patterns.

6.3 Conclusion

In this chapter, we present an end-to-end trainable model for single-turn and multi-turn VideoQA. Our proposed framework takes the question, I3D-RGB video frame features and dialogue context as input. Using the question information as guidance, the video features are summarized as compact representations to augment the question information, which are jointly used with the dialogue context to generate a natural language answer to the question. Specifically, our proposed question-guided video representation module is able to summarize the video features efficiently for each question token using an attention mechanism and perform feature selection through a gating mechanism. In empirical evaluation, our proposed models for single-turn and multi-turn VideoQA outperform existing approaches on several automatic natural language generation evaluation metrics. Detailed analyses are performed, and it is shown that our model effectively attends to relevant frames in the video feature sequence for summarization, and the gating mechanism shows topic-specific patterns in the feature dimension selection within a frame.

Chapter 7

Conclusion and Future Work

In this thesis, we improve the conventional spoken dialogue systems via learning an end-to-end model which fuses separate modules in the pipeline and leveraging multimodal context from the user and the environment for situated dialogue interaction. First, we study end-to-end and scalable dialogue state tracking. Without using a separate language understanding module, the model directly predicts dialogue states from natural language input and does not require a predefined ontology for slot values. For robust speech recognition in situated dialogues, we propose to recognize users' speech using multimodal input by enhancing our speech recognizer with visual information and speaker embedding to make it robust against adverse acoustic environments. We then train an end-to-end model for vision-grounded instruction following and boost the agent's policy training via curriculum learning and human-agent dialogue. The agent is trained to navigate through the environment, execute the instruction given by the user based on visual perception, and converse with human collaborator to receive feedback. Finally, we consider end-to-end dialogue model for video question answering, where the system's task is to generate natural language answers to questions regarding content in videos by extracting the relevant information from multimodal inputs: textual question and video grounding evidence.

For future work, we propose two directions: Multimodal Learning and Multi-Domain Dialogue.

Multimodal Learning In our current solutions, we leverage visual cues on top of either the speech or textual user input. In some situated dialogue interactions, more modalities may be available at the same time, including audio, text, video, and other contextual information, e.g. date, time, temperature, location. A model that jointly incorporates the multiple present modalities from user and environmental context will be the most useful.

Multi-Domain Dialogue Our current solutions are trained specifically for their individual domain, e.g. vision-grounded instruction following and video question answering, and the models are trained to learn domain specific skills. A future direction is to perform knowledge transfer on multiple domain datasets in the training process and learn a unified multimodal model across the different domains of situated dialogue interactions.

BIBLIOGRAPHY

[1] W. L. Taylor, “"cloze procedure": A new tool for measuring readability,” *Journalism Bulletin*, 1953.

[2] H. McGurk and J. MacDonald, “Hearing lips and seeing voices,” *Nature*, 1976.

[3] L. R. Bahl, P. F. Brown, P. V. De Souza, and R. L. Mercer, “Maximum mutual information estimation of hidden markov model parameters for speech recognition,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1986.

[4] J. S. Bridle and S. J. Cox, “Recnorm: Simultaneous normalisation and classification applied to speech recognition,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 1990.

[5] C. Bregler and Y. Konig, “"Eigenlips" for robust speech recognition,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1994.

[6] P. Comon, “Independent component analysis, a new concept?” *Signal processing*, 1994.

[7] S. Choi, H. Hong, H. Glotin, and F. Berthommier, “Multichannel signal separation for cocktail party speech recognition: A dynamic recurrent network,” *Neurocomputing*, 2002.

[8] G.-J. Jang and T.-W. Lee, “A probabilistic approach to single channel blind signal separation,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2002.

[9] A. V. Nefian, L. Liang, X. Pi, X. Liu, and K. Murphy, “Dynamic bayesian networks for audio-visual speech recognition,” *EURASIP Journal on Advances in Signal Processing*, 2002.

[10] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: A method for automatic evaluation of machine translation,” in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.

