
Building Goal-Oriented Dialogue Systems with Situated Visual Context

Sanchit Agarwal Jan Jezabek Arijit Biswas Emre Barut Shuyang Gao

Tagyoung Chung

Amazon Alexa AI

{agsanchi, jzabek, barijit, ebarut, shuyag, tagyoung}@amazon.com

Abstract

Most popular goal-oriented dialogue agents are capable of understanding the conversational context. However, with the surge of virtual assistants with screen, the next generation of agents are required to also understand screen context in order to provide a proper interactive experience, and better understand users' goals. In this paper, we propose a novel multimodal conversational framework, where the dialogue agent's next action and their arguments are derived jointly conditioned both on the conversational and the visual context. Specifically, we propose a new model, that can reason over the visual context within a conversation and populate API arguments with visual entities given the user query. Our model can recognize visual features such as color and shape as well as the metadata based features such as price or star rating associated with a visual entity. In order to train our model, due to a lack of suitable multimodal conversational datasets, we also propose a novel multimodal dialog simulator to generate synthetic data and also collect realistic user data from MTurk to improve model robustness. The proposed model achieves a reasonable model accuracy, without high inference latency. We also demonstrate the proposed approach in a prototypical furniture shopping experience for a multimodal virtual assistant.

1 Introduction

Goal-oriented dialogue systems enable users to complete specific goals such as booking a flight. The user informs their intent and the agent will ask for the slot values such as time and number of people before booking. The traditional goal-oriented systems are usually aware of the conversational context between the chatbot and the user (1; 20; 11; 16), where users can refer to entities from the context. However, with the recent surge of virtual assistants with screens (e.g., Echo Show), the screen becomes an additional source of context that is vital in fulfilling the users' goal. For example, when a user browsing for chairs on Echo Show says, "*What is the price of the black checkered one?*", the virtual assistant needs to identify the right product based on its visual characteristics and then respond with the price. Moreover, users may want to visually compare items from the current screen with those shown on earlier screens by asking questions such as "*Is this cheaper than the green t-shirt you showed earlier?*". The dialogue agent needs to understand the multimodal context in order to resolve the relevant entities and then compare their prices. Since, most current dialogue systems do not take the visual context into account, they are unable to fulfill such goals.

In this paper, we propose a novel multimodal goal-oriented dialogue system that can reason over the current and historical screen context as well as the conversational context in order to complete users' goal. An example conversation with an agent built using our proposed method is demonstrated in

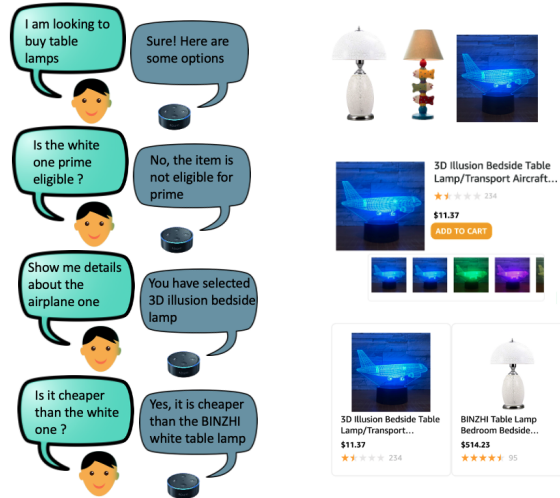


Figure 1: Example Multimodal Conversation.

Figure 1. Users can refer to on-screen visual entities by attributes such as color (e.g., “the white one”) or shape (e.g., “the one shaped like an airplane”). They can also refer to the visual elements using associated metadata such as rating or prime-eligibility (e.g., “five-star one”, “prime-eligible one”) or by their relative positions (e.g., “middle one”). They can also refer to visual entities from historical context. The dialogue agent can automatically resolve the relevant visual entities from the context, determine the next action, and fill the slots. For example, “Zoom on the red-striped shirt” calls the *ZOOM* action with the argument being the entity representing the “red-striped shirt” in the visual context. We enable multimodal understanding capabilities by introducing a new visual grounding model that can reason over the visual context given the user query and populate API arguments with visual entities¹.

