Deep Reinforcement Learning with a Combinatorial Action Space for Predicting Popular Reddit Threads

Ji He[*], **Mari Ostendorf**[*], **Xiaodong He**[†], **Jianshu Chen**[†], **Jianfeng Gao**[†], **Lihong Li**[†], **Li Deng**[†]

[*]Department of Electrical Engineering, University of Washington, Seattle, WA 98195, USA
{jvking, ostendor}@uw.edu
[†]Microsoft Research, Redmond, WA 98052, USA
{xiaohe, jianshuc, jfgao, lihongli, deng}@microsoft.com

Abstract

We introduce an online popularity prediction and tracking task as a benchmark task for reinforcement learning with a combinatorial, natural language action space. A specified number of discussion threads predicted to be popular are recommended, chosen from a fixed window of recent comments to track. Novel deep reinforcement learning architectures are studied for effective modeling of the value function associated with actions comprised of *interdependent* sub-actions. The proposed model, which represents dependence between sub-actions through a bi-directional LSTM, gives the best performance across different experimental configurations and domains, and it also generalizes well with varying numbers of recommendation requests.

1 Introduction

This paper is concerned with learning policies for sequential decision-making tasks, where a system takes actions given options characterized by natural language with the goal of maximizing a long-term reward. More specifically, we consider tasks with a combinatorial action space, where each action is a set of multiple interdependent sub-actions. The problem of a combinatorial natural language action space arises in many applications. For example, in real-time news feed recommendation, a user may want to read diverse topics of interest, and an action (i.e. recommendation) from the computer agent would consist of a set of news articles that are not all similar in topics (Yue and Guestrin, 2011). In advertisement placement, an action is a selection of sev-

eral ads to display, and bundling with complementary products might receive higher click-through-rate than displaying all similar popular products.

In this work, we consider Reddit popularity prediction, which is similar to newsfeed recommendation but different in two respects. First, our goal is not to make recommendations based on an individual's preferences, but instead based on the anticipated long-term interest level of a broad group of readers from a target community. Second, we try to predict rather than detect popularity. Unlike individual interests, community interest level is not often immediately clear; there is a time lag before the level of interest starts to take off. Here, the goal is for the recommendation system to identify and track written documents (e.g. news articles, comments in discussion forum threads, or scientific articles) in real time – attempting to identify hot updates before they become hot to keep the reader at the leading edge. The premise is that the user's bandwidth is limited, and only a limited number of things can be recommended out of several possibilities. In our experimental work, we use discussion forum text, where the recommendations correspond to recent posts or comments, assessing interest based on community response as observed in "likes" or other positive reactions to those comments. For training purposes, we can use community response measured at a time much later than the original post or publication. This problem is well-suited to the reinforcement learning paradigm, since the reward (the level of community uptake or positive response) is not immediately known, so the system needs to learn a mechanism for estimating future reactions. Different from

typical reinforcement learning, the action space is combinatorial since an action corresponds to a set of comments (sub-actions) chosen from a larger set of candidates. A sub-action is a written comment (or document, for another variant of this task).

Two challenges associated with this problem include the potentially high computational complexity of the combinatorial action space and the development of a framework for estimating the long-term reward (the Q-value in reinforcement learning) from a combination of sub-actions characterized by natural language. Here, we focus on the second problem, exploring different deep neural network architectures in an effort to efficiently account for the potential redundancy and/or temporal dependency of different sub-actions in relation to the state space. We sidestep the computational complexity issue (for now) by working with a task where the number of combinations is not too large and by further reducing costs by random sampling.

There are two main contributions in this paper. First, we propose a novel reinforcement learning task with both states and combinatorial actions defined by natural language,[1] which is introduced in section 2. This task, which is based on comment popularity prediction using data from the Reddit discussion forum, can serve as a benchmark in social media recommendation and trend spotting. The second contribution is the development of a novel deep reinforcement learning architecture for handling a combinatorial action space associated with natural language. Prior work related to both the task and deep reinforcement learning is reviewed in section 3, Details for the new models and baseline architectures are described in section 4. Experimental results in section 5 show the proposed methods outperform baseline models and that a bidirectional LSTM is effective for characterizing the combined utility of sub-actions. A brief summary of findings and open questions are in section 6.