[11] C.-Y. Lin and F. J. Och, “Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics,” in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2004.

[12] S. Banerjee and A. Lavie, “Meteor: An automatic metric for mt evaluation with improved correlation with human judgments,” in *ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005.

[13] M. Cooke, J. Barker, S. Cunningham, and X. Shao, “An audio-visual corpus for speech perception and automatic speech recognition,” *Journal of the Acoustical Society of America*, 2006.

[14] M. N. Schmidt and R. K. Olsson, “Single-channel speech separation using sparse non-negative matrix factorization,” in *INTERSPEECH*, 2006.

[15] A. Bauer, D. Wollherr, and M. Buss, “Human–robot collaboration: A survey,” *International Journal of Humanoid Robotics*, 2008.

[16] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *International Conference on Machine Learning (ICML)*, 2009.

[17] S. R. Branavan, H. Chen, L. S. Zettlemoyer, and R. Barzilay, “Reinforcement learning for mapping instructions to actions,” Annual Meeting of the Association for Computational Linguistics (ACL), 2009.

[18] J. H. McDermott, “The cocktail party problem,” *Current Biology*, 2009.

[19] A. Vogel and D. Jurafsky, “Learning to follow navigational directions,” in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[20] S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, “The hidden information state model: A practical framework for pomdp-based spoken dialogue management,” *Computer Speech & Language*, 2010.

[21] O. Chapelle and L. Li, “An empirical evaluation of thompson sampling,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[22] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2011.

[23] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal deep learning,” in *International Conference on Machine Learning (ICML)*, 2011.

[24] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition,” *IEEE Signal Processing Magazine*, 2012.

[25] O. Abdel-Hamid and H. Jiang, “Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.

[26] ——, “Rapid and effective speaker adaptation of convolutional neural network based models for speech recognition,” in *INTERSPEECH*, 2013.

[27] J. Barker, E. Vincent, N. Ma, H. Christensen, and P. Green, “The pascal chime speech separation and recognition challenge,” *Computer Speech & Language*, 2013.

[28] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International Conference on Machine Learning (ICML)*, 2013.

[29] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, “Speaker adaptation of neural network acoustic models using i-vectors.,” in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2013.

[30] S. Young, M. Gašić, B. Thomson, and J. D. Williams, "POMDP-based statistical spoken dialog systems: A review," *Proceedings of the IEEE*, 2013.

[31] R. Doddipatla, M. Hasan, and T. Hain, "Speaker dependent bottleneck layer training for speaker adaptation in automatic speech recognition.," in *INTERSPEECH*, 2014.

[32] M. Henderson, B. Thomson, and J. D. Williams, "The second dialog state tracking challenge," in *Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014.

[33] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *INTERSPEECH*, 2014.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.

[35] A. Rousseau, P. Deléglise, Y. Esteve, *et al.*, "Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks.," in *International Conference on Language Resources and Evaluation (LREC)*, 2014.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.

[37] P. Xu and R. Sarikaya, "Targeted feature dropout for robust slot filling in natural language understanding," in *Annual Conference of the International Speech Communication Association*, 2014.

[38] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2014.

[39] M. J. Alam, V. Gupta, P. Kenny, and P. Dumouchel, "Speech recognition in reverberant and noisy environments employing multiple feature extractors and i-vector speaker adaptation," *EURASIP Journal on Advances in Signal Processing*, 2015.

[40] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[41] D. Kingma and J. Ba, "A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[42] Y. Miao, H. Zhang, and F. Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2015.

[43] Y. Mroueh, E. Marcheret, and V. Goel, "Deep multimodal learning for audio-visual speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[44] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Applied Intelligence*, 2015.

[45] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[46] M. Ren, R. Kiros, and R. Zemel, "Exploring models and data for image question answering," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[47] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[48] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[49] S. Sukhbaatar, J. Weston, R. Fergus, *et al.*, "End-to-end memory networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[50] F. de la Torre, W.-S. Chu, X. Xiong, F. Vicente, X. Ding, and J. Cohn, "Intraface," in *International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, 2015.