Our main contributions are as follows:

1. We propose a novel multimodal conversational system that considers screen context, in addition to dialogue context, while deciding the agent’s next action.
2. The proposed visual grounding model takes both metadata and images as input allowing it to reason over metadata and visual information.
3. Our solution encodes the user query and each visual entities and then computes the similarity between them. To improve the visual entity encoding, we introduced query-guided attention and entity self-attention layers.
4. For training data, we create a multimodal dialogue simulator. In addition, to capture real user behavior and collect more realistic data, we run a large scale Mechanical Turk (MTurk) survey (~95K samples).
5. Our model is able to achieve ~85% accuracy on difficult crowd-sourced test set and we perform a thorough ablation study and show the contribution of each factor in model.

2 Related Works

In essence, our problem requires a combination of (i) a visual grounding solution that can exploit additional metadata to select an object based on a referring expression and (ii) a multimodal dialogue solution that can utilize dialogue and visual history to label the user intent and perform the implied tasks on a shared interactive medium. We review the related work in each of these topics separately.

Visual Grounding Until recently, state-of-the-art visual grounding models relied on combining the data from two modalities -visual and natural language- using various early fusion techniques (22; 2). Over the last couple of years, these methods have been surpassed by transformer-based solutions

¹A video demo of our skill can be found at <https://tinyurl.com/multimodal-conversations>

that are pre-trained on various combinations of image and text masking tasks (17; 12; 10; 21; 6). Our setting differs significantly from the commonly studied setup due to (i) model having access to the exact location of each object, which removes the need for object detection -a fundamental component of visual grounding- and (ii) additional metadata about the entities on the screen, which might contain both visual (e.g., color, material) and non-visual (e.g., prime eligibility, brand name) information about these entities. Thus, we cannot utilize other visual grounding solutions out of the box, and propose an attention based metadata injection mechanism that we present in Section 4.

Task-oriented Multimodal Dialogue Systems Our visual grounding solution enables selection of an item based on visual cues, but a multimodal shopping experience involves tasks that require higher order reasoning, such as co-referencing and disambiguation. Compared to the rich literature in multimodality, the research multimodal dialogues is rather sparse. One line of work considers visual question answering (VQA) tasks over a dialogue, (3; 8; 4; 18). Another line of work are the fruits of the SIMMC challenge (14) which considers a multimodal conversational shopping experience that involves an experience that is close to ours. In this line of work, the literature relies on large scale transformers (e.g., GPT-2 (15) and BART (9)) that generates a system action and a response based on the current and past utterances (i.e., the dialogue context) and the multimodal context (7). Our approach involves building a visual understanding solution on top of an established conversational AI solution (1) and thus we cannot rely on these generation based solutions.

3 Data

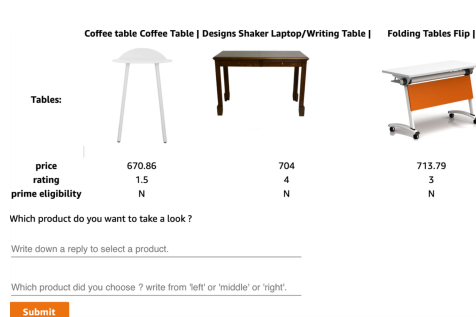
One of the major bottlenecks for building a multimodal goal oriented system is lack of publicly available datasets. Although there are few, none of them fully capture all variations that we want to support in our framework. For example, the SIMMC challenge (14) dataset is not usable because (i) out of all the APIs in this dataset, only one (*FocusOnFurniture*) requires visual context for correct prediction and that is referenced with an ordinal number most of the time. It does not contain examples with visual attributes like pattern or shape, except a few utterances for referencing based on color; (ii) DSTC9 API annotations only provide the dominant API even though there could have been multiple API calls between two turns. This issue, if not fixed, can lead to model confusion.

Due to the lack of readily available datasets, we explore two different approaches for dataset creation. The first approach involves a dialogue simulation technique with multimodal context that primarily uses the available catalogues and associated metadata to create a domain-specific multimodal conversational dataset. This approach enables creating seed data for model training for any new domain without much effort. However, the models built may not be very robust. Hence, we complement our data with a second approach, where we design an MTurk collection pipeline to capture the actual user behavior while interacting with visual context. This allows us to collect a more realistic dataset with more language variations.

