Embodied BERT: A Transformer Model for Embodied, Language-guided Visual Task Completion

Alessandro Suglia[1] **Qiaozi Gao**[2] **Jesse Thomason**[2,3]

Govind Thattai[2] **Gaurav S. Sukhatme**[2,3]

[1]Heriot-Watt University; [2]Amazon Alexa AI; [3]University of Southern California

Abstract

Language-guided robots performing home and office tasks must navigate in and interact with the world. Grounding language instructions against visual observations and actions to take in an environment is an open challenge. We present Embodied BERT (EmBERT), a transformer-based model which can attend to high-dimensional, multi-modal inputs across long temporal horizons for language-conditioned task completion.Additionally, we bridge the gap between successful object-centric navigation models used for non-interactive agents and the language-guided visual task completion benchmark, ALFRED, by introducing object navigation targets for EmBERT training. EmBERT achieves competitive performance on the ALFRED benchmark, and is the first model to use a full, pretrained BERT stack while handling the long-horizon, dense, multi-modal histories of ALFRED. Model code is available at the following link: `https://github.com/amazon-research/embert`

1 Introduction

Language is grounded in agent experience based on interactions with the world and other actors in it (Bisk et al., 2020; Bender and Koller, 2020). Task-oriented, instructional language focuses on *objects* and *interactions* between objects and actors, as seen in instructional datasets (Damen et al., 2020; Koupaee and Wang, 2018), as a function of the inextricable relationship between language and objects (Quine, 1960). That focus yields language descriptions of object targets for manipulation such as *put the strawberries on the cutting board and slice them into pieces* (Chai et al., 2018). We demonstrate that predicting navigational object landmarks *in addition* to manipulation object targets improves the performance of an instruction following agent in a rich, 3D simulated home environment. We posit that object-centric navigation is a key piece of semantic and topological navigation (Kuipers and Byun, 1991) for Embodied AI (EAI) agents generally.

Substantial modeling (Majumdar et al., 2020) and benchmark (Qi et al., 2020b) efforts in EAI navigation focus on identifying object landmarks (Blukis et al., 2018) and destinations (Batra et al., 2020b). However, for agent *task completion*, where agents must navigate an environment and manipulate objects towards a specified goal (Gordon et al., 2017; Shridhar et al., 2020), modeling efforts thus far have predicted movement actions without explicitly identifying *navigation* object targets (Singh et al., 2020; Pashevich et al., 2021; Nguyen et al., 2021; Abramson et al., 2020). We address this gap, grounding navigation instructions like *Head to the sink in the corner* by predicting the spatial locations of the goal `sink` object at each timestep (Figure 1).

Transformer-based models in EAI score the alignment between a language instruction and an already-completed path (Majumdar et al., 2020) or introduce recurrence by propagating part of the hidden state to the next timestep (Hong et al., 2020). The former requires beam search over sequences of environment actions, which is not feasible when actions cannot be undone, such as `slicing` an `apple`. The latter introduces a heavy memory requirement, and is feasible only with short trajectories of four to six steps. We overcome both limitations by *decoupling* the embedding of language and visual features from the prediction of what action to take next in the environment. We first embed language and visual observations *at single timesteps* using a multi-modal transformer architecture, then train a transformer decoder model to consume sequences of such embeddings to decode actions (Figure 3).

We introduce Embodied BERT (EmBERT), which implements these two key insights:

1. **Object-centric Navigation** unifies the dis-

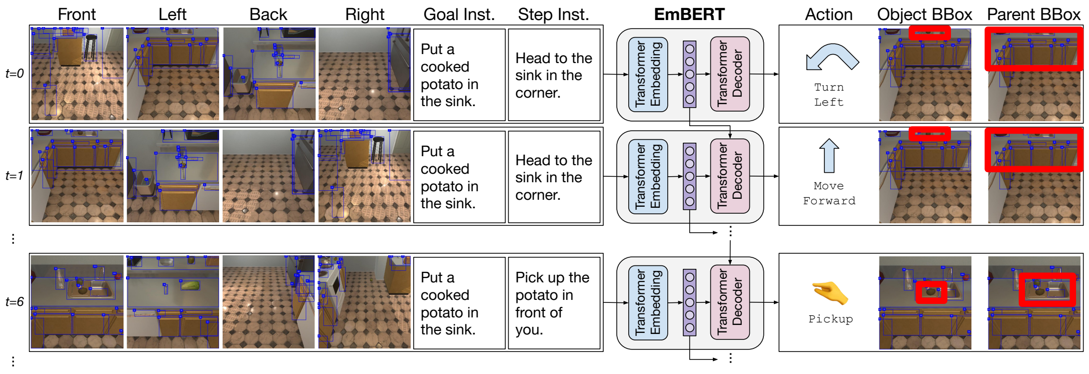


Figure 1: **Embodied BERT.** EmBERT attends to object detections in a panoramic view around an agent, then predicts an action and both a target object and target object parent for both navigation and manipulation actions. For example, at timesteps $t = 0, 1$ above, the model must predict the `sink` object target and its parent, the `countertop`, while at $t = 6$ it predicts both the object `potato` to pick up and the `sink` on which it rests.

joint *navigation* and *interaction* action sequences in ALFRED, giving navigation actions per-step object landmarks.

2. **Decoupled Multimodal Transformers** enable extending transformer based multimodal embeddings and sequence-to-sequence prediction to the fifty average steps present in ALFRED trajectories.

2 Related Work

Natural language guidance of robots (Tellex et al., 2020) has been explored in contexts from furniture assembly (Tellex et al., 2011) to quadcoptor flight control (Blukis et al., 2019).

Embodied AI. For task completion benchmarks, actions like `pickup` must be coupled with object targets in the visual world, with specification ranging from mask prediction only (Shridhar et al., 2020) to proposals for full low level gripper control (Batra et al., 2020a). Similarly, navigation benchmarks incorporate objects as targets in tasks like object navigation (Qi et al., 2020b; Batra et al., 2020b; Kurenkov et al., 2020), and explicitly modeling those objects assists generally at navigation success (Shrivastava et al., 2021; Qi et al., 2020a, 2021). Many successful modeling approaches for navigation benchmarks incorporate multimodal transformer models that require large memory from recurrence (Hong et al., 2020), beam search over potential action sequences (Majumdar et al., 2020), or shallow layers without large-scale pretraining to encode long histories (Pashevich et al., 2021; Magassouba et al., 2021). In this work, we incorporate navigation object targets into the ALFRED task completion benchmark (Shridhar et al., 2020), and decouple transformer-based multimodal state embedding from transformer-based translation of state embeddings to action and object target predictions. In addition, differently from other approaches that train from scratch their language encoder, we successfully exploit the BERT stack in our multi-modal architecture. In this way, EmBERT can be applied to other language-guided tasks such as VLN and Cooperative Vision-and-Dialog Navigation (Thomason et al., 2019).