[51] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[52] L. Yu, E. Park, A. C. Berg, and T. L. Berg, "Visual Madlibs: Fill in the blank description generation and question answering," in *International Conference on Computer Vision (ICCV)*, 2015.

[53] N. Zehngut, "Audio visual speech recognition using facial landmark based video frontalization," unpublished technical report, 2015.

[54] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *International Conference on Computer Vision (ICCV)*, 2015.

[55] C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, *et al.*, "Deepmind lab," *Computing Research Repository*, vol. arXiv:1612.03801, 2016. [Online]. Available: `http://arxiv.org/abs/1612.03801`.

[56] A. Biswas, P. Sahu, and M. Chandra, "Multiple cameras audio visual speech recognition using active appearance model visual features in car environment," *International Journal of Speech Technology*, 2016.

[57] G.-L. Chao, W. Chan, and I. Lane, "Speaker-targeted audio-visual models for speech recognition in cocktail-party environments," in *INTERSPEECH*, 2016.

[58] O. Firat, K. Cho, and Y. Bengio, "Multi-way, multilingual neural machine translation with a shared attention mechanism," in *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[59] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, 2016.

[60] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, “MS-Celeb-1M: A dataset and benchmark for large-scale face recognition,” in *European Conference on Computer Vision (ECCV)*, 2016.

[61] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[62] S. Kim, B. Raj, and I. Lane, “Environmental noise embeddings for robust speech recognition,” *Computing Research Repository*, vol. arXiv:1601.02553, 2016. [Online]. Available: `http://arxiv.org/abs/1601.02553`.

[63] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, “Ask me anything: Dynamic memory networks for natural language processing,” in *International Conference on Machine Learning (ICML)*, 2016.

[64] Y. Liu and K. Kirchhoff, “Novel front-end features based on neural graph embeddings for DNN-HMM and LSTM-CTC acoustic modeling.,” in *INTERSPEECH*, 2016.

[65] J. Lu, J. Yang, D. Batra, and D. Parikh, “Hierarchical question-image co-attention for visual question answering,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[66] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, “Purely sequence-trained neural networks for ASR based on lattice-free MMI,” in *INTERSPEECH*, 2016.

[67] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[68] S. D. Ramchurn, F. Wu, W. Jiang, J. E. Fischer, S. Reece, S. Roberts, T. Rodden, C. Greenhalgh, and N. R. Jennings, “Human–agent collaboration for disaster response,” *Autonomous Agents and Multi-Agent Systems*, 2016.

[69] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, “Hollywood in homes: Crowdsourcing data collection for activity understanding,” in *European Conference on Computer Vision (ECCV)*, 2016.

[70] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, “Deep neural network-based speaker embeddings for end-to-end speaker verification,” in *Spoken Language Technology Workshop (SLT)*, 2016.

[71] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler, “MovieQA: Understanding stories in movies through question-answering,” in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[72] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *Computing Research Repository*, vol. arXiv:1609.08144, 2016. [Online]. Available: `http://arxiv.org/abs/1609.08144`.

[73] C. Xiong, S. Merity, and R. Socher, "Dynamic memory networks for visual and textual question answering," in *International Conference on Machine Learning (ICML)*, 2016.

[74] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *European Conference on Computer Vision (ECCV)*, 2016.

[75] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[76] B. Zoph and K. Knight, "Multi-source neural translation," in *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[77] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Parikh, and D. Batra, "VQA: Visual question answering," *International Journal of Computer Vision (IJCV)*, 2017.

[78] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the Kinetics dataset," in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[79] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from RGB-D data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[80] X. Cui, V. Goel, and G. Saon, "Embedding-based speaker adaptive training of deep neural networks," in *INTERSPEECH*, 2017.