3.1 Multimodal Dialogue Simulator

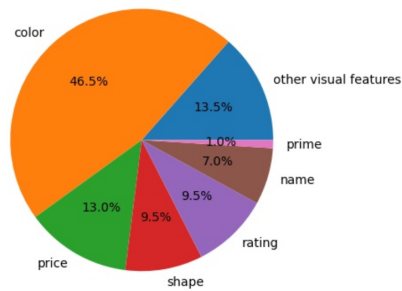
To ensure a large degree of variation in the training data, we implement a multimodal dialogue simulator. It can generate a large number of synthetic conversations that capture numerous variations in how a visual entity is selected (e.g., by name, position) and what operations are applied to it (e.g., selection, zoom-in). It also generates variations in the visual contexts and examples of interactions spanning multiple turns (e.g., comparison, intent carryover). The simulator works by randomly generating screen layouts with visual entities and subsequently simulating user interactions with entities on the screen or with other entities mentioned recently. The method of operation is as follows:

1. Randomly select up to two entities associated with components on the current simulated screen.
2. Randomly decide the method by which to refer to the chosen entities (e.g., color, position, item type). Generate a phrase that references the object (e.g., “*the one on the right*”, “*the blue one*”).
3. Check if the phrase uniquely identifies each of the entities among the ones visible on screen. If not, repeat step 2.
4. Randomly pick an action or a sequence of actions (e.g., in the case of comparison or intent carryover) to be performed on the entities. Generate a simulated sequence of utterances, for example “*Show me the left one. Is it cheaper than the red one?*”.
5. Update the visual state based on the execution of the selected actions.



(a) Amazon Mechanical Turk Interface

sentence type annotations for production selection



(b) MTurk Data Distribution

Figure 2: MTurk Multimodal Data Collection

6. Repeat until MAX_DIALOGUE_LENGTH is reached.

Simulated interactions include inquiring about properties of objects (e.g., “What is the material for the left one?”), comparing objects, as well as taking actions on them (e.g., “Add the red one to the cart”). In each interaction, objects can be referenced by metadata (e.g., name, price, rating) or by their location on screen (e.g., “left one”, “last one”). Besides one-turn requests, simulator also generates multi-turn conversational features like anaphoric referencing (e.g., “What is the price of the rightmost one? Add it to my cart”) and intent-carryover (e.g., “What is the rating of the blue one? How about the green one?”) leading to a diverse conversational dataset. Implementing the dialogue simulator also allows us to quickly bootstrap a basic model without any additional training data. One limitation of simulator, however, is that it can only generate data based on metadata attributes present in catalogue. As a result, referencing based on shapes or patterns cannot be generated as these attributes are typically not present in catalogue. We rely on MTurk collection to mitigate this problem, which we discuss next.

3.2 MTurk Collection

The variations covered in the simulated data may not be enough to build very robust models. Moreover, training data should be as possibly close as it can be to real user behavior, and we are interested in exploring the variations of user utterances that refer to visual entities. Motivated by these factors, we crowdsourced data collection using MTurk in a restricted setup to improve the robustness of our models and also obtain a more realistic evaluation test dataset. We create an interface that presents MTurkers three images representing three products along with their name, price, rating, and prime eligibility information. We then ask the MTurkers to provide an utterance to select a product among these three products as well as the position of the referred product (left, middle or right). The web interface is shown in Figure 2a. We provide close to 20K different product combinations and for each combination we assign five different MTurkers to complete the task, which results in 100K product selection utterances in total. In order to see what type of attributes MTurkers usually use to select the product, we also manually annotate the utterance type among a small user-group with a subsample of 200 instances. We divide the selection sentence types into selection by color, price, shape, rating, name, prime eligibility, and miscellaneous visual features. The annotation results are shown in Figure 2b. We find that 69.5% utterances use visual features to select the particular product, indicating the need for multimodal models.

4 Approach

Our proposed model for grounding the user query with respect to visual context consists of three main components: the query encoder, the visual entity encoder, and the candidate scorer. At a high level; (i) we first encode the query and each visual entity in a candidate set; (ii) next, we compute a score for each (query, visual entity) pair; and (iii) finally we choose the highest scoring entity. In the next few subsections, we define the notion of visual context and visual entity and describe how we create the candidate set. We then discuss the details of each component of the model architecture.