Language-Guided Task Completion. Table 1 summarizes how EmBERT compares to current ALFRED modeling approaches. ALFRED language instructions are given as both a single high level goal and a sequence of step-by-step instructions (Figure 2). At each timestep, we encode the goal instruction and a predicted current step-by-step instruction. We train EmBERT to predict when to advance to the next instruction, a technique introduced by LWIT (Nguyen et al., 2021).

EmBERT uses a panoramic view space to see all around the agent. Rather than processing dense, single vector representations (Shridhar et al., 2020; Singh et al., 2020; Pashevich et al., 2021; Kim et al., 2021; Blukis et al., 2021), EmBERT attends directly over object bounding box predictions embedded with their spatial relations to the agent, inspired by LWIT (Nguyen et al., 2021) and a recurrent VLN BERT model (Hong et al., 2020). We similarly follow prior work (Singh et al., 2020; Pashevich et al., 2021; Nguyen et al., 2021; Kim et al., 2021; Zhang and Chai, 2021) in predicting these bounding boxes as object targets for actions like `Pickup`, rather than directly predicting a dense object segmentation mask (Shridhar et al., 2020).

Consider the step *heat the mug of water in the*

	Language Obs.		Visual Obs.		Historical Obs.		Inference	
	Goal Inst. Structure	Inst. Split	Views	Features	As Inputs	Hidden States	Mask Pred.	Nav Obj. Pred.
SEQ2SEQ (Shridhar et al., 2020)	✗	✗	Single	Dense	✗	LSTM	Direct	✗
MOCA (Singh et al., 2020)	✓	✗	Single	Dense	✗	LSTM	BBox	✗
ET (Pashevich et al., 2021)	✗	✗	Single	Dense	TF	✗	BBox	✗
LWIT (Nguyen et al., 2021)	✓	✓	Multi	BBox	✗	LSTM	BBox	✗
ABP (Kim et al., 2021)	✓	✗	Multi	Dense	✗	LSTM	BBox	✗
HITUT (Zhang and Chai, 2021)	✓	✓	Single	BBox	SG	✗	BBox	✗
HLSM (Blukis et al., 2021)	✓	-	Single	Dense	SG+Map	✗	Direct	✗
EmBERT	✓	✓	Multi	BBox	✗	TF	BBox	✓

Table 1: **Model comparison.** EmBERT uses a multimodal transformer (TF) to embed language *inst*ructions and detected objects in a panoramic view, and a transformer decoder to produce action and object predictions. Ours is the first ALFRED model to add object prediction to *navigation* steps. Other methods maintain history by taking previous transformer states (TF) (Pashevich et al., 2021), subgoal prediction structures (SG) (Zhang and Chai, 2021; Blukis et al., 2021), or maintained voxel maps (Blukis et al., 2021) as input.

microwave, where the visual observation before turning the `microwave` on and after turning the `microwave` off are identical. Transformer encodings of ALFRED's large observation history are possible only with shallow networks (Pashevich et al., 2021) that cannot take advantage of large scale, pretrained language models used on shorter horizons (Hong et al., 2020). We decouple multimodal transformer state encoding from sequence to sequence state to action prediction, drawing inspiration from the AllenNLP SQuAD (Rajpurkar et al., 2016) training procedure (Gardner et al., 2017).

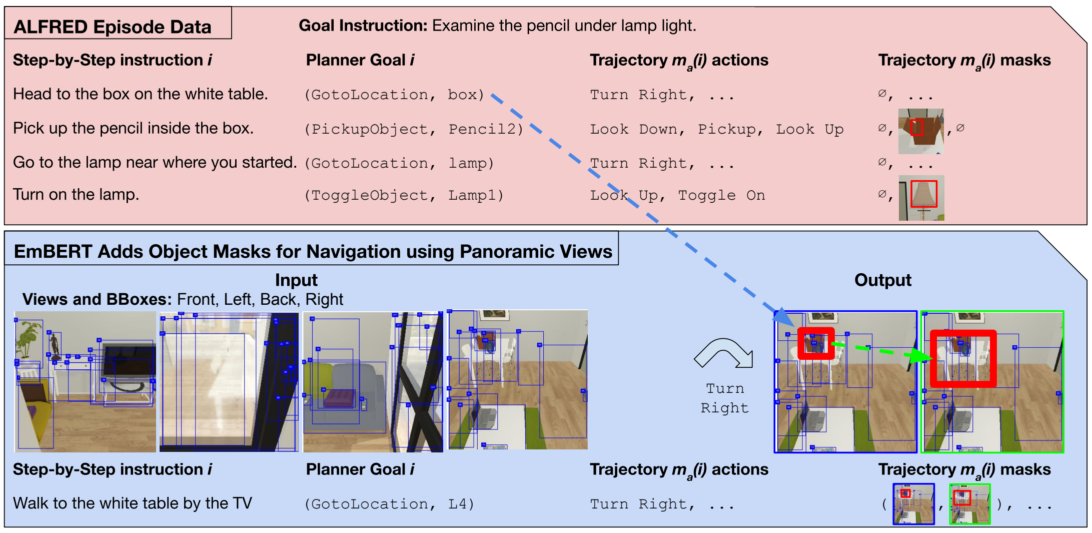
Our EmBERT model is the first to utilize an auxiliary, object-centric navigation prediction loss during joint navigation and manipulation tasks, building on prior work that predicted only the *direction* of the target object (Storks et al., 2021) or honed in on landmarks during navigation-only tasks (Shrivastava et al., 2021). While mapping environments during inference has shown promise on both VLN (Fang et al., 2019; Chen et al., 2021) and ALFRED (Blukis et al., 2021), we leave the incorporation of mapping to future work.

3 The ALFRED Benchmark

The ALFRED benchmark (Shridhar et al., 2020) pairs household task demonstrations with written English instructions in 3d simulated rooms (Kolve et al., 2017). ALFRED tasks are from seven categories: PICK & PLACE, STACK & PLACE, PICK TWO & PLACE, CLEAN & PLACE, HEAT & PLACE, COOL & PLACE, and EXAMINE IN LIGHT. Each task involves one or more objects that need to be manipulated, for example an `apple`, and a final receptacle on which they should come to rest, for example a `plate`. Many tasks involve intermediate state changes, for example HEAT & PLACE requires *cooking* the target object in a `microwave`.