2 Popularity Prediction and Tracking

Our experiments are based on Reddit[2], one of the world's largest public discussion forums. On Red-
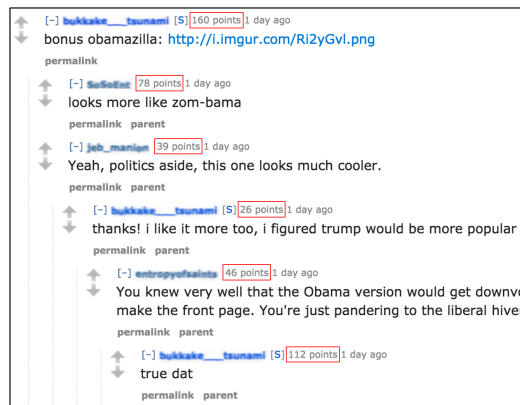

Figure 1: A snapshot of the top of a Reddit discussion tree, where karma scores are shown in red boxes.

dit, registered users initiate a post and people respond with comments, either to the original post or one of its associated comments. Together, the comments and the original post form a discussion tree, which grows as new comments are contributed. It has been show that discussions tend to have a hierarchical topic structure (Weninger et al., 2013), i.e. different branches of the discussion reflect narrowing of higher level topics. Reddit discussions are grouped into different domains, called subreddits, according to different topics or themes. Depending on the popularity of the subreddit, a post can receive hundreds of comments.

Comments (and posts) are associated with positive and negative votes (i.e., likes and dislikes) from registered users that are combined to get a *karma score*, which can be used as a measure for popularity. An example of the top of a Reddit discussion tree is given in Figure 1. The scores in red boxes mark the current karma (popularity) of each comment, and it is quite common that a lower karma comment (e.g. "Yeah, politics aside, this one looks much cooler", compared to "looks more like zom-bama") will lead to more children and popular comments in the future (e.g. "true dat"). Note that the karma scores are dynamic, changing as readers react to the evolving discussion and eventually settling down as the discussion trails off. In a real-time comment recommendation system, the eventual karma of a comment is not immediately available, so prediction of popularity is based on the text in the comment in the context of prior comments in the subtree and other comments in the current time window.

Popularity prediction and tracking in the Reddit setting is used in this paper for studying reinforcement learning to model long-term rewards in a combinatorial action space. At each time step, the state corresponds to the collection of comments previously recommended. The system aims at automatically picking a few lines of the discussion to follow from the new set of comments in a given window, which is a combinatorial action. Thread popularity tracking can be thought of as a proxy task for news or scientific article recommendation. It has the advantages that "documents" (comments) are relatively short and that the long-term reward can be characterized by Reddit voting scores, which makes this task easier to work with for algorithm development than these larger related tasks.

In this work, we only consider new comments associated with the threads of the discussion that we are currently following to limit the number of possible sub-actions at each time step and with the assumption that prior context is needed to interpret the comments. In other words, the new recommendation should focus on comments that are in the subtrees of previously recommended comments. (A variant relaxing this restriction is suggested in the conclusion section.) Typically, one would expect some inter-dependencies between comments made in the same window if they fall under the same subtree, because they correspond to a reply to the same parent. In addition, there may be some temporal dependency, since one sub-action may be a comment on the other. These dependencies will affect the combined utility of the sub-actions.

According to our experiments, the performance is significantly worse when we learn a myopic policy compared to reinforcement learning with the same feature set. This shows that long-term dependency indeed matters, as illustrated in Figure 1. This serves as a justification that reinforcement learning is an appropriate approach for modeling popularity of a discussion thread.

3   Related Work

There is a large body of work on reinforcement learning. Among those of most interest here are deep reinforcement learning methods that leverage neural networks because of their success in handling large discrete state/action spaces. Early work such as TD-gammon used a neural network to approximate the state value function (Tesauro, 1995). Recent advances in deep learning (LeCun et al., 2015; Deng and Yu, 2014; Hinton et al., 2012; Krizhevsky et al., 2012; Sordoni et al., 2015) inspired significant progress by combining deep learning with reinforcement learning (Mnih et al., 2015; Silver et al., 2016; Lillicrap et al., 2016; Duan et al., 2016). In natural language processing, reinforcement learning has been applied successfully to dialogue systems that generate natural language and converse with a human user (Scheffler and Young, 2002; Singh et al., 1999; Wen et al., 2016). There has also been interest in mapping text instructions to sequences of executable actions and extracting textual knowledge to improve game control performance (Branavan et al., 2009; Branavan et al., 2011).