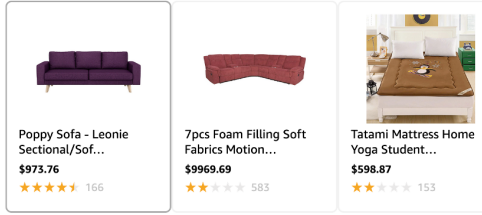


Figure 3: Schema Based Screen Context

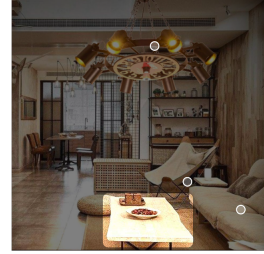


Figure 4: Full Image Screen Context

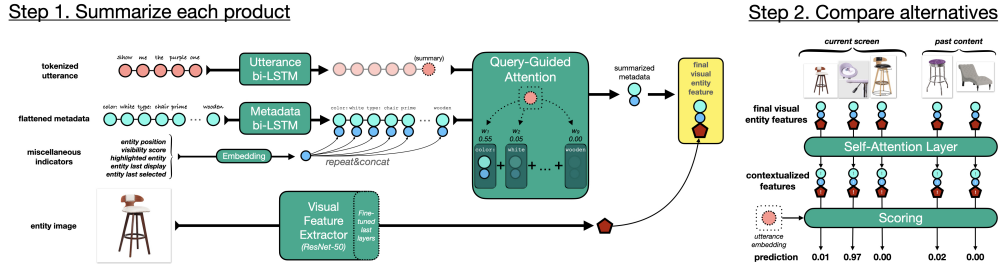


Figure 5: Model Architecture

4.1 Visual Context and Visual Entity

In a goal-oriented dialogue system, the screen content is populated by the back-end service providers after executing the API predicted by the agent. The screen layout is typically defined using schemas. There are generally multiple schemas used within an experience and each schema may contain multiple visual elements. For instance, for a shopping experience, the search page may contain three sofa images, a next page button and a go back button for a total of 5 visual elements. We consider each such visual element as one visual entity. Visual context corresponding to a particular turn is a collection of all the visual entities within a page. During a conversation, the user will typically interact with the screen (verbally or through touch) via these visual elements, and hence those are likely candidates for visual grounding. Figure 3 shows an example of a schema-based screen, with 3 sofa entities. Note that a visual entity may have not only the visual modality (i.e., image) but also textual modality (e.g., metadata such as name and price) as seen in the Figure.

However, often the schema may not be as well defined, or the full screen may contain just an image. In these cases, we use an off-the-shelf object detector to annotate the contents on the screen. We consider each detected object as a visual entity and the set of all the detected objects as the visual context. This is necessary as during the conversation, a user may be interested in items embedded within the image on the screen. For instance, in Figure 4 the user may ask the agent to zoom on the coffee table. We utilize an Amazon internal tool called StyleSnap for object detection. It provides the bounding box, a category label and a finer-label for each of the detected objects. As before, these visual entities also have both visual modality and the corresponding textual modality.

To construct the candidate set, we include all the visual entities from the current screen. Furthermore, we also include historical visual entities from previous screens in earlier turns of the conversation. This allows the user to refer to past entities, for example, “Compare the red sofa with the grey one that you showed earlier”. We apply de-duplication on the candidate set.

4.2 Model Architecture

We present the full architecture in Figure 5. We describe its various components in the following subsections.

4.2.1 Query Encoder

The query consists of multiple components: dialogue context, the current user utterance, and the API argument that needs to be filled. An API can have multiple arguments; the visual grounding model needs to uniquely point to the referred entity for each argument. The dialogue context is flattened and encoded using a bi-directional LSTM. The current user utterance is also encoded with a (separate) bi-directional LSTM. For the API argument, we encode the argument name and argument type using dense embeddings. The final query representation is obtained by concatenating the final hidden states from dialogue context and current utterance LSTMs together with the API argument embeddings.

4.2.2 Visual Entity Encoder

A visual entity has two main components: the metadata and the image. Metadata features are required as a user can refer to visual entities via information present in metadata such as name, rating, and prime eligibility. For instance, users can say *“What are the dimensions of the four star one”*. Image features are also required as the users can also refer to entities via visual characteristics. For instance, users can say *“Add the striped chair to my cart”*. In addition, we also include the spatial properties of the entity as displayed on screen. This includes the location and visibility of the visual entity on screen. This allows users to refer visual entities based on their geometry in the visual context, for instance, *“Tell me details about the third one”*. Visibility is needed so that the model can ignore partially visible entities (that might have appeared during touch based interactions with the screen). This allows the model to learn an optimal acceptance threshold for visibility from the data, without having to hard-code a fixed value. We also encode the state of the visual entity where state is a binary feature indicating whether an entity is highlighted or not on the screen. An entity can be highlighted when the user or the agent mentions it. This allows the model to resolve anaphoric references to that entity, for instance, *“Add it to my cart”*. Finally, we also include memory features indicating the turn when the entity was last visible and when it was last selected by the user. This allows the model to distinguish past entities from current screen entities and to bias towards recently-referenced entities.

We encode the metadata using a bi-directional LSTM over a flattened sequence of metadata property names and their values. The flattened metadata sequence can be arbitrarily long, since there could be large number of attributes. To allow the model to focus on the desired attributes, we introduce query-guided attention. The encoded query attends to each token in the metadata sequence, and we compute the query-attended metadata encoding. For the image, we extract visual features using the learned top of the pre-trained ResNet-50 model. For encoding the entity location, we extract the position of the entity on screen and encode it using sinusoidal positional embeddings (19). We assume a linear layout of visual entities thus only encode the x -coordinate positions. In general, the 4-D bounding box can directly be encoded using sinusoidal embeddings similar to (5). For visibility, we bucketize the visibility score and embed it using 1-hot vector. For encoding the highlighted state, we use a small dense embedding. Finally, memory features are encoded with sinusoidal positional embeddings as well. All the feature encodings are concatenated to form the joint representation of the visual entity. In order to make each visual entity aware of all the other entities in the visual context, we add a self-attention layer on top of the joint representation. This allows the model to refer entities based on their relative positions on the screen, for instance, *“Select the rightmost one”*. This also allows the model to perform comparative reasoning, for instance, *“Is the highest rated one prime eligible”*. The self-attended encoding is the final representation for visual entities.

4.2.3 Candidate Scorer

The scorer is a simple bi-linear attention layer. We compute the attention between the query encoding and visual entity encoding for each visual entity in the candidate set. We use resulting attention scores as relevance scores and the entity with the highest score is returned as the output.

5 Experiments and Discussion

We present the results of our experiments on our furniture shopping dataset that contains simulated data, as well as data from MTurkers. The product catalog for our dataset is built using a small subset of Amazon catalogue with close to 50K furnitures. We generate the simulated dataset using the

Model	Accuracy (%)	
	Simulated Test Set	MTurk Test Set
Random Baseline	12.68	33.44
Proposed Model (Only current screen context)	84.89	84.23
Proposed Model (Screen context from last 3 turns)	97.73	85.13

Table 1: MTurk set only contains examples for referencing visual entities from current screen. Simulated set has examples for user reference both current screen and past screens.

multimodal dialogue simulator described in Section 3.1. The simulated training and test set contain about 300K and 19.5K examples respectively. We use non-overlapping subsets of catalogues for training and test data generation. To improve diversity in training set and collect a more realistic test set, we augment the simulated data with an MTurk collection following the setup described in Section 3.2. We collect about 95K examples, which we split into 8 : 1 : 1 for training, development, and test without overlap. We use FastText for word embeddings (13). LSTM hidden size is 50 for query and metadata encoders. For image encoder, we use a ResNet-50 model adding a 50-dim top layer and fine-tuning the top-3 layers. For encoding positions and highlighted state, we use embeddings of size 50 and 5 respectively. The visibility score is separated into 20 buckets. We train the models using Adam optimizer without weight decay and minimize the cross-entropy loss. We use a batch size of 128 and a learning rate of 0.001.

Table 1 shows the performance of the proposed visual grounding model on simulated and MTurk test sets. We train two variants of the proposed model, one where the candidate set is created only from the current screen and the other where it also includes past visual entities. The model architecture is same in both the cases. For baseline, we randomly choose a visual entity from candidate set. Note that, the random baseline performance is better on the MTurk set. This is because the way the MTurk collection is set up, it only contains examples with current screen context and thereby results in smaller candidate set. For the same reason, the performance on MTurk test set is largely unchanged whether or not we train the proposed model with historical visual context. However, on the simulated set, including historical entities gives significant improvement, primarily for cases where the user tries to reference a previously shown entity, e.g., *“Is the blue one cheaper than the previous red one”*. Our proposed model is able to achieve reasonable accuracy of $\sim 85\%$ on a realistic and relatively difficult MTurk set.

We also conduct ablation studies across different dimensions to gauge impact of each component on model performance. Table 2 shows the results of our ablation study. We observe that:

1. The MTurk data and simulated data are complimentary. Training with only simulated data leads to poor performance on MTurk set and vice-versa. More importantly, when the model is trained with combined data, it leads to knowledge transfer resulting in better performance on both test sets.
2. Both metadata and visual features contribute to the model accuracy. When trained without metadata features, the performance drops significantly on both test sets. The resulting model cannot resolve entities based on metadata attributes such as name and price. However, it can still reference based on visual features like color, which is reflected in the almost unchanged accuracy for reference by color. On the other hand, when trained without visual features, color reference accuracy drops significantly. With this model, the drop in overall accuracy is much higher on MTurk set compared to the simulated set. This is because $\sim 70\%$ examples in the MTurk set require referencing via visual characteristics while the simulated set has only $\sim 17\%$ such cases.
3. We see a $\sim 4\%$ accuracy improvement from the query guided attention. It helps the model to focus on the right attributes in the metadata based on the user query. Entity self-attention layer makes entities aware of each other and gives another significant performance boost. We see higher gains on the MTurk set as it has many examples requiring comparative reasoning, e.g., *“Show me the highest rated one”*. Changing the loss function from binary cross-entropy to cross-entropy allows the model to better discriminate entities in the candidate set and provides an another 1–2% improvement.