Supervision Data. Each ALFRED episode comprises an initial state for a simulated room, language instructions, planning goals, and an expert demonstration trajectory. The language instructions are given as a high-level goal instruction $\mathcal{I}_g$, for example *Put a cooked egg in the sink*, together with a sequence of step-by-step instructions $\vec{\mathcal{I}}$, for example *Turn right and go to the sink*, *Pick up the egg on the counter to the right of the sink*, ... The planning goals $\mathcal{P}$ (or *subgoals*) are tuples of goals and arguments, such as `(SliceObject, Apple)` that unpack to low-level sequences of actions like picking up a `knife`, performing a `slice` action on an `apple`, and putting the `knife` down on a `countertop`. The expert demonstration trajectory $\mathcal{T}$ is a sequence of action and object mask pairs, where $\mathcal{T}_j = (a_j, M_j)$. Each step-by-step instruction $\mathcal{I}_i$ corresponds to a sub-sequence of the expert demonstration, $\mathcal{T}_{j:k}$ given by alignment lookup $m_a(i) = (j, k)$ and to a planning goal $\mathcal{P}_b$ by alignment lookup $m_p(i) = b$. For example, in Figure 2, instruction $\mathcal{I}_0$ corresponds to a `GotoLocation` navigation goal, as well as a sequence of turning and movement API actions that a model must predict.

Model Observations. At the beginning of each episode in timestep $t = 0$, an ALFRED agent receives the high-level and step-by-step language instructions $\mathcal{I}_g, \vec{\mathcal{I}}$. At every timestep t, the agent receives a 2d, RGB visual observation representing the front-facing agent camera view, $\mathcal{V}^{\mathcal{F}}$. ALFRED models produce an action a_t from among 5 navigation (e.g., `Turn Left`, `Move Forward`, `Look Up`) and 7 manipulation actions (e.g., `Pickup`,



ALFRED Episode Data
Goal Instruction: Examine the pencil under lamp light.
Step-by-Step instruction i
Planner Goal i
Trajectory m_a(i) actions
Trajectory m_a(i) masks
Head to the box on the white table.
(GotoLocation, box)
Turn Right, ...
∅, ...
Pick up the pencil inside the box.
(PickupObject, Pencil2)
Look Down, Pickup, Look Up
∅, , ∅
Go to the lamp near where you started.
(GotoLocation, lamp)
Turn Right, ...
∅, ...
Turn on the lamp.
(ToggleObject, Lamp1)
Look Up, Toggle On
∅,
EmBERT Adds Object Masks for Navigation using Panoramic Views
Input
Views and BBoxes: Front, Left, Back, Right
Output
Turn Right
Step-by-Step instruction i
Planner Goal i
Trajectory m_a(i) actions
Trajectory m_a(i) masks
Walk to the white table by the TV
(GotoLocation, L4)
Turn Right, ...
(,), ...

Figure 2: **EmBERT Auxiliary Predictions.** ALFRED provides goal and step-by-step language instructions that are aligned with planner goals and sequences of trajectory actions in an expert demonstration (top). EmBERT additionally identifies *navigational* object targets in a panoramic view (bottom). EmBERT predicts an object target and its higher visibility parent receptacle, such as the `table` on which the `box` rests.

`ToggleOn`, `Slice`), as well as an object mask M_t. Predicted action a_t and mask M_t are executed in the ALFRED environment to yield the next visual observation. For navigation actions, prediction M_t is ignored, and there is no training supervision for objects associated with navigation actions.

EmBERT Predictions. EmBERT gathers additional visual data (Figure 2). After every navigation action, we turn the agent in place to obtain left, backwards, and right visual frames $\mathcal{V}^{\mathcal{L}}$, $\mathcal{V}^{\mathcal{B}}$, $\mathcal{V}^{\mathcal{R}}$. Following prior work (Singh et al., 2020), we run a pretrained Mask-RCNN (He et al., 2017) model to extract bounding boxes from our visual observations at each view. We train EmBERT to select the bounding box which has the highest intersection-over-union with M_t (more details in Section 4).

We define a *navigation object target* for navigation actions. For navigation actions taken during language instruction $\mathcal{I}_i$, we examine the frame $\mathcal{V}^{\mathcal{F}}{}_k$ at time k for $\mathcal{T}_k$; $m_a(i) = (j, k)$. We identify the object instance O of the class specified in the planning goal $\mathcal{P}_{m_b(i)}$ in $\mathcal{V}^{\mathcal{F}}{}_k$. We define this object O as the navigation object target for all navigation actions in $\mathcal{T}_{j:k}$ by pairing those actions with object mask M^O to be predicted during training. We also add a training objective to predict the *parent receptacle* $P(O)$ of O. Parent prediction enables navigating to landmarks such as the `table` for instructions like *Turn around and head to the box on the table*, where the `box` compared to the table on which it rests (Figure 2).

4 Embodied BERT

EmBERT uses a transformer encoder for jointly embedding language and visual tokens and an transformer decoder for long-horizon planning and object-centric navigation predictions (Figure 3).

4.1 Multimodal encoder

We use OSCAR (Li et al., 2020) as a backbone transformer module to fuse language and visual features at each ALFRED trajectory step. We obtain subword tokens for the goal instruction $\mathcal{I}_g = \{g_1, g_2, \ldots, g_n\}$ and the step-by-step instruction $\mathcal{I}_j = \{i_1, i_2, \ldots, i_m\}$ using the WordPiece tokenizer (Wu et al., 2016) and process the sequence as: `[CLS]` $\mathcal{I}_g$ `[SEP]` $\mathcal{I}_j$ `[SEP]`, using token type ids to distinguish the goal and step instructions. We derive token embeddings $\mathbf{L} \in \mathcal{R}^{(m+n+3)\times d_e}$ using the BERT (Devlin et al., 2019) embedding layer, where d_e is the embedding dimensionality.

We provide EmBERT with object-centric representations by using MaskRCNN (He et al., 2017) features to represent detected objects in every frame of the panorama view. We freeze the weights of a MaskRCNN model fine-tuned for AI2-THOR frames (Singh et al., 2020). We fix the number of object detections in the front view $\mathcal{V}^{\mathcal{F}}$ to 36, while limiting those in the side views to 18. We represent each object $o \in \mathcal{O}$ as an embedding $\mathbf{o} \in \mathbb{R}^{d_o}$, which is a concatenation of: 1) detection ResNet (He et al., 2016) features; 2) bounding box coordinates; 3) bounding box relative area; and