[81] A. Das, S. Kottur, K. Gupta, A. Singh, D. Yadav, J. M. Moura, D. Parikh, and D. Batra, "Visual dialog," in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[82] K. M. Hermann, F. Hill, S. Green, F. Wang, R. Faulkner, H. Soyer, D. Szepesvari, W. M. Czarnecki, M. Jaderberg, D. Teplyashin, *et al.*, "Grounded language learning in a simulated 3d world," *Computing Research Repository*, vol. arXiv:1706.06551, 2017. [Online]. Available: `http://arxiv.org/abs/1706.06551`.

[83] Y. Jang, Y. Song, Y. Yu, Y. Kim, and G. Kim, "TGIF-QA: Toward spatio-temporal reasoning in visual question answering," in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[84] R. Jia, L. Heck, D. Hakkani-Tür, and G. Nikolov, "Learning concepts through conversations in spoken dialogue systems," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[85] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, "The kinetics human action video dataset," *Computing Research Repository*, vol. arXiv:1705.06950, 2017. [Online]. Available: `http://arxiv.org/abs/1705.06950`.

[86] V. Kazemi and A. Elqursh, "Show, ask, attend, and answer: A strong baseline for visual question answering," *Computing Research Repository*, vol. arXiv:1704.03162, 2017. [Online]. Available: `http://arxiv.org/abs/1704.03162`.

[87] K.-M. Kim, M.-O. Heo, S.-H. Choi, and B.-T. Zhang, "DeepStory: Video story QA by deep embedded memory networks," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2017.

[88] B. King, I.-F. Chen, Y. Vaizman, Y. Liu, R. Maas, S. H. K. Parthasarathi, and B. Hoffmeister, "Robust speech recognition via anchor word representations," in *INTERSPEECH*, 2017.

[89] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, *et al.*, "Visual Genome: Connecting language and vision using crowdsourced dense image annotations," *International Journal of Computer Vision (IJCV)*, 2017.

[90] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: An end-to-end neural speaker embedding system," *Computing Research Repository*, vol. arXiv:1705.02304, 2017. [Online]. Available: `http://arxiv.org/abs/1705.02304`.

[91] J. Libovický and J. Helcl, "Attention strategies for multi-source sequence-to-sequence learning," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[92] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[93] J. Mun, P. Hongsuck Seo, I. Jung, and B. Han, "MarioQA: Answering questions by watching gameplay videos," in *International Conference on Computer Vision (ICCV)*, 2017.

[94] M. Muthukrishnan, A. Tomkins, L. Heck, A. Geramifard, and D. Agarwal, "The future of artificially intelligent assistants," in *SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2017.

[95] S. Na, S. Lee, J. Kim, and G. Kim, "A read-write memory network for movie story understanding," in *International Conference on Computer Vision (ICCV)*, 2017.

[96] A. Rastogi, D. Hakkani-Tür, and L. Heck, "Scalable multi-domain dialogue state tracking," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

[97] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," in *International Conference on Learning Representations (ICLR)*, 2017.

[98] S. Sharma, L. El Asri, H. Schulz, and J. Zumer, "Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation," *Computing Research Repository*, vol. arXiv:1706.09799, 2017. [Online]. Available: `http://arxiv.org/abs/1706.09799`.

[99] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *INTERSPEECH*, 2017.

[100] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning.," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[101] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[102] Z. Tüske, W. Michel, R. Schlüter, and H. Ney, "Parallel neural network features for improved tandem acoustic modeling," in *INTERSPEECH*, 2017.

[103] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[104] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. R. Barahona, P.-H. Su, S. Ultes, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," in *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[105] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[106] D. Xu, Z. Zhao, J. Xiao, F. Wu, H. Zhang, X. He, and Y. Zhuang, "Video question answering via gradually refined attention over appearance and motion," in *International Conference on Multimedia*, 2017.

[107] Y. Ye, Z. Zhao, Y. Li, L. Chen, J. Xiao, and Y. Zhuang, "Video question answering via attribute-augmented attention network learning," in *SIGIR Conference on Research and Development in Information Retrieval*, 2017.

