Known unknowns: Learning novel concepts using reasoning-by-elimination Supplementary Material

Harsh Agrawal[1,2] **Eli A. Meirom**[1] **Yuval Atzmon**[1] **Shie Mannor**[1] **Gal Chechik**[1]

[1]NVIDIA Research, Israel
[2]Georgia Tech, Georgia, USA

1 ARCHITECTURE DETAILS

For the textual encoder, we use the first three layers of a pretrained BERT model [Devlin et al., 2019] to extract the text encoding of the attributes mentioned in the instruction. We feed each attribute (color or shape) individually through a BERT model and use the 768-dimensional output feature corresponding to the [CLS] token as the textual encoding. The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token is often used as the aggregate sequence representation for classification tasks [Devlin et al., 2019]. We then project this vector to a more compact 32-dimensional vector. The projection module comprises of a linear layer followed by ReLU non-linearity, Layer Norm [Ba et al., 2016] and another linear layer.

The visual encoder is a CNN model comprised of three convolutional layers, each followed by ReLU non-linearity. The output of the final convolutional layer is flattened and passed through a linear layer followed by ReLU non-linearity to obtain a 512-dimensional vector. Similar to the text-encoder, we use a non-linear projection module to project the visual representation of the image, $\mathbf{x} \in \mathbb{R}^{512}$, to the individual shape and color representations $\mathbf{v}^S$, $\mathbf{v}^C \in \mathbb{R}^{32}$.

The policy obtains the 32-dimensional state-embedding from $4 \times |\mathcal{O}|$-dimensional object tuple by passing through a projection layer. It consists of two linear layers separated by a ReLU non-linearlity. Similarly, the agent's position embedding is obtained by learning an embedding layer that encodes the agent's position $p \in [0, 5)$ into a 16-dimensional vector. Finally, the 16-dimensional instruction embedding is obtained by passing the 2-dimensional instruction tuple through a similar projection layer. These three features are concatenated to form a 64-dimensional input to the recurrent policy. For the policy, we use a single layer GRU [Cho et al., 2014] with a 64 dimensional hidden state. The policy is parameterized by a GRU [Cho et al., 2014]. The GRU output is used to produce a softmax distribution over the action space $\mathcal{A}$ and an estimate of the value function.

2 HYPER-PARAMETERS

Next, we list down the important hyper-parameters associated with training the RL policy.

Hyperparameter	**Value**
PPO	
Clip parameter Schulman et al. [2017]	0.2
Rollout timesteps	128
Epochs	2
Value loss coefficient	0.5
Discount factor (γ)	0.99
GAE parameter (λ)	0.95
Normalize advantage	False
Training	
Optimizer	Adam
(β_1, β_2) for Adam	$(0.9, 0.999)$
Learning rate	$1.0e^{-4}$
Gradient clip norm	0.2

References

Jimmy Ba, J. Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.

Kyunghyun Cho, B. V. Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *ArXiv*, abs/1406.1078, 2014.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

John Schulman, F. Wolski, Prafulla Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.