WordCraft: An Environment for Benchmarking Commonsense Agents

Minqi Jiang [1] **Jelena Luketina** [2] **Nantas Nardelli** [2] **Pasquale Minervini** [1]
Philip H. S. Torr [2] **Shimon Whiteson** [2] **Tim Rocktäschel** [1]

Abstract

The ability to quickly solve a wide range of real-world tasks requires a commonsense understanding of the world. Yet, how to best extract such knowledge from natural language corpora and integrate it with reinforcement learning (RL) agents remains an open challenge. This is partly due to the lack of lightweight simulation environments that sufficiently reflect the semantics of the real world and provide knowledge sources grounded with respect to observations in an RL environment. To enable research on benchmarking agents with commonsense knowledge, we propose WordCraft, an RL environment based on LittleAlchemy2. This environment is small and fast to run, but built upon entities and relations inspired by real-world semantics. We evaluate several representation learning methods on this benchmarks and propose a new method for integrating knowledge graphs within an RL agent.

1. Introduction

Recent progress in Reinforcement Learning (RL) has been impressive (Silver et al., 2016; Vinyals et al., 2019; Berner et al., 2019), but it has mostly been fueled by *tabula-rasa* learning. For example, while Go is an extremely difficult game to master, its rules and thus environment dynamics are simple, not heavily relying on knowledge about concepts that can be encountered in the real world. Focusing on tabula-rasa learning confines state-of-the-art approaches to simulation environments that can be solved without requiring transfer of commonsense, world or domain knowledge.

Many real-world applications (*e.g.* personal assistants and household robots) require agents that can learn fast and generalise well to novel situations, something that is likely not possible without commonsense and general knowledge about the world. Consider, for example, an agent tasked with solving common household problems who has never seen a dirty ashtray. If presented with this new object, the agent needs to know that a reasonable set of actions include cleaning the ashtray, but not feeding it to the cat.

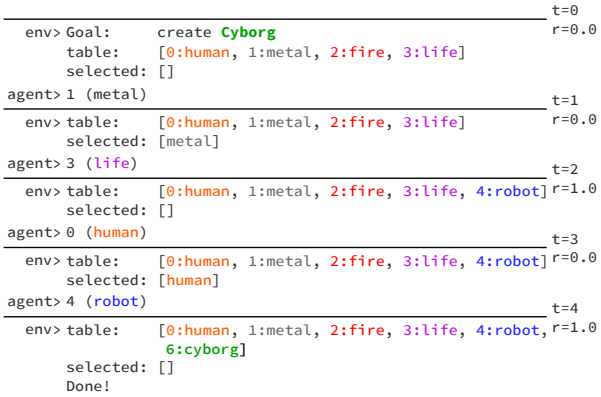

Figure 1. Sample episode of WordCraft: given a target entity (in this case, *cyborg*) and a collection of starting entities, the agent needs to combine the available entities so to create the target entity.

Humans communicate much of such knowledge through written language, for example, for example Wikipedia encodes such knowledge explicitly or implicitly through corpus statistics. Outside of RL, learning to represent and utilise such knowledge has improved considerably over the last years. For example, a common approach is to pre-train neural language models and fine-tuning them on downstream tasks (Devlin et al., 2019; Raffel et al., 2019). Recent works highlight that such pre-trained models capture several aspects of commonsense (Da & Kasai, 2019) and relational knowledge (Petroni et al., 2019), in addition to linguistic knowledge (Jawahar et al., 2019; Hewitt & Manning, 2019; Reif et al., 2019). Another approach is to represent commonsense knowledge explicitly in Knowledge Graphs (KGs), such as ATOMIC (Sap et al., 2019) and ConceptNet (Speer et al., 2017). We refer to Nickel et al. (2016) for an overview of knowledge base construction methods.

However, the question how to best utilise general or commonsense knowledge (either in texts or KGs) for downstream RL environments has not received the same level of

[*]Equal contribution [1]University College London, London, UK [2]University of Oxford, Oxford, UK. Correspondence to: Minqi Jiang <minqi.jiang.19@ucl.ac.uk>.

Accepted to the 1st Workshop on Language in Reinforcement Learning, ICML 2020, Copyright 2020 by the author(s).

attention, partly due scarce simulation environments that directly benefit from transferring such knowledge. Robotics environments such as Mujoco require knowledge that is not expressed in natural language such as intuitive understanding of physics (Todorov et al., 2012; Yu et al., 2019). Environments like Go or Chess (Silver et al., 2017; 2018), while very hard to master, are based on simple transition dynamics and largely disconnected from entities and concepts that one would encounter in the real world. On the other hand, environments such as (Vinyals et al., 2019; Berner et al., 2019) that contain many complex interactions with large number of entities (many of which have real-world analogues), commonsense knowledge could be helpful. However, running such environment is very costly and learning to ground knowledge from external sources is very challenging as natural language annotations of states and actions are not readily available.

To study agents with commonsense knowledge, we present WordCraft, a fast RL environment based on the popular game *Little Alchemy 2*. In this game, the player is tasked to craft over 700 different entities by combining already discovered entities. For example, combining "water" and "earth" creates the new entity "mud". Learning to play the game in a way that generalizes to unseen concepts requires commonsense and general knowledge about the world, yet the environment is quick to run which allows us to train agents for many steps (around 8,000 steps per second). As the entities in the game are described by words, they can be easily grounded with the external knowledge sources. In addition, we introduce a method for conditioning agents trained in WordCraft on knowledge graphs, which can be utilized for incorporating commonsense KGs such as ConceptNet.

2. Background

We employ a standard formulation of the RL problem: the agent-environment interface is framed as a Markov Decision Problem (MDP) (Sutton & Barto, 2018) defined by the tuple (S, A, T, R, γ), where $s \in S$ is a state, $a \in A$ is an action, $T(s, a) \rightarrow s'$ a transition function, $R(s, a) \rightarrow \mathbb{R}$ a reward function, and γ a discount factor. The goal is to find a policy π that maximises the expected cumulative return $\sum_{k=0}^{\infty} \gamma^k r_{k+1}$.

We represent knowledge about compounds and ingredients as a Knowledge Graph, *i.e.* a collection of $\langle s, p, o \rangle$ triples, each encoding a relationship of type p (predicate) between the subject s and the object o of the triple. In order to be able to work with incomplete knowledge graph and infer missing relationships, we use the ComplEx model (Trouillon et al., 2016) with the loss function and regulariser proposed by Lacroix et al. (2018) (see Appendix A for details).

3. WordCraft

To benchmark commonsense knowledge of RL agents, we present `WordCraft` based on *Little Alchemy 2*[1] which is a simple associative game where, starting from a set of four basic ingredients, the player must create as many new items as possible. Each new item can be created by combining an existing two items. For example, combining "moon" and "butterfly" yields "moth", whereas combining "human" and "medusa" yields "statue". There are in total 700 items (including rare and compound words) and 3277 permissible item combinations (referred to as valid recipes throughout this paper) (IGN, 2020). Solving the game without trying every possible combination of ingredients requires general knowledge of the world and the relationships between concepts.

WordCraft is based on entities and valid recipes from Little Alchemy 2, but otherwise significantly simplifies the problem. First, there is no graphical interface, the agent operates by selecting entities in a table. Second, instead of one long open-ended task, WordCraft contains a large number of smaller tasks. Each tasks is created by randomly sampling a goal entity, entities that can be combined towards this goal and distractors entities. The agent must choose which entities to combine in order to create the goal entity. The task difficulty can be varied by increasing the number of distractor items or increasing the number of intermediary entities that need to be created in order to reach the goal entity (we refer to this variant as *depth-*n, with $n-1$ intermediate entities). The training and testing split is created by selecting a set of test entities, goals or valid recipes which are not seen or required to complete the task during training. A relatively large number of entities and valid recipes enables the study of generalization to unseen goal entities.

The environment is deterministic and fully-observable. At each step, the state s consists of the goal entity $\mathbf{x}_g$, a set of k currently available entities $\mathbf{x}_{\text{table}} = \{\mathbf{x}_i\}_{i=0}^{k}$ and the currently selected entity $\mathbf{x}_{s,1}$. The action a corresponds to choosing one of the available entities as the second selected entity $\mathbf{x}_{s,2} \in \mathbf{x}_{\text{table}}$. When no entity is currently selected, $\mathbf{x}_{s,1}$ is set to None. If selected $(\mathbf{x}_{s,1}, \mathbf{x}_{s,2})$ corresponds to one of the valid recipes, a new entity is added to the table $\mathbf{x}_{\text{table}}$ and $\mathbf{x}_{s,1}$ is set to None. The episode terminates when the newly created entity is $\mathbf{x}_g$ or after a maximum number of steps is reached. The maximum number of steps is kept low to discourage brute-force solutions (for depth-1, trying out all possible pairs is $\mathcal{O}(k^2)$). The reward can be sparse (with $r = 1$ or 0 received only at the end of the episode) or shaped (with a penalty for choosing a wrong pair of items and reward for creating intermediary ingredients).