[108] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[109] K.-H. Zeng, T.-H. Chen, C.-Y. Chuang, Y.-H. Liao, J. C. Niebles, and M. Sun, "Leveraging video descriptions to learn video question answering," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[110] K. Žmolíková, M. Delcroix, K. Kinoshita, T. Higuchi, A. Ogawa, and T. Nakatani, "Learning speaker representation for neural network based multichannel speaker extraction," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

[111] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[112] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[113] M. Chevalier-Boisvert, L. Willems, and S. Pal, *Minimalistic gridworld environment for openai gym*, `https://github.com/maximecb/gym-minigrid`, 2018.

[114] M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[115] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein, "Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation," *ACM Transactions on Graphics (TOG)*, 2018.

[116] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, "Speaker-follower models for vision-and-language navigation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[117] J. Gao, R. Ge, K. Chen, and R. Nevatia, "Motion-appearance co-memory networks for video question answering," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[118] R. Goel, S. Paul, T. Chung, J. Lecomte, A. Mandal, D. Hakkani-Tur, and A. A. AI, "Flexible and scalable state tracking framework for goal-oriented dialogue systems," in *NeurIPS Conversational AI workshop*, 2018.

[119] C. Hori, H. Alamri, J. Wang, G. Winchern, T. Hori, A. Cherian, T. K. Marks, V. Cartillier, R. G. Lopes, A. Das, *et al.*, "End-to-end audio visual scene-aware dialog using multimodal attention-based video features," *Computing Research Repository*, vol. arXiv:1806.08409, 2018. [Online]. Available: `http://arxiv.org/abs/1806.08409`.

[120] J. Lei, L. Yu, M. Bansal, and T. Berg, "TVQA: Localized, compositional video question answering," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[121] C. Ma, C. Shen, A. Dick, Q. Wu, P. Wang, A. van den Hengel, and I. Reid, "Visual question answering with memory-augmented networks," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[122] D. Misra, A. Bennett, V. Blukis, E. Niklasson, M. Shatkhin, and Y. Artzi, "Mapping instructions to actions in 3d environments with visual goal prediction," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[123] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *International Conference on Robotics and Automation (ICRA)*, 2018.

[124] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[125] Y. Qian, X. Chang, and D. Yu, "Single-channel multi-talker speech recognition with permutation invariant training," *Speech Communication*, 2018.

[126] A. Rastogi, R. Gupta, and D. Hakkani-Tur, "Multi-task learning for joint language understanding and dialogue state tracking," in *Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2018.

[127] L. Ren, K. Xie, L. Chen, and K. Yu, "Towards universal dialogue state tracking," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[128] P. Shah, D. Hakkani-Tür, G. Tür, A. Rastogi, A. Bapna, N. Nayak, and L. Heck, "Building a conversational agent overnight with dialogue self-play," *Computing Research Repository*, vol. arXiv:1801.04871, 2018. [Online]. Available: `http://arxiv.org/abs/1801.04871`.

[129] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[130] A. Vaswani, S. Bengio, E. Brevdo, F. Chollet, A. Gomez, S. Gouws, L. Jones, Ł. Kaiser, N. Kalchbrenner, N. Parmar, *et al.*, "Tensor2Tensor for neural machine translation," in *Conference of the Association for Machine Translation in the Americas*, 2018.

[131] P. Xu and Q. Hu, "An end-to-end approach for handling unknown slot values in dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[132] H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba, "The sound of pixels," in *European Conference on Computer Vision (ECCV)*, 2018.

[133] Z. Zhao, Z. Zhang, S. Xiao, Z. Yu, J. Yu, D. Cai, F. Wu, and Y. Zhuang, "Open-ended long-form video question answering via adaptive hierarchical reinforced networks," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2018.