Model	Accuracy (%)		
	Simulated Test Set		MTurk Test Set
	Overall	Referencing by Color	Overall
Random Baseline	33.52	27.23	33.44
Data Source			
Simulated + MTurk data	96.22	84.68	85.89
– MTurk data	95.58	82.71	57.33
– Simulated data	43.00	55.32	82.50
Feature Type			
Metadata + Visual features	96.22	84.68	85.89
– Metadata features	80.66	83.61	62.43
– Visual features	91.47	54.83	72.63
Model Component (Without Visual Features)			
Vanilla concatenation	82.00	49.63	52.42
+ Query-Guided attention	86.27	50.12	56.55
+ Entity self-attention	90.59	51.47	70.53
+ Cross-Entropy loss	91.47	54.83	72.63

Table 2: Ablation Experiments

6 Conclusion

We introduce a model that can resolve visual entities within dialogues. The new model can perform visual reasoning by comparing various entities and choosing the correct visual entity based on dialogue and visual context. To train and test model, we introduce a new multimodal dialogue simulator as well as a mechanism for crowd-sourcing multimodal data. On data that are close to real world, that relies on actual products from Amazon.com and utterances collected through crowd-sourcing, we show that our models have high accuracy of 85%. We also provide a detailed ablation study to show how each of component contribute to this accuracy. In the future, we plan to integrate visual context and dialogue context into a single model rather than having separate representation as presented in this work.

References

- [1] Anish Acharya, Suranjit Adhikari, Sanchit Agarwal, Vincent Auvray, Nehal Belgamwar, Arijit Biswas, Shubhra Chandra, Tagyoung Chung, Maryam Fazel-Zarandi, Raefer Gabriel, Shuyang Gao, Rahul Goel, Dilek Hakkani-Tur, Jan Jezabek, Abhay Jha, Jiun-Yu Kao, Prakash Krishnan, Peter Ku, Anuj Goyal, Chien-Wei Lin, Qing Liu, Arindam Mandal, Angeliki Metallinou, Vishal Naik, Yi Pan, Shachi Paul, Vittorio Perera, Abhishek Sethi, Minmin Shen, Nikko Strom, and Eddie Wang. Alexa conversations: An extensible data-driven approach for building task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 125–132, Online, June 2021. Association for Computational Linguistics.
- [2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.
- [3] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335, 2017.

- [4] Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5503–5512, 2017.
- [5] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018.
- [6] Ronghang Hu and Amanpreet Singh. Unit: Multimodal multitask learning with a unified transformer, 2021.
- [7] Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. Simmc 2.0: A task-oriented dialog dataset for immersive multimodal conversations. *arXiv preprint arXiv:2104.08667*, 2021.
- [8] Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. Clev-dialog: A diagnostic dataset for multi-round reasoning in visual dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 582–595, 2019.
- [9] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.
- [10] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020.
- [11] Bing Liu, Gokhan Tur, Dilek Hakkani-Tur, Pararth Shah, and Larry Heck. End-to-end optimization of task-oriented dialogue model with deep reinforcement learning. *arXiv:1711.10712*, 2017.
- [12] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic vi-
siolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019.
- [13] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhersch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.
- [14] Seungwhan Moon, Satwik Kottur, Paul A Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difrango, Ahmad Beirami, Eunjoon Cho, et al. Situated and interactive multimodal conversations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1103–1121, 2020.
- [15] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [16] Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. Building a Conversational Agent Overnight with Dialogue Self-Play. (i), 2018.
- [17] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*, 2019.
- [18] Tao Tu, Qing Ping, Govindarajan Thattai, Gokhan Tur, and Prem Natarajan. Learning better visual dialog agents with pretrained visual-linguistic representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5622–5631, 2021.

- [19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.
- [20] Tsung Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, 1:438–449, 2017.
- [21] Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 2020.
- [22] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315, 2018.