Recently, Narasimhan et al. (2015) studied the task of text-based games with a deep Q-learning framework. He et al. (2016) proposed to use a separate deep network for handling natural language actions and to model Q-values via state-action interaction. Nogueira and Cho (2016) have also proposed a goal-driven web navigation task for language-based sequential decision making. Narasimhan et al. (2016) applied reinforcement learning for acquiring and incorporating external evidence to improve information extraction accuracy. The study that we present with Reddit popularity tracking differs from these other text-based reinforcement learning tasks in that the language in both state and action spaces is unconstrained and quite rich.

Dulac-Arnold et al. (2016) also investigated a problem of large discrete action spaces. A Wolpertinger architecture is proposed to reduce computational complexity of evaluating all actions. While a combinatorial action space can be large and discrete, their method does not directly apply in our case, because the possible actions are changing over different states. In addition, our work differs in that its focus is on modeling the combined action-value function rather than on reducing computational complexity. Other work that targets a structured action space includes: an actor-critic algorithm, where actions can have real-valued parameters (Hausknecht and Stone, 2016); and the factored Markov Decision Process (MDP) (Guestrin et al., 2001; Sallans and

Hinton, 2004), with certain independence assumptions between a next-state component and a sub-action. As for a bandits setting, Yue and Guestrin (2011) considered diversification of multi-item recommendation, but their methodology is limited to using linear approximation with hand-crafted features.

The task explored in our paper – detecting and tracking popular threads in a discussion – is somewhat related to topic detection and tracking (Allan, 2012; Mathioudakis and Koudas, 2010), but it differs in that the goal is not to track topics based on frequency, but rather based on reader response. Thus, our work is more closely related to popularity prediction for social media and online news. These studies have explored a variety of definitions (or measurements) of popularity, including: the volume of comments in response to blog posts (Yano and Smith, 2010) and news articles (Tasgkias et al., 2009; Tatar et al., 2011), the number of Twitter shares of news articles (Bandari et al., 2012), the number of reshares on Facebook (Cheng et al., 2014) and retweets on Twitter (Suh et al., 2010; Hong et al., 2011; Tan et al., 2014; Zhao et al., 2015), the rate of posts related to a source rumor (Lukasik et al., 2015), and the difference in the number of reader up and down votes on posts and comments in Reddit discussion forums (Lakkaraju et al., 2013; Jaech et al., 2015). An advantage of working with the Reddit data is that both positive and negative reactions are accounted for in the karma score. Of the prior work on Reddit, the task explored here is most similar to (Jaech et al., 2015) in that it involves choosing relatively high karma comments (or threads) from a time-limited set rather than directly predicting comment (or post) karma. Prior work on popularity prediction used supervised learning; this is the first work that frames tracking hot topics in social media with deep reinforcement learning.

4 Characterizing a combinatorial action space

4.1 Notation

In this sequential decision making problem, at each time step t, the agent receives a text string that describes the state $s_t \in \mathcal{S}$ (i.e., "state-text") and picks a text string that describes the action $a_t \in \mathcal{A}$ (i.e.,

"action-text"), where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. Here, we assume a_t is chosen from a set of given candidates. In our case both $\mathcal{S}$ and $\mathcal{A}$ are described by natural language. Given the state-text and action-texts, the agent aims to select the best action in order to maximize its long-term reward. Then the environment state is updated $s_{t+1} = s'$ according to a probability $p(s'|s, a)$, and the agent receives a reward r_{t+1} for that particular transition. We define the action-value function (i.e. the Q-function) $Q(s, a)$ as the expected return starting from s and taking the action a:

$$Q(s,a) = \mathbb{E}\left\{\sum_{l=0}^{+\infty} \gamma^l r_{t+1+l} | s_t = s, a_t = a\right\}$$

where $\gamma \in (0, 1)$ denotes a discount factor. The Q-function associated with an optimal policy can be found by the Q-learning algorithm (Watkins and Dayan, 1992):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta_t \cdot \left(r_{t+1} + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right)$$

where η_t is a learning rate parameter.