Compared to other environments that combine language and

[1] https://littlealchemy2.com/

RL (Janner et al., 2018; Chevalier-Boisvert et al., 2019; Côté et al., 2018; Das et al., 2018), WordCraft contains a much larger number of objects, with the relationships between the objects reflecting the structure of the real world. For example, the underlying knowledge graph of TextWorld (which is perhaps the closest comparison), contains about 99 different objects and 10 types of relations (Zelinka et al., 2019). Furthermore, WordCraft is lightweight and fast to train. It allows us to focus on incorporating commonsense knowledge without requiring the agent to learn to ground states to text or dealing with a combinatorially large action space (e.g. as in text games). Furthermore, the environment does not contain movement actions and the observation space is small, yet generalizing to novel entities is challenging.

4. Guiding Agents with a Knowledge Graph

We propose an agent architecture that makes use of information from an external knowledge graph to guide the agent's policy. At a high level, the model consists of a self-attention-based actor-critic network and an external knowledge-graph link prediction model. Given that the recipes in WordCraftare based on real-world semantics among common entities, conditioning on commonsense knowledge in a knowledge graph should enable agents to learn more efficiently by constraining their search space to policies biased toward interactions with underlying commonsense semantics.

4.1. Self-attention Actor-Critic Network

In order to handle a variable number of selectable entities at each time step and ensure order invariance of selection decisions with respect to these entities in the environment, our policy network makes use of a scaled dot-product attention mechanism introduced in (Vaswani et al., 2017). The self-attention policy network encodes the features $\mathbf{x}_g$ and current selection features $\mathbf{x}_{s,1:2}$ into attention weights over the set of selectable entities. These attention weights α_i act as the logits to the action distribution over the set of selectable entities e_i at each time step.

$$\alpha_i = \frac{1}{\sqrt{d_k}} Q([\mathbf{x}_g; \mathbf{x}_{s,1}; \mathbf{x}_{s,2}]) K(\mathbf{x}_{\text{table}})^\top$$
$$\pi(a_t = e_i | s_t = \mathbf{x}) = \frac{e^{\alpha_i}}{\sum_j e^{\alpha_j}}$$

where $\mathbf{x}_{\text{table}}$ are learned representations of selectable entities.

In order to predict value estimates, we first apply a linear transform W to $\mathbf{x}_{\text{table}}$ and encode the state as an attention-weighted sum of these linearly transformed entity features. This encoding is passed through a fully-connected layer to yield value estimates $V(\mathbf{x}) = \text{MLP}(\boldsymbol{\alpha}^\top W(\mathbf{x}_{\text{table}}))$.

4.2. Link Prediction Model

We assume access to a link prediction model $\mathbf{L}$ trained on an external knowledge graph that contains information relevant to the RL task. While our model makes use of ComplEx for link prediction (Trouillon et al., 2016), in general this choice can be substituted by any other link prediction model. We chose ComplEx for its reliable performance in practice.

The link prediction model is trained on a set of subject-object-predicate relation triplets of the form (s, p, o) to predict higher likelihood scores for true relation triplets. In the context of WordCraft, each recipe $\{e_1, e_2\} \rightarrow e_3$ reduces to four relevant relation triplets: $(e_1, \texttt{combinesWith}, e_2)$, $(e_2, \texttt{combinesWith}, e_1)$, $(e_1, \texttt{componentOf}, e_3)$, and $(e_2, \texttt{componentOf}, e_2)$. These triplets, when aggregated across all recipes, forms the recipe graph. In our experiments, we train ComplEx on a subgraph of the recipe graph containing all nodes to ensure full entity coverage.

At each time step, we compute relation score vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ between the goal and selected entities and each e_i on the table and $e_{s,j}$ in the selection: $\boldsymbol{u}_i = \mathbf{L}(e_i, \texttt{combinesWith}, e_{s,j})$, $\boldsymbol{v}_i = \mathbf{L}(e_i, \texttt{componentOf}, e_g)$. These scores are then mixed component-wise with the attention weights, before passing the mixed weights into the softmax policy head, thereby guiding the agent toward relevant predictions from the knowledge graph model. Details on learning the mixing coefficients are included in the Appendix.

5. Experiments

All of our experiments focus on zero-shot generalization performance. We split the set of all valid recipes into train (80%) and test (20%) sets. Train tasks are generated such that task goals does not involve any recipes in the test set. The model is trained using IMPALA/TorchBeast (Espeholt et al., 2018; Küttler et al., 2019).

5.1. WordCraft Benchmarks

To demonstrate how well the task captures the real-world relationships between entities, when evaluating the algorithm from Section 4.1, we will be using GloVe embeddings (Pennington et al., 2014). If semantically similar entities have similar uses in this environment, we expect GloVe to help generalization. We compare the zero-shot performance of our agent model without the link prediction module on depth-1 and depth-2 tasks with 1, 2, and 8 distractors.

We also collect a human baseline on the same difficulty settings of WordCraft. This human baseline serves as an estimate of the zero-shot performance that can be achieved using commonsense and general knowledge (see the Appendix for details about the procedure for human evaluation).

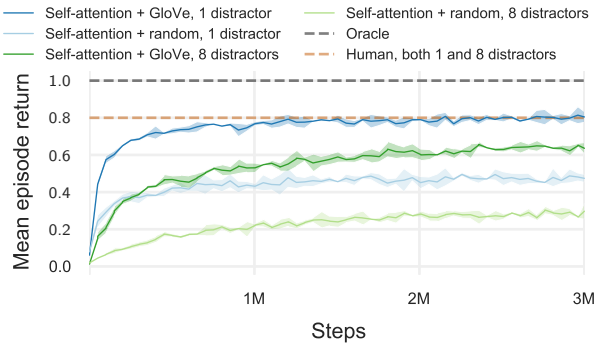

Figure 2. Zero-shot success rates over training steps of agents trained using different embeddings on depth-1 tasks with 1 and 8 distractors.

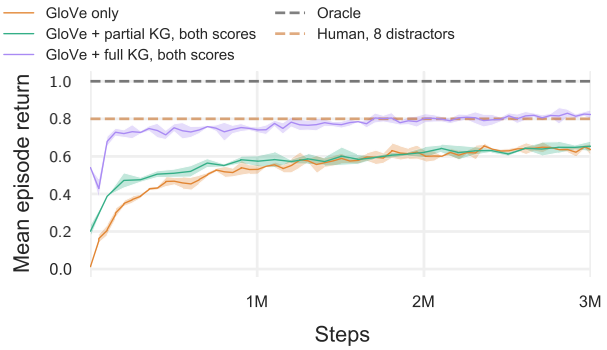

Figure 3. Zero-shot success rates over training steps of agents with full and partial KG models on depth-1 tasks with 8 distractors. The full KG model is trained on full recipe graph, while the partial KG model trained only on training recipes.

Agents trained with GloVe embeddings generalize much better to tasks with recipes that were not part of any training task (see Figure 2). This margin of improvement persists in harder task settings, even as the test success rate falls.

5.2. Knowledge Graph Benchmarks

We benchmark variants of our full agent with the link prediction module, and compare zero-shot performance under different combinations of link prediction scores used to modulate the self-attention weights: both `combinesWith` and `componentOf` scores, or each score in isolation. In these experiments, $\mathbf{L}$ is trained on the same train set of recipes used to generate the training tasks.

Our results show that access to a full KG model drastically improves the agent's zero-shot performance, as expected (see Figure 3). However, while partial-KG-model agents reach an equivalent zero-shot success rate as an agent without any KG model in fewer training steps, they ultimately reach comparable levels of test performance as training progresses (see Appendix for a more detailed discussion).

6. Related Work

Recently, several new environments that incorporate a degree of commonsense dynamics and text-base corpora have been proposed. For example, TextWorld (Côté et al., 2018) is a framework that enables to create novel textual RL environments as well as interfacing with existing commercial ones (such as Zork games). Chevalier-Boisvert et al. (2019); Küttler et al. (2020) respectively propose BabyAI and NLE, grid-world environments that evaluate agents on procedural, goal-conditioned tasks with varying degrees of commonsense structure and available textual corpora. These environments are interesting and challenging because of multiple reasons: complex dynamics, large action spaces, sparse rewards, long planning horizons. However, these are confounding elements do not easily enable us to isolate the problem of conditioning on external knowledge sources.

Another line of related work conditions policies on textual knowledge and/or knowledge bases, recently surveyed in Luketina et al. (2019). Fulda et al. (2017); Zahavy et al. (2018) demonstrate that word embeddings can be an effective modelling tool for policies, while Branavan et al. (2012); Narasimhan et al. (2018); Zhong et al. (2020) present work that directly conditions policies on domain knowledge encoded in textual corpora. Marzoev et al. (2020); Hill et al. (2020) show that language models can enable generalising to unseen subgoal-instructions, which indicates that at least some linguistic knowledge can be successfully transferred to RL. Closest to our work, Murugesan et al. (2020) concurrently propose to solve a set of TextWorld tasks using a model that encodes a combination of domain knowledge, via ConceptNet, and environment state into a knowledge graph, noting that while usable, knowledge graphs can provide a too strong prior.

7. Future Work

There are multiple avenues that we wish to further explore. WordCraft allows users to play Little Alchemy 2 as if the game had a very short horizon, however, the full game, in which the user a set of basic elements, and needs to discover as many entities as possible could be an interesting problem for RL as well. In addition, we are interested in including additional modalities such as images, as well as integrating alchemy into existing gridworld environments.

We are also interested in extending the environment to directly condition on textual knowledge sources. Furthermore, we believe the architecture we presented could have a strong impact in more complex RL tasks with access to large corpora of task-specific knowledge, such as NLE. Doing so requires automatically constructing knowledge graphs from available corpora, or further research on agents that can retrieve and condition on natural language texts.

References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. G. Dbpedia: A nucleus for a web of open data. In *ISWC/ASWC*, volume 4825 of *Lecture Notes in Computer Science*, pp. 722–735. Springer, 2007.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., de Oliveira Pinto, H. P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019.

Branavan, S. R. K., Silver, D., and Barzilay, R. Learning to Win by Reading Manuals in a Monte-Carlo Framework. *JAIR*, 2012.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. BabyAI: A Platform to Study the Sample Efficiency of Grounded Language Learning. In *ICLR*, 2019.

Côté, M., Kádár, Á., Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Hausknecht, M. J., Asri, L. E., Adada, M., Tay, W., and Trischler, A. Textworld: A learning environment for text-based games. *CoRR*, abs/1806.11532, 2018. URL http://arxiv.org/abs/1806.11532.

Da, J. and Kasai, J. Cracking the contextual commonsense code: Understanding commonsense reasoning aptitude of deep contextual representations. *CoRR*, abs/1910.01157, 2019.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. Embodied Question Answering. In *CVPR*, 2018.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.

Fulda, N., Ricks, D., Murdoch, B., and Wingate, D. What can you do with a rock? affordance extraction via word embeddings. *CoRR*, abs/1703.03429, 2017. URL http://arxiv.org/abs/1703.03429.

Hewitt, J. and Manning, C. D. A structural probe for finding syntax in word representations. In *NAACL-HLT (1)*, pp. 4129–4138. Association for Computational Linguistics, 2019.

Hill, F., Mokra, S., Wong, N., and Harley, T. Human instruction-following with deep reinforcement learning via transfer-learning from text. *arXiv preprint arXiv:2005.09382*, 2020.

IGN. Little Alchemy 2 Cheats - List of All Combinations - Little Alchemy 2 Wiki Guide - IGN. https://uk.ign.com/wikis/little-alchemy-2/Little_Alchemy_2_Cheats_-_List_of_All_Combinations, 2020. Online; accessed 8 June 2020.

Janner, M., Narasimhan, K., and Barzilay, R. Representation learning for grounded spatial reasoning. *TACL*, 2018.

Jawahar, G., Sagot, B., and Seddah, D. What does BERT learn about the structure of language? In *ACL (1)*, pp. 3651–3657. Association for Computational Linguistics, 2019.

Küttler, H., Nardelli, N., Lavril, T., Selvatici, M., Sivakumar, V., Rocktäschel, T., and Grefenstette, E. TorchBeast: A PyTorch Platform for Distributed RL. *arXiv preprint arXiv:1910.03552*, 2019. URL https://github.com/facebookresearch/torchbeast.

Küttler, H., Nardelli, N., Raileanu, R., Selvatici, M., Grefenstette, E., and Rocktäschel, T. The NetHack Learning Environment. In *Workshop on Beyond Tabula Rasa in Reinforcement Learning (BeTR-RL)*, 2020. URL https://github.com/facebookresearch/nle.

Lacroix, T., Usunier, N., and Obozinski, G. Canonical tensor decomposition for knowledge base completion. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2869–2878. PMLR, 2018.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., Whiteson, S., and Rocktaschel, T. A survey of reinforcement learning informed by natural language. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6309–6317. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

Marzoev, A., Madden, S., Kaashoek, M. F., Cafarella, M., and Andreas, J. Unnatural language processing: Bridging the gap between synthetic and natural language data. *arXiv preprint arXiv:2004.13645*, 2020.

Murugesan, K., Atzeni, M., Shukla, P., Sachan, M., Kapanipathi, P., and Talamadupula, K. Enhancing text-based reinforcement learning agents with commonsense knowledge. *arXiv preprint arXiv:2005.00811*, 2020.

Narasimhan, K., Barzilay, R., and Jaakkola, T. Grounding Language for Transfer in Deep Reinforcement Learning. *JAIR*, 2018.

Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016. doi: 10.1109/JPROC.2015.2483592.

Noy, N. F., Gao, Y., Jain, A., Narayanan, A., Patterson, A., and Taylor, J. Industry-scale knowledge graphs: lessons and challenges. *Commun. ACM*, 62(8):36–43, 2019.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *EMNLP*, 2014.

Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P. S. H., Bakhtin, A., Wu, Y., and Miller, A. H. Language models as knowledge bases? In *EMNLP/IJCNLP (1)*, pp. 2463–2473. Association for Computational Linguistics, 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683, 2019.

Reif, E., Yuan, A., Wattenberg, M., Viégas, F. B., Coenen, A., Pearce, A., and Kim, B. Visualizing and measuring the geometry of BERT. In *NeurIPS*, pp. 8592–8600, 2019.

Sap, M., Bras, R. L., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N. A., and Choi, Y. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *AAAI*, pp. 3027–3035. AAAI Press, 2019.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354, 2017.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Speer, R., Chin, J., and Havasi, C. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pp. 4444–4451. AAAI Press, 2017.

Suchanek, F. M., Kasneci, G., and Weikum, G. Yago: a core of semantic knowledge. In *WWW*, pp. 697–706. ACM, 2007.

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

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. Complex embeddings for simple link prediction. International Conference on Machine Learning (ICML), 2016.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention Is All You Need. 2017.

Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W. M., Dudzik, A., Huang, A., Georgiev, P., Powell, R., et al. Alphastar: Mastering the real-time strategy game starcraft ii. *DeepMind blog*, pp. 2, 2019.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *arXiv preprint arXiv:1910.10897*, 2019.

Zahavy, T., Haroush, M., Merlis, N., Mankowitz, D. J., and Mannor, S. Learn What Not to Learn: Action Elimination with Deep Reinforcement Learning. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 3562–3573. Curran Associates, Inc., 2018.

Zelinka, M., Yuan, X., Cote, M.-A., Laroche, R., and Trischler, A. Building dynamic knowledge graphs from text-based games. *arXiv preprint arXiv:1910.09532*, 2019.

Zhong, V., Rocktäschel, T., and Grefenstette, E. Rtfm: Generalising to new environment dynamics via reading. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgob6NKvH.

A. Background

Knowledge Graphs Knowledge graphs naturally represent the ontology of concepts underlying commonsense knowledge about the world. In a knowledge graph, nodes represent concepts, and edges between two nodes represent a relationship between the corresponding concepts. Several projects provide freely available commonsense knowledge graphs, including Freebase, DBPedia, and ConceptNet (Speer et al., 2017). Such large-scale knowledge graphs hold an enormous amount of useful prior knowledge about the world, that should be useful for guiding an RL agent in any environment with real-world semantics. Let $\mathcal{E}$ and $\mathcal{R}$ denote a set of entities and a set of relation types, respectively. Formally, a KG $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is a set of subject-predicate-object triples $\langle s, p, o \rangle$, with $s, o \in \mathcal{E}$ and $p \in \mathcal{R}$, where each triple encodes a relationship of type p between the subject s and the object o. There are several examples of large-scale KGs, both from academia (such as YAGO (Suchanek et al., 2007) and DBpedia (Auer et al., 2007)) and industry (Such as Microsoft's Bing Knowledge Graph and the Google Knowledge Graph (Noy et al., 2019)). Especially relevant to this work are ATOMIC (Sap et al., 2019) and ConceptNet (Speer et al., 2017), two KGs encoding common-sense knowledge.

Representation Learning of KGs Similarly to language, an effective way of enabling statistical learning on KGs consists in learning continuous, distributed representations, also referred to as *embeddings*, for all entities in a KG – we refer to Nickel et al. (2016) for a recent survey on this topic. In this work, we use the ComplEx (Trouillon et al., 2016) model, with the loss functions and regularisers proposed by Lacroix et al. (2018). Given a subject-predicate-object triple $\langle s, p, o \rangle \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, ComplEx defines the score $\phi(s, p, o)$ of such a triple as:

$$\phi(s, p, o) = \mathrm{Re}\left(\langle \mathbf{e}_s, \mathbf{e}_p, \overline{\mathbf{e}_o} \rangle\right), \tag{1}$$

where $\langle \cdot, \cdot, \cdot \rangle$ denotes the tri-linear dot product, $\mathbf{e}_s, \mathbf{e}_p, \mathbf{e}_o \in \mathbb{C}^k$ denote the complex-valued k-dimensional representations of s, p, and o, $\overline{\mathbf{x}}$ denote the complex conjugate of $\mathbf{x} \in \mathbb{C}^k$, $\mathrm{Re}\,(\mathbf{x})$ is the real part of $\mathbf{x}$. In order to learn the embedding representations Θ for all entities and relation types in a KG $\mathcal{G}$, we follow Lacroix et al. (2018) and minimise the following objective via gradient-based optimisation:

$$\begin{aligned}\mathcal{L}(\mathcal{G}) = \sum_{\langle s,p,o \rangle \in \mathcal{G}} & \left[\log \sum_{\hat{o} \in \mathcal{E}} \exp \phi(s, p, \hat{o}) \right] \\ & + \left[\log \sum_{\hat{s} \in \mathcal{E}} \exp \phi(\hat{s}, p, o) \right] - 2\phi(s, p, o),\end{aligned} \tag{2}$$

regularised via the weighted Nuclear p-norm regulariser proposed by Lacroix et al. (2018).

B. Model Details

B.1. Learning Mixing Coefficients

The link-prediction scores between goal and table entities for the `componentOf` relation and between each selected entity and table entity for the `combinesWith` relation are combined component-wise with the self-attention weights α_i using mixing coefficients to yield final policy logits β:

$$\beta_i = \lambda_\alpha \alpha_i + \lambda_u \boldsymbol{u}_i + \lambda_v \boldsymbol{v}_i \tag{3}$$

$$\pi(a_t = e_i | s_t = \mathbf{x}) = \frac{e^{\beta_i}}{\sum_j e^{\beta_j}}. \tag{4}$$

The mixing coefficients $\boldsymbol{\lambda}$ are computed by passing the self-attention-weighted table encoding through a fully-connected layer:

$$\boldsymbol{\lambda} = \mathrm{MLP}(\boldsymbol{\alpha}^T W_\lambda(\mathbf{x}_{table})) \tag{5}$$

C. Experiments

C.1. Training Parameters

Parameter	Value
Training set ratio	0.8
Discounting (γ)	0.99
Learning rate	0.001
Batch size	128
Unroll length	2
RMSProp ϵ	0.01
Entropy regularization	0.01
Reward clipping	none
Total steps	3,000,000

C.2. Model Parameters

Parameter	Value
Self-attention key size	300
Self-attention value size	300
ComplEx embedding size	128
Simple-MLP hidden size	300

C.3. Extended Discussion

We benchmark the performance of the self-attention actor-critic agent described in Section 4.1 with and without access to an instance of ComplEx trained on a subgraph of the full recipe graph. We compare agents trained with access to

ComplEx trained on a partial graph and those trained on the full graph. The partial KG models are trained on the relations within the set of training recipes.

As the method for incorporating ComplEx's predicted relation scores into the policy prediction performs a weighted sum of each score with the self-attention weights, we investigate the effect of only using different subsets of the scores. The subset of relation scores that are considered by the policy seem to vary across task depth and distractor settings.

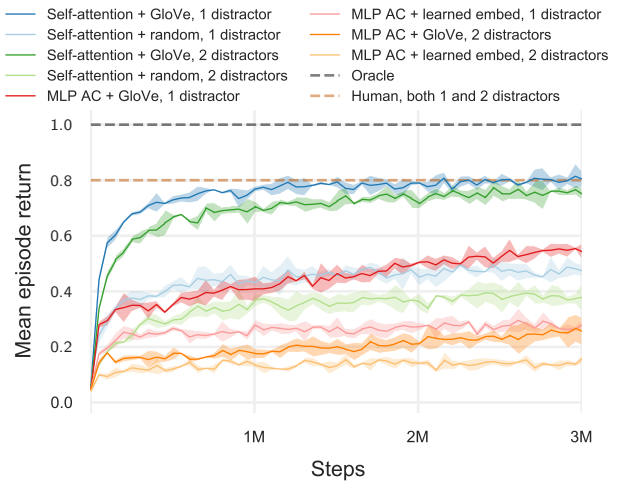

Figure 4. Zero-shot success rates of agents trained using different embeddings on depth-1 tasks with 1 and 2 distractors.

We generally find that access to a full KG model drastically improves the agent's zero-shot success rate, as might be expected. In fact, one might expect that having access to the ground-truth recipe graph as encoded by ComplEx should enable the agent to achieve 100% zero-shot success rates across all task settings. In order to test the extent to which the full-KG ComplEx scores are predictive of the best actions, we also assessed the performance of agents whose final policy logit mixture only assigned positive weights to different subsets of the relation scores. Such full-KG-only agents that choose items only based on the `combinesWith` relation score will still often choose the same entity twice, believing the entity combines with itself. This is likely because the recipe graph includes several entities that combine with themselves, while the set of all self-combination triplets are not included in the training set.

Full-KG-only agents that choose items only based on the `componentOf` relation score will tend to choose the same item twice, as this approach will always assign the highest action probability on selecting the entity on the table that ComplEx predicts to be most likely to be a component of the goal entity.

Interestingly, partial-KG-agents do not always do much better than a KG-free agent. This implies that as the KG-free agent sees more and more training tasks, it becomes comparable with ComplEx in predicting unseen relations among entities.

Early on in training, the relation scores predicted by ComplEx seem to help the agent achieve higher zero-shot success rates in fewer training steps compared to KG-free agents (see Figure 6).

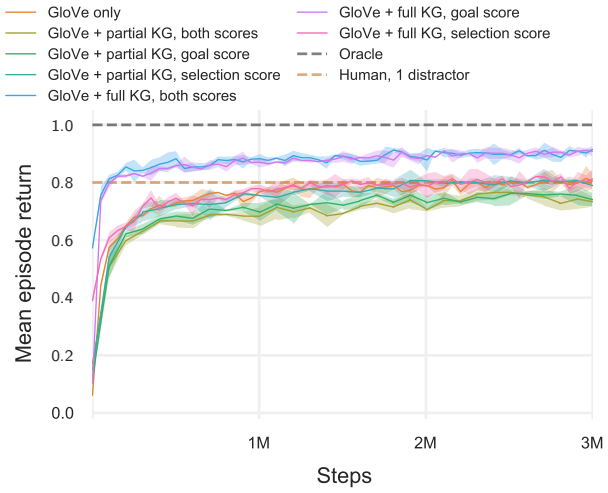

Figure 5. Zero-shot success rates of variations of partial- and full-KG agents on depth-1 tasks with 1 distractor. These results show the effect of including each of the two relation scores from ComplEx in modulating the policy logits.

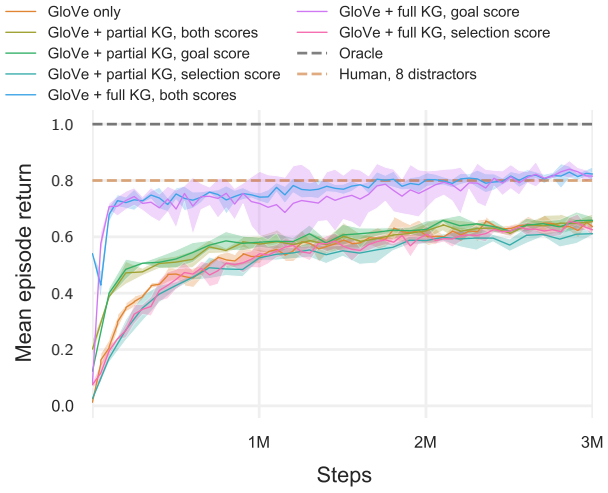

Figure 6. Zero-shot success rates of agents trained with and without KG models on depth-1 tasks with 8 distractors. Full KG refers to a ComplEx model trained on the full recipe graph, while partial KG refers to one trained on only training recipes.

C.4. Human Evaluation Protocol

We estimate generalization ability of humans players on depth-1 tasks with 1 and 8 distractors. Each difficulty setting is tested on a different individual, and we test one individual per setting. The selected individuals were not familiar with the game and got to train on 40 randomly sampled training tasks before being evaluated on 40 testing tasks. The train and test split was the same as in the RL experiments. Surprisingly, we found that changing the number of distractors from 1 to 8 did not significantly influence human performance.

D. Example recipes

Resulting entity	Entity combination(s)
airplane	bird metal; bird, steel
alcohol	fruit, time; juice, time
batter	flour, milk
cereal	wheat, milk
catnip	cat, plant
charcoal	fire, wood
dew	water, grass; fog, grass
farmer	human, field; human, plant; human, pitchfork
geyser	steam, earth
glacier	ice, mountain
hay bale	hay, hay
iced tea	ice, tea
ivy	plant, wall
juice	water, fruit; pressure, fruit
kite	wind, paper; sky, paper
lake	water, pond; river, dam
milk	farmer, cow; cow, human
milk shake	milk, ice cream
narwhal	unicorn, ocean; unicorn, water; sea, unicorn
oasis	desert, water
paper	wood, pressure
pasta	flour, egg
rainbow	rain, sun; rain, light
reindeer	Santa, wild animal; livestock, Santa
sand castle	sand, castle
Santa	human, Christmas tree
scythe	blade, grass; blade, wheat
telescope	glass, sky; glass, star; glass, space
umbrella	tool, rain; rain, fabric
volcano	lava, earth; lava, mountain; fire, mountain
wallet	leather, money
watch	human, clock
x-ray	light, bone; light, skeleton
yogurt	milk, bacteria
zombie	corpse, life