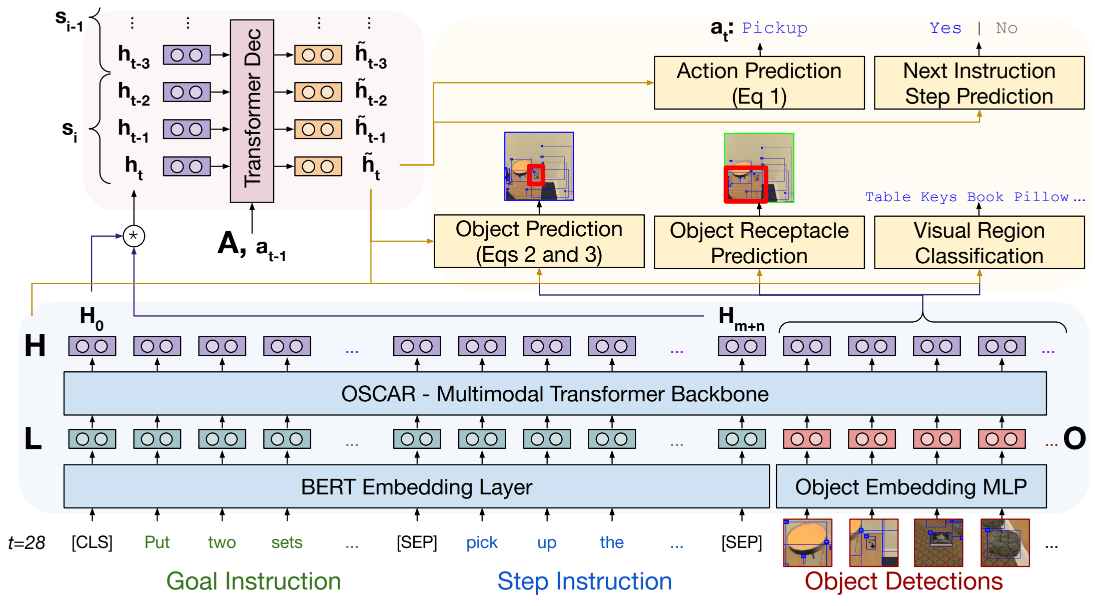


Figure 3: **Proposed Embodied BERT model.** A multimodal encoder embeds goal- and step-level instructions alongside object detections from a panoramic view around the agent. This encoder produces a temporally independent hidden state h_t. A sequence of such hidden states are attended by a segment-level recurrent action decoder to produce time-dependent states $\tilde{h}_t$. EmBERT is trained in segments s_i to balance gradient flow over time with memory constraints, and previous segments are cached to be attended over in future timesteps. Time-dependent state $\tilde{h}_t$ is used to predict the next action, whether to start attending to the next step-by-step instruction, what object to target in the environment, that object's parent receptacle, and detected object classes.

4) vertical and horizontal heading of the object related to the current agent position, following prior work (Storks et al., 2021). These representations make up the observed object embeddings $\mathbf{O}$. We use a one layer MLP to map object embeddings of dimensionality d_o to size d_e.[1] The multi-modal transformer backbone consumes the token and object embeddings to produce multi-modal hidden states $\mathbf{H} \in \mathbb{R}^{m+n+|O| \times d_e}$. We obtain these state representations, $\mathbf{h}_t$, for each timestep t by computing an element-wise product between $\mathbf{H}_0$ and $\mathbf{H}_{m+n}$, the hidden state of the `[CLS]` token and the last `[SEP]` token placed between language tokens and objects, similar in spirit to the approach described in (Zhou et al., 2020). In this way, we can generate *temporally independent* agent states for an entire trajectory resulting in a sequence of states $\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{|\mathcal{T}|}\}$.

4.2 Segment-Level Recurrent Action Decoder

The ALFRED challenge requires models to learn to complete action sequences averaging 50 steps and spanning multiple navigation and manipulation sub-goals. However, due to the quadratic complexity of the self-attention mechanism, feeding long sequences to transformers is computationally expensive (Beltagy et al., 2020). Inspired by the TransformerXL model (Dai et al., 2019), we design the Segment-Level Recurrent Action Decoder architecture that models long trajectories with recurrent segment-level state reuse. At training time we divide trajectories into temporal segments of size s. Given two consecutive segments, $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$, EmBERT caches the representations generated for segment $\mathbf{s}_i$. The computed gradient does not flow from $\mathbf{s}_{i+1}$ to $\mathbf{s}_i$, but cached representations are used as extended context. When predicting the next action, the model can still perform self-attention over the previous segment representations, effectively incorporating additional contextual information that spans an high number of previous timesteps.

The TransformerXL model is intended as an encoder-only architecture which is not able to perform cross-attention with some encoder hidden states. Therefore, we introduce two novel elements to its architecture: 1) *encoder hidden states cache*; 2) *cross-attention over encoder states*. First, our extended context is composed of both agent state representations and hidden states from the previous segment $\mathbf{s}_i$. In addition, to perform

[1]In our experiments, in order to reuse the visual embedding available in the OSCAR checkpoint, we use an additional one layer MLP to adapt our visual features to the visual embeddings space learned by OSCAR.

cross-attention between decoder and encoder hidden states, we modify the TransformerXL self-attention mechanism following common practice in designing transformer decoders (Vaswani et al., 2017). EmBERT encodes the previous actions for the current timestep a_{t-1} and extracts an action embedding $\mathbf{a}_t$ from a learnable embedding matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{A}| \times d_a}$. In the TransformerXL's multi-head self-attention layers, we generate *keys* and *values* from the agent state representations (*encoder*) and *queries* from the action embeddings (*decoder*). We obtain *time-dependent* agent state representations $\{\tilde{\mathbf{h}}_1, \tilde{\mathbf{h}}_2, \ldots, \tilde{\mathbf{h}}_{|\mathcal{T}|}\}$ as output.

Given time-dependent hidden states, the model predicts action and object mask outputs. We learn a probability distribution over the agent actions $\mathcal{A}$ by using a two layer feedforward network (FFN) with dropout and GeLU (Hendrycks and Gimpel, 2016) activation receiving the hidden state $\tilde{\mathbf{h}}_t$ for the timestep t:

$$\tilde{\mathbf{h}}^1_t = \text{GeLU}(\tilde{\mathbf{h}}_t \mathbf{W}^1) \quad P(a_t|\tilde{\mathbf{h}}_t) = \text{softmax}(\tilde{\mathbf{h}}^1_t \mathbf{W}^2), \tag{1}$$

where $\mathbf{W}^1 \in \mathbb{R}^{d_e \times d_e}$ and $\mathbf{W}^2 \in \mathbb{R}^{d_e \times |\mathcal{A}|}$ are two weight matrices. We use sequence-based cross-entropy loss (Sutskever et al., 2014), $\mathcal{L}_A$, to supervise the action prediction task. In addition, we derive *time-dependent* fine-grained representations of token and object embeddings. We use conditional scaling (Dumoulin et al., 2018) to fuse the decoder hidden state $\tilde{\mathbf{h}}_t$ with the embedding $\mathbf{H}$ to produce the time-dependent embeddings $\tilde{\mathbf{H}}$:

$$\tilde{\mathbf{c}} = \mathbf{W}_t \tilde{\mathbf{h}} \qquad \tilde{\mathbf{H}}_i = \tilde{\mathbf{c}} \cdot \mathbf{H}_{i, i=\{1,\ldots,(m+n+|O|)\}}\,, \tag{2}$$

where $\mathbf{W}_t \in \mathbb{R}^{d_e \times d_e}$ is a weight matrix used to adapt the representation of the original decoder hidden state $\tilde{\mathbf{h}}$. We predict target objects by selecting one bounding box among the detections in $\mathcal{V}^{\mathcal{F}}$ for manipulation actions, or any view for navigation actions. We treat object mask prediction as a classification task where the model first extracts time-dependent object embeddings $\tilde{\mathbf{O}} = \tilde{\mathbf{H}}_i$, $i = \{(m+n), \ldots, (m+n+|O|)\}$, and then generates logits for each object as follows:

$$\tilde{\mathbf{o}}^1_i = \text{GeLU}(\tilde{\mathbf{O}} \mathbf{W}^1_o) \quad P(o_i|\tilde{\mathbf{O}}_i) = \text{softmax}(\tilde{\mathbf{o}}^1_i \mathbf{W}^2_o), \tag{3}$$

where $\mathbf{W}^1_o \in \mathbb{R}^{d_e \times d_e}$ and $\mathbf{W}^2_o \in \mathbb{R}^{d_e \times 1}$ are two weight matrices. At training time, we determine the target object by using the Intersection-Over-Union score between the predicted object masks generated by MaskRCNN for each object and the gold object mask. To supervise this classification task, we use sequence-based cross-entropy loss, $\mathcal{L}_O$.

4.3 Auxiliary tasks

During the EmBERT training, we jointly optimize $\mathcal{L}_A$, $\mathcal{L}_O$, and several auxiliary tasks.

Next Instruction Prediction. Several existing models for ALFRED encode the sequence of language instructions $\mathcal{I}$ together with the goal (Table 1), or concatenate step-by-step instructions. These simplifications can prevent the model from carefully attending to relevant parts of the visual scene. EmBERT takes the first instruction at time $t = 0$, and performs add an auxiliary prediction task to advance from instruction $\mathcal{I}_j$ to instruction $\mathcal{I}_{j+1}$. To supervise the next-instruction decision, we create a binary label for each step of the trajectory that indicates whether that step is the last step for a specific sub-goal, as obtained by $m_a(i)$. We use a similar FNN as Equation 1to model a Bernoulli variable used to decide when to advance to the next instruction. We denote the binary cross-entropy loss used to supervise this task as $\mathcal{L}_{INST}$.

Object Target Predictions. EmBERT predicts a target object for navigation actions, together with the receptacle object containing the target, for example a `table` on which a `box` sits (Figure 2). For these tasks, we use an equivalent prediction layer to the one used for object prediction. We denote the cross-entropy loss associated with these task by $\mathcal{L}_{NAV}$ and $\mathcal{L}_{RECP}$.

Visual Region Classification. Class-conditioned representations are useful for agent manipulation, especially when combined with hand-crafted procedures for object selections (Singh et al., 2020). Inspired by masked region modeling tasks (Chen et al., 2020b; Shrivastava et al., 2021), we select with $\%15$ probability some objects part of the agent view in a given timestep t and we ask the model to predict their classes. Given the instruction *Turn around and walk to the book on the desk*, at the very first timestep of the trajectory it is likely that none of the mentioned objects are visible. Thus, we assume that at the last step of a sub-goal the agent will have in view the objects associated with the instruction. For the prediction task, we directly use the time-dependent object embeddings $\tilde{\mathbf{O}}$ and use an FFN (similar to Equation 1) to estimate a probability distribution over the ALFRED object labels. We use a cross-entropy loss denoted by $\mathcal{L}_{VRC}$ as supervision for this task.

	Leaderboard Test Fold Performance			
	Seen		*Unseen*	
Model	Task (PLW)	GC (PLW)	Task (PLW)	GC (PLW)
SEQ2SEQ (Shridhar et al., 2020)	3.98 (2.02)	9.42 (6.27)	.39 (0.08)	7.03 (4.26)
HITUT (Zhang and Chai, 2021)	21.27 (11.10)	29.97 (17.41)	13.87 (**5.86**)	20.31 (11.51)
MOCA (Singh et al., 2020)	22.05 (15.10)	28.29 (22.05)	5.30 (2.72)	14.28 (9.99)
HLSM (Blukis et al., 2021)	25.11 (6.69)	35.79 (11.53)	**16.29** (4.34)	**27.24** (8.45)
LWIT (Nguyen et al., 2021)	30.92 (25.90)	40.53 (**36.76**)	9.42 (5.60)	20.91 (**16.34**)
EMBERT	31.77 (23.41)	39.27 (31.32)	7.52 (3.58)	16.33 (10.42)
ET (Pashevich et al., 2021)	38.42 (**27.78**)	45.44 (34.93)	8.57 (4.10)	18.56 (11.46)
ABP (Kim et al., 2021)	**44.55** (3.88)	**51.13** (4.92)	15.43 (1.08)	24.76 (2.22)

Table 2: **Test Fold Performance.** Path weighted metrics are given in parentheses.

5 Experiments and Results

EmBERT achieves competitive performance with state of the art models on the ALFRED leaderboard test sets (Table 2), surpassing all but ET (Pashevich et al., 2021) and ABP (Kim et al., 2021) on *Seen* test fold performance (Table 3) at the time of writing. Notably, EmBERT achieves this performance without augmenting ALFRED data with additional language instructions, as is done in ET (Pashevich et al., 2021), or visual distortion as used in ABP (Kim et al., 2021).

Implementation Details. EmBERT is implemented using AllenNLP (Gardner et al., 2017), PyTorch-Lightning,[2] and Huggingface-Transformers (Wolf et al., 2019). We train using the Adam optimizer with weight fix (Loshchilov and Hutter, 2017), learning rate $2e^{-5}$, and linear rate scheduler without warmup steps. We use dropout of 0.1 for the hidden layers of the FFN modules and gradient clipping of 1.0 for the overall model weights. Our TransformerXL-based decoder is composed of 2 layers, 8 attention heads, and uses a memory cache of 200 slots. At training time, we segment the trajectory into 10 timesteps. In order to optimize memory consumption, we use bucketing based on the trajectory length. We use teacher forcing (Williams and Zipser, 1989) to supervise EmBERT during the training process. To decide when to stop training, we monitor the average between action and object selection accuracy for every timestep based on gold trajectories. The best epoch according to that metric computed on the *validation seen* set is used for evaluation. The total time for each epoch is about 1 hour for a total of 20 hours for each model configuration using EC2 instances `p3.8xlarge` using 1 GPU.

Action Recovery Module. For obstacle avoidance, if a navigation action fails, for example the agent choosing `MoveAhead` when facing a wall, we take the next most confident navigation action at the following timestep, as in MOCA (Singh et al., 2020). We introduce an analogous object interaction recovery procedure. When the agent chooses an interaction action such as `Slice`, we first select the bounding box of highest confidence to retrieve an object interaction mask. If the resulting API action fails, for example if the agent attempts to `Slice` a `Kettle` object, we choose the next highest confidence bounding box at the following timestep. The ALFRED challenge ends an episode when an agent causes 10 such API action failures.

Comparison to Other Models. Table 2 gives EmBERT performance against top and baseline models on the ALFRED leaderboard at the time of writing. *Seen* and *Unseen* sets refer to tasks in rooms that were or were not seen by the agent at training time. We report *Task* success rate and *Goal Conditioned (GC)* success rate. Task success rate is the average number of episodes completed successfully. Goal conditioned success rate is more forgiving; each episode is scored in $[0, 1]$ based on the number of subgoals satisfied, for example, in a STACK & PLACE task if one of two `mugs` are put on a `table`, the GC score is 0.5 (Shridhar et al., 2020). Path weighted success penalizes taking more than the number of expert actions necessary for the task.

EmBERT outperforms MOCA (Singh et al., 2020) on *Unseen* scenes, and several models on *Seen* scenes. The primary leaderboard metric is *Unseen* success rate, measuring models' generalization abilities. Among competitive models, EmBERT outperforms only MOCA at *Unseen* generalization success. Notably, EmBERT remains competitive on *Unseen path-weighted* metrics, because it does not perform any kind of exploration or mapping as in HLSM (Blukis et al., 2021) and

[2] `https://www.pytorchlightning.ai/`

		EmBERT				Validation Fold Performance			
						Seen		*Unseen*	
Init Weights	#SB	Mem	Nav O	$P(O)$	VRC	Task	GC	Task	GC
OSCAR	18	200	✓	✓	✓	28.54 (22.88)	38.69 (31.28)	1.46 (.72)	10.19 (6.25)
OSCAR	18	200	✓		✓	34.76 (28.46)	41.30 (35.50)	3.66 (1.55)	12.61 (7.49)
OSCAR	18	200	✓	✓		36.22 (27.05)	44.57 (35.23)	4.39 (2.21)	13.03 (7.54)
OSCAR	18	200	✓			**37.44** (**28.81**)	**44.62** (**36.41**)	**5.73** (**3.09**)	**15.91** (**9.33**)
OSCAR	18	200				23.66 (17.62)	29.97 (24.16)	2.31 (1.24)	12.08 (7.62)
BERT	18	200	✓	✓		26.46 (19.41)	35.70 (27.04)	3.53 (1.77)	13.02 (7.57)
OSCAR	9	200	✓	✓	✓	29.30 (20.14)	36.28 (27.21)	3.06 (1.13)	12.17 (6.69)
OSCAR	9	200	✓	✓		31.75 (23.52)	38.80 (32.21)	2.56 (1.28)	12.97 (8.24)
OSCAR	9	200		✓	✓	20.37 (16.30)	28.64 (23.11)	1.46 (0.75)	10.47 (6.26)
OSCAR	9	200	✓			28.33 (20.77)	36.83 (28.03)	2.68 (1.18)	11.60 (6.78)
OSCAR	9	200				27.84 (20.66)	36.59 (27.97)	2.44 (1.06)	11.46 (6.76)
OSCAR	0	200	✓	✓		25.31 (18.79)	34.27 (26.09)	3.42 (1.49)	12.25 (7.34)
OSCAR	9	1	✓	✓		20.98 (13.98)	33.33 (22.74)	1.10 (0.60)	10.33 (4.69)
OSCAR	18	1	✓	✓		21.95 (12.99)	35.04 (22.31)	1.58 (.54)	11.08 (6.18)
MOCA (Singh et al., 2020)						18.90 (13.20)	28.02 (21.81)	3.65 (1.94)	13.63 (8.50)

Table 3: **Validation Fold Performance.** We present ablations adjusting the number of side-view bounding boxes, attended memory length, with and without predicting navigation target O, target parent object $P(O)$, and visual region classification (VRC) loss. We also explore initializing our multi-modal encoder with BERT versus OSCAR initialization. The highest values per fold and metric are shown in **blue**. Path weighted metrics are given in parenthesis.

ABP (Kim et al., 2021).

We do not utilize the MOCA *Instance Association in Time* module (Singh et al., 2020) that is mimicked by ET (Pashevich et al., 2021). That module is conditioned based on the object class of the target object selected across timesteps. Because we directly predict object *instances* without conditioning on a predicted object class, our model must learn instance associations temporally in an implicit manner, rather than using such an inference time "fix".

EmBERT Ablations. Removing the object-centric navigation prediction unique to EmBERT decreases performance on all metrics (Table 3). We show that limiting memory for the action decoder to a single previous timestep, initializing with BERT rather than OSCAR weights, and limiting vision to the front view all decrease performance in both *Seen* and *Unseen* folds.

We find that our parent prediction and visual region classification losses, however, do not improve performance. To investigate whether a smaller model would benefit more from these two auxiliary losses, we ran EmBERT with only 9 bounding boxes per side view, which enables fitting longer training segments in memory (we use 14 timesteps, rather than 10). We found that those losses improved EmBERT performance on the *Unseen* environment via both success rate and goal conditions metrics, and improved success rate alone in *Seen* environments when the non-frontal views were limited to 9, rather than 18, bounding boxes. Given the similar performance of EmBERT with all three auxiliary losses at 18 and 9 side views, we believe EmBERT is over-parameterized with the additional losses and 18 side view bounding boxes. It is possible that data augmentation efforts to increase the volume of ALFRED training data, such as those in ET (Pashevich et al., 2021), would enable us to take advantage of the larger EmBERT configuration.

6 Conclusions

We apply the insight that object-centric navigation is helpful for language-guided Embodied AI to a benchmark of tasks in home environments. Our proposed Embodied BERT (EmBERT) model adapts the pretrained language model transformer OSCAR (Li et al., 2020), and we introduce a decoupled transformer embedding and decoder step to enable attending over many features per timestep as well as a history of previous embedded states (Figure 1). EmBERT is the first to bring object-centric navigation to bear on language-guided, manipulation and navigation-based task completion. We find that EmBERT's object-centric navigation and ability to attend across a long time horizon both contribute to its competitive performance with state-of-the-art ALFRED models (Table 3).