The set of comments that are being tracked at time step t is denoted as M_t. All previously tracked comments, as well as the post (root node of the tree), is considered as state s_t ($s_t = \{M_0, M_1, \cdots, M_t\}$), and we initialize $s_0 = M_0$ to be the post. An action is taken when a total of N new comments $\{c_{t,1}, c_{t,2}, \cdots, c_{t,N}\}$ appear as nodes in the subtree of M_t, and the agent picks a set of K comments to be tracked in the next time step $t + 1$. Thus we have:

$$a_t = \{c_t^1, c_t^2, \cdots, c_t^K\}, \quad c_t^i \in \{c_{t,1}, c_{t,2}, \cdots, c_{t,N}\}$$
$$\text{and } c_t^i \neq c_t^j \text{ if } i \neq j \quad (1)$$

and $M_{t+1} = a_t$. At the same time, by taking action a_t at state s_t, the reward r_{t+1} is the accumulated karma scores, i.e. sum over all comments in M_{t+1}. Note that the reward signal is used in online training, while at model deployment (testing stage), the scores are only used as an evaluation metric.

Following the reinforcement learning tradition, we call tracking of a single discussion tree from start (root node post) to end (no more new comments appear) an *episode*. We also randomly partition all

discussion trees into separate training and testing sets, so that texts seen by the agent in training and testing are from the same domain but different discussions. For each episode, depending on whether training/testing, the simulator randomly picks a discussion tree, and presents the agent with the current state and N new comments.

4.2 Q-function alternatives

With the real-time setting, it is clear that action a_t will affect the next state s_{t+1} and furthermore the future expected reward. The action a_t consists of K comments (sub-actions), making modeling Q-values $Q(s_t, a_t)$ difficult. To handle a large state space, Mnih et al. (2015) proposed a Deep Q-Network (DQN). In case of a large action space, we may use both state and action representations as input to a deep neural network. It is shown that the Deep Reinforcement Relevance Network (DRRN, Figure 2(b)), i.e. two separate deep neural networks for modeling state embedding and action embedding, performs better than per-action DQN (PA-DQN in Figure 2(a)), as well as other DQN variants for dealing with natural language action spaces (He et al., 2016).

Our baseline models include Linear, PA-DQN and DRRN. We concatenate the K sub-actions/comments to form the action representation. The Linear and PA-DQN (Figure 2(a)) take as input a concatenation of state and action representations, and model a single Q-value $Q(s_t, a_t)$ using linear or DNN function approximations. The DRRN consists of a pair of DNNs, one for the state-text embedding and the other for action-text embeddings, which are then used to compute $Q(s_t, a_t)$ via a pairwise interaction function (Figure 2(b)).

One simple alternative approach by utilizing this combinatorial structure is to compute an embedding for each sub-action c_t^i. We can then model the value in picking a particular sub-action, $Q(s_t, c_t^i)$, through a pairwise interaction between the state and this sub-action. $Q(s_t, c_t^i)$ represents the expected accumulated future rewards by including this sub-action. The agent then greedily picks the top-K sub-actions with highest values to achieve the highest $Q(s_t, a_t)$. In this approach, we are assuming the long-term rewards associated with sub-actions are independent of each other. More specifically, greedily picking

the top-K sub-actions is equivalent to maximizing the following action-value function:

$$Q(s_t, a_t) = \sum_{i=1}^{K} Q(s_t, c_t^i) \qquad (2)$$

while satisfying (1). We call this proposed method DRRN-Sum, and its architecture is shown in Figure 2(c). Similarly as in DRRN, we use two networks to embed state and actions separately. However, for different sub-actions, we keep the network parameters tied. We also use the same top layer dimension and the same pairwise interaction function for all sub-actions.

In the case of a linear additive interaction, such as an inner product or bilinear operation, Equation (2) is equivalent to computing the interaction between the state embedding and an action embedding, where the action embedding is obtained linearly by summing over K sub-action embeddings. When sub-actions have strong correlation, this independence assumption is invalid and can result in a poor estimation of $Q(s_t, a_t)$. For example, most people are interested in the total information stored in the combined action a_t. Due to content redundancy in the sub-actions $c_t^1, c_t^2, \cdots, c_t^K$, we expect $Q(s_t, a_t)$ to be smaller than $\sum_i Q(s_t, c_t^i)$.

To come up with a general model for handling a combinatorial action-value function, we further propose the DRRN-BiLSTM (Figure 2(d)). In this architecture, we use a DNN to generate an embedding for each comment. Then a Bidirectional Long Short-Term Memory (Graves and Schmidhuber, 2005) is used to combine a sequence of K comment embeddings. As the Bidirectional LSTM has a larger capacity due to its nonlinear structure, we expect it will capture more details on how the embeddings for the sub-actions combine into an action embedding. Note that both of our proposed methods (DRRN-Sum and DRRN-BiLSTM) can handle a varying value of K, while for the DQN and DRRN baselines, we need to use a fixed K in training and testing.