[134] V. Zhong, C. Xiong, and R. Socher, "Global-locally self-attentive encoder for dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[135] H. Alamri, V. Cartillier, A. Das, J. Wang, S. Lee, P. Anderson, I. Essa, D. Parikh, D. Batra, A. Cherian, T. K. Marks, and C. Hori, "Audio visual scene-aware dialog," in *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[136] H. Alamri, C. Hori, T. K. Marks, D. Batra, and D. Parikh, "Audio visual scene-aware dialog (avsd) track for natural language generation in dstc7," in *DSTC7 at AAAI 2019 Workshop*, 2019.

[137] G.-L. Chao, C. C. Hu, B. Liu, J. P. Shen, and I. Lane, "Audio-visual TED corpus: Enhancing the TED-LIUM corpus with facial information, contextual text and object recognition," in *UbiComp Workshop on Continual and Multimodal Learning for Internet of Things*, 2019.

[138] G.-L. Chao, J. P. Shen, and I. Lane, "Deep speaker embedding for speaker-targeted automatic speech recognition," in *International Conference on Natural Language Processing and Information Retrieval (NLPIR)*, 2019.

[139] M. Chevalier-Boisvert, D. Bahdanau, S. Lahlou, L. Willems, C. Saharia, T. H. Nguyen, and Y. Bengio, "Babyai: A platform to study the sample efficiency of grounded language learning," in *International Conference on Learning Representations (ICLR)*, 2019.

[140] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[141] C. Fan, X. Zhang, S. Zhang, W. Wang, C. Zhang, and H. Huang, "Heterogeneous memory enhanced multimodal attention model for video question answering," in *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[142] G. Hoffman, "Evaluating fluency in human–robot collaboration," *IEEE Transactions on Human-Machine Systems*, 2019.

[143] S. H. Kumar, E. Okur, S. Sahay, J. J. A. Leanos, J. Huang, and L. Nachman, "Context, attention and audio feature explorations for audio visual scene-aware dialog," in *DSTC7 at AAAI 2019 workshop*, 2019.

[144] H. Le, S. C. Hoi, D. Sahoo, and N. F. Chen, "End-to-end multimodal dialog systems with hierarchical multimodal attention on video features," in *DSTC7 at AAAI 2019 workshop*, 2019.

[145] K.-Y. Lin, C.-C. Hsu, Y.-N. Chen, and L.-W. Ku, "Entropy-enhanced multimodal attention model for scene-aware dialogue generation," in *DSTC7 at AAAI 2019 workshop*, 2019.

[146] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher-student curriculum learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[147] D. T. Nguyen, S. Sharma, H. Schulz, and L. El Asri, "From film to video: Multi-turn question answering with multi-modal context," in *DSTC7 at AAAI 2019 workshop*, 2019.

[148] R. Pasunuru and M. Bansal, "Dstc7-avsd: Scene-aware video-dialogue systems with dual attention," in *DSTC7 at AAAI 2019 workshop*, 2019.

[149] R. Sanabria, S. Palaskar, and F. Metze, "Cmu sinbad's submission for the dstc7 avsd challenge," in *DSTC7 at AAAI 2019 workshop*, 2019.

[150] Y.-T. Yeh, T.-C. Lin, H.-H. Cheng, Y.-H. Deng, S.-Y. Su, and Y.-N. Chen, "Reactive multi-stage feature fusion for multimodal dialogue modeling," in *DSTC7 at AAAI 2019 Workshop*, 2019.

[151] B. Zhuang, W. Wang, and T. Shinozaki, "Investigation of attention-based multimodal fusion and maximum mutual information objective for dstc7 track3," in *DSTC7 at AAAI 2019 workshop*, 2019.

[152] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks," in *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[153] *Amazon Alexa*, `https://developer.amazon.com/alexa`, Accessed: 2020-12-10.

[154] *Google Assistant*, `https://assistant.google.com`, Accessed: 2020-12-10.

[155] *Sox - sound exchange*, Online; accessed 2021-06-07. [Online]. Available: `http://sox.sourceforge.net`.