Moving forward, we will apply EmBERT to other benchmarks involving multimodal input through time, such as vision and audio data (Chen et al., 2020a), as well as wider arrays of tasks to accomplish (Puig et al., 2018). To further improve performance on the ALFRED benchmark, we could conceivably continue training the Mask RCNN model from MOCA (Singh et al., 2020) *forever* by randomizing scenes in AI2THOR (Kolve et al., 2017) and having the agent view the scene from randomized vantage points with gold-standard segmentation masks available from the simulator. For language supervision, we could train and apply a *speaker* model for ALFRED to generate additional training data for new expert demonstrations, providing an initial multimodal alignment for EmBERT, a strategy shown effective in VLN tasks (Fried et al., 2018).

7 Implications and Impact

We evaluated EmBERT only on ALFRED, whose language directives are provided as a one-sided "recipe" accomplishing a task. The EmBERT architecture is applicable to single-instruction tasks like VLN, as long as auxiliary navigation object targets can be derived from the data as we have done here for ALFRED, by treating the "recipe" of step-by-step instructions as empty. In future work, we would like to incorporate our model on navigation tasks involving dialogue (Thomason et al., 2019; de Vries et al., 2018) and real robot platforms (Banerjee et al., 2020) where lifelong learning is possible (Thomason et al., 2015; Johansen et al., 2020). Low-level physical robot control is more difficult than the abstract locomotion used in ALFRED, and poses a separate set of challenges (Blukis et al., 2019; Anderson et al., 2020). By operating only in simulation, our model also misses the full range of *experience* that can ground language in the world (Bisk et al., 2020), such as haptic feedback during object manipulation (Thomason et al., 2020, 2016; Sinapov et al., 2014), and audio (Chen et al., 2020a) and speech (Harwath et al., 2019; Ku et al., 2020) features of the environment. Further, in ALFRED an agent never encounters novel object classes at inference time, which represent an additional challenge for successful task completion (Suglia et al., 2020).

The ALFRED benchmark, and consequently the EmBERT model, only evaluates and considers written English. EmBERT inherently excludes people who cannot use typed communication. By training and evaluating only on English, we can only speculate whether the object-centric navigation methods introduced for EmBERT will generalize to other languages. We are cautiously optimistic that, with the success of massively multi-lingual language models (Pires et al., 2019), EmBERT would be able to train with non-English language data. At the same time, we acknowledge the possibility of pernicious, inscrutable priors and behavior (Bender et al., 2021) and the possibility for targeted, language prompt-based attacks (Song et al., 2021) in such large-scale networks.

References

Josh Abramson, Arun Ahuja, Arthur Brussee, Federico Carnevale, Mary Cassin, Stephen Clark, Andrew Dudzik, Petko Georgiev, Aurelia Guy, Tim Harley, Felix Hill, Alden Hung, Zachary Kenton, Jessica Landon, Timothy Lillicrap, Kory Mathewson, Alistair Muldal, Adam Santoro, Nikolay Savinov, Vikrant Varma, Greg Wayne, Nathaniel Wong, Chen Yan, and Rui Zhu. 2020. Imitating interactive intelligence. *arXiv*.

Peter Anderson, Ayush Shrivastava, Joanne Truong, Arjun Majumdar, Devi Parikh, Dhruv Batra, and Stefan Lee. 2020. Sim-to-real transfer for vision-and-language navigation. In *Conference on Robot Learning (CoRL)*.

Shurjo Banerjee, Jesse Thomason, and Jason J. Corso. 2020. The RobotSlang Benchmark: Dialog-guided robot localization and navigation. In *Conference on Robot Learning (CoRL)*.

Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. 2020a. Rearrangement: A challenge for embodied AI. In *arXiv*.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. 2020b. ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects. In *arXiv*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv*.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Association for Computational Linguistics (ACL)*.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience Grounds Language. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Valts Blukis, Dipendra Misra, Ross A. Knepper, and Yoav Artzi. 2018. Mapping navigation instructions to continuous control actions with position visitation prediction. In *Conference on Robot Learning (CoRL)*.

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. 2021. A persistent spatial semantic representation for high-level natural language instruction execution. In *Embodied AI Workshop CVPR*.

Valts Blukis, Yannick Terme, Eyvind Niklasson, Ross A. Knepper, and Yoav Artzi. 2019. Learning to map natural language instructions to physical quadcopter control using simulated flight. In *Conference on Robot Learning (CoRL)*.

Joyce Y. Chai, Qiaozi Gao, Lanbo She, Shaohua Yang, Sari Saba-Sadiya, and Guangyue Xu. 2018. Language to action: Towards interactive task learning with physical agents. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. 2020a. Soundspaces: Audio-visual navigaton in 3d environments. In *European Conference on Computer Vision (ECCV)*.

Kevin Chen, Junshen K. Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. 2021. Topological planning with transformers for vision-and-language navigation. In *Computer Vision and Pattern Recognition (CVPR)*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020b. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision (ECCV)*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. 2020. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Harm de Vries, Kurt Shuster, Dhruv Batra, Devi Parikh, Jason Weston, and Douwe Kiela. 2018. Talk the walk: Navigating new york city through grounded dialogue. *arXiv*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. 2018. Feature-wise transformations. *Distill*.

Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. 2019. Scene memory transformer for embodied agents in long-horizon tasks. In *Computer Vision and Pattern Recognition (CVPR)*.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Neural Information Processing Systems (NeurIPS)*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A deep semantic natural language processing platform. *arXiv*.

Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. 2017. Iqa: Visual question answering in interactive environments. In *Computer Vision and Pattern Recognition (CVPR)*.

David Harwath, Adrià Recasens, Dídac Surís, Galen Chuang, Antonio Torralba, and James Glass. 2019. Jointly discovering visual objects and spoken words from raw sensory input. *International Journal of Computer Vision*.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. 2017. Mask R-CNN. *International Conference on Computer Vision (ICCV)*.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *Computer Vision and Pattern Recognition (CVPR)*.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (GELUs). *arXiv*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2020. A recurrent vision-and-language BERT for navigation. *arXiv*.

Jared Sigurd Johansen, Thomas Victor Ilyevsky, and Jeffrey Mark Siskind. 2020. The Amazing Race TM: Robot Edition. *arXiv*.

Byeonghwi Kim, Suvaansh Bhambri, Kunal Pratap Singh, Roozbeh Mottaghi, and Jonghyun Choi. 2021. Agent with the big picture: Perceiving surroundings for interactive instruction following. In *Embodied AI Workshop CVPR*.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. 2017. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv*.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412.

Benjamin Kuipers and Yung-Tai Byun. 1991. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and autonomous systems*, 8(1-2):47–63.

Andrey Kurenkov, Roberto Martín-Martín, Jeff Ichnowski, Ken Goldberg, and Silvio Savarese. 2020. Semantic and geometric modeling with neural message passing in 3d scene graphs for hierarchical mechanical search. *arXiv*.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision (ECCV)*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Aly Magassouba, Komei Sugiura, and Hisashi Kawai. 2021. Crossmap transformer: A crossmodal masked path transformer using double back-translation for vision-and-language navigation. *arXiv*.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision (ECCV)*.

Van-Quang Nguyen, Masanori Suganuma, and Takayuki Okatani. 2021. Look wide and interpret twice: Improving performance on interactive instruction-following tasks. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic Transformer for Vision-and-Language Navigation. *arXiv*.

Telmo Pires, Eva Schlinger, , and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Association for Computational Linguistics (ACL)*.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *Computer Vision and Pattern Recognition (CVPR)*.

Yuankai Qi, Zizheng Pan, Yicong Hong, Ming-Hsuan Yang, Anton van den Hengel, and Qi Wu. 2021. Know what and know where: An object-and-room informed sequential BERT for indoor vision-language navigation. *arXiv*.

Yuankai Qi, Zizheng Pan, S. Zhang, A. V. Hengel, and Qi Wu. 2020a. Object-and-action aware model for visual language navigation. In *European Conference on Computer Vision (ECCV)*.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020b. Reverie: Remote embodied visual referring expression in real indoor environments. In *Computer Vision and Pattern Recognition (CVPR)*.

Willard Van Orman Quine. 1960. *Word and object*. MIT Press.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *arXiv*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *Computer Vision and Pattern Recognition (CVPR)*.

Ayush Shrivastava, Karthik Gopalakrishnan, Yang Liu, Robinson Piramuthu, Gokhan Tür, Devi Parikh, and Dilek Hakkani-Tür. 2021. VISITRON: Visual semantics-aligned interactively trained object-navigator. In *Visually Grounded Interaction and Language (ViGIL) Workshop @ NAACL*.

Jivko Sinapov, Connor Schenck, and Alexander Stoytchev. 2014. Learning relational object categories using behavioral exploration and multimodal perception. In *International Conference on Robotics and Automation (ICRA)*.

Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. 2020. MOCA: A modular object-centric approach for interactive instruction following. *arXiv*.

Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2021. Universal adversarial attacks with natural triggers for text classification. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Shane Storks, Qiaozi Gao, Govind Thattai, and Gokhan Tur. 2021. Are we there yet? learning to localize in embodied instruction following. In *HAI @ AAAI 2021*.

Alessandro Suglia, Antonio Vergari, Ioannis Konstas, Yonatan Bisk, Emanuele Bastianelli, Andrea Vanzo, and Oliver Lemon. 2020. Imagining grounded conceptual representations from perceptual information in situated guessing games. In *Conference on Computational Linguistics (COLING)*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Stefanie Tellex, Nakul Gopalan, Hadas Kress-Gazit, and Cynthia Matuszek. 2020. Robots that use language. *The Annual Review of Control, Robotics, and Autonomous Systems*, 15.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *AAAI Conference on Artificial Intelligence*.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2019. Vision-and-dialog navigation. In *Conference on Robot Learning (CoRL)*.

Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond J. Mooney. 2020. Jointly improving parsing and perception for natural language commands through human-robot dialog. *The Journal of Artificial Intelligence Research (JAIR)*, 67.

Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond J. Mooney. 2016. Learning multi-modal grounded linguistic semantics by playing “I spy”. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Jesse Thomason, Shiqi Zhang, Raymond Mooney, and Peter Stone. 2015. Learning to interpret natural language commands through human-robot dialog. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neural Information Processing Systems (NeurIPS)*.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv*.

Yichi Zhang and Joyce Chai. 2021. Hierarchical task learning from language instructions with unified transformers and self-monitoring. In *Findings of Association for Computational Linguistics (ACL Findings)*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. In *AAAI Conference on Artificial Intelligence*.

A Appendix

A.1 Additional Auxiliary Losses

In this section we describe alternative auxiliary losses that we designed for EmBERT training using ALFRED data. After validation, these configurations did not produce results comparable with the best performing model. This calls for a more detailed analysis of how to adequately design and combine such losses in the complex training regime of the ALFRED benchmark.

Masked Language Modeling The task-oriented language in ALFRED differs from the web crawl text used to train large-scale Transformers. We tune our initial model weights using a masked language modeling objective (Devlin et al., 2019). We mask with a $\%15$ probability a token among the ones in $\mathcal{I}_g$ and $\mathcal{I}_t$ at the very last step of a sub-goal. Differently from captions data or Wikipedia, *when* which such supervision should be provided is crucial. Given the instruction *Turn around and walk to the book on the desk*, at the very first timestep of the trajectory it is likely that none of the mentioned objects are visible. Thus, we assume that at the last step of a subgoal the agent will have in view the objects associated with the instruction. We apply the same conditional scaling approach to generate time-dependent language representations $\tilde{\mathbf{L}}$ as the one used in Equation 2. We denote the masked language modeling loss used for this task by $\mathcal{L}_{MLM}$.

Masked Region Modeling This is analogous to the Visual Region Classification (VCR) loss that we integrated in the model. The main difference is that 15% of the visual features are entirely masked (i.e., replaced with zero values) and we ask the model to predict them given the time-dependent representations generated by EmBERT for them.

Image-text Matching The masked region and language modeling losses encourage the model to learn fine-grained object and language token representations, respectively. However, we are also interested in global representations that are expressive enough to encode salient information of the visual frames. For this reason, we design an additional loss $\mathcal{L}_{IM}$. Given the state representation for the current timestep t, EmBERT predicts whether the current visual features can be associated with the corresponding language features or not. We maximize the cosine similarity between the visual features of the current timestep t and the corresponding language features while, at the same time, minimizing the cosine similarity between the current visual features and other language instructions in the same batch. In this task, just like when modeling the robot state, we use $\tilde{\mathbf{L}}_0$ as the language features and $\tilde{\mathbf{L}}_{m+n}$ as the visual features. We define $\mathcal{L}_{IM}$ the same way as the contrastive loss in CLIP (Radford et al., 2021). However, we expect the model to use the time-dependent representation of the agent state in order to truly understand the meaning of a language instruction. In this case the meaning of an instruction can be appreciated only after several timesteps when the corresponding sequence of actions has been executed.

A.2 EmBERT Asset Licenses

AI2THOR (Kolve et al., 2017) is released under the Apache-2.0 License, while the ALFRED benchmark (Shridhar et al., 2020) is released under the MIT License.