5 Experiments

5.1 Datasets and Experimental Configurations

Our data consists of 5 subreddits (askscience, askmen, todayilearned, worldnews, nfl) with diverse

(a) Per-action DQN

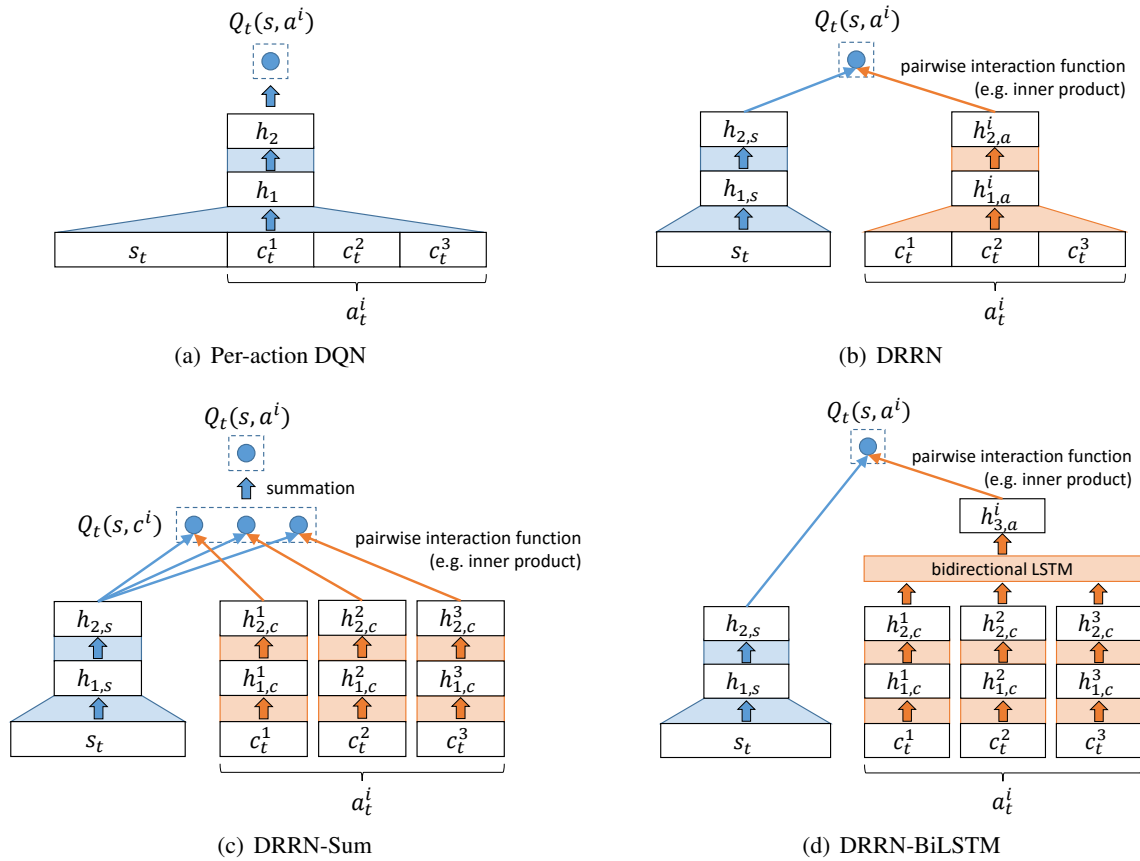(b) DRRN

(c) DRRN-Sum

(d) DRRN-BiLSTM

Figure 2: Different deep Q-learning architectures

topics and genres. In our experiments, in order to have long enough discussion threads, we filter out discussion trees with fewer than 100 comments. For each subreddit, we randomly partition 90% of the data for online training, and 10% of the data for testing (deployment). The basic subreddit statistics are shown in Table 1. We report the random policy performances and heuristic upper bound performances (averaged over 10,000 episodes) in Table 2 and Table 3.[3] The upper bound performances are obtained using stabilized karma scores and offline constructed tree structure. The mean and standard deviation are obtained by 5 independent runs.

In all our experiments we set $N = 10$. Explicitly representing all N-choose-K actions requires a lot of memory and does not scale up. We therefore use a variant of Q-learning: when taking the max over

Subreddit	# Posts (in k)	# Comments (in M)
askscience	0.94	0.32
askmen	4.45	1.06
todayilearned	9.44	5.11
worldnews	9.88	5.99
nfl	11.73	6.12

Table 1: Basic statistics of filtered subreddit data sets

Subreddit	Random	Upper bound
askscience	321.3 (7.0)	2109.0 (16.5)
askmen	132.4 (0.7)	651.4 (2.8)
todayilearned	390.3 (5.7)	2679.6 (30.1)
worldnews	205.8 (4.5)	1853.4 (44.4)
nfl	237.1 (1.4)	1338.2 (13.2)

Table 2: Mean and standard deviation of random and upper-bound performance (with $N = 10, K = 3$) across different subreddits.

---

[3]Upper bounds are estimated by greedily searching through each discussion tree to find K max karma discussion threads (overlapped comments are counted only once). This upper bound may not be attainable in a real-time setting.

1843

K	Random	Upper bound
2	201.0 (2.1)	1991.3 (2.9)
3	321.3 (7.0)	2109.0 (16.5)
4	447.1 (10.8)	2206.6 (8.2)
5	561.3 (18.8)	2298.0 (29.1)

Table 3: Mean and standard deviation of random and upper-bound performance on askscience, with $N = 10$ and $K = 2, 3, 4, 5$.

possible next-actions, we instead randomly subsample m' actions and take the max over them. We set $m' = 10$ throughout our experiments. This heuristic technique works well in our experiments.

For text preprocessing we remove punctuation and lowercase capital letters. For each state s_t and comment c_t^i, we use a bag-of-words representation with the same vocabulary in all networks. The vocabulary contains the most frequent 5,000 words; the out-of-vocabulary rate is 7.1%.

In terms of the Q-learning agent, fully-connected neural networks are used for text embeddings. The network has $L = 2$ hidden layers, each with 20 nodes, and model parameters are initialized with small random numbers. ϵ-greedy is used for exploration-exploitation, and we keep $\epsilon = 0.1$ throughout online training and testing. We pick the discount factor $\gamma = 0.9$. During online training, we use experience replay (Lin, 1992) and the memory size is set to 10,000 tuples of $(s_t, a_t, r_{t+1}, s_{t+1})$. For each experience replay, 500 episodes are generated and stored in a first-in-first-out fashion, and multiple epochs are trained for each model. Minibatch stochastic gradient descent is implemented with a batch size of 100. The learning rate is kept constant: $\eta_t = 0.000001$.

The proposed methods are compared with three baseline models: Linear, per-action DQN (PA-DQN), and DRRN. For both Linear and PA-DQN, the state and comments are concatenated as an input. For the DRRN, the state and comments are sent through two separate deep neural networks. However, in our baselines, we do not explicitly model how values associated with each comment are combined to form the action value. For the DRRN baseline and proposed methods (DRRN-Sum and DRRN-BiLSTM), we use an inner product as the pairwise interaction function.
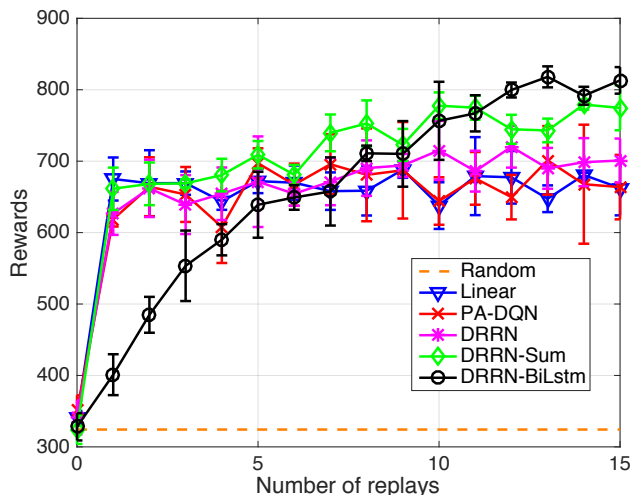

Figure 3: Learning curves of baselines and proposed methods on "askscience"

5.2 Experimental Results

In Figure 3 we provide learning curves of different models on the askscience subreddit during online learning. In this experiment, we set $N = 10, K = 3$. Each curve is obtained by averaging over 3 independent runs, and the error bars are also shown. All models start with random performance, and converge after approximately 15 experience replays. The DRRN-Sum converges as fast as baseline models, with better converged performance. DRRN-BiLSTM converges slower than other methods, but with the best converged performance.

After we train all the models on the training set, we fix the model parameters and apply (deploy) on the test set, where the models predict which action to take but no reward is shown until evaluation. The test performance is averaged over 1000 episodes, and we report mean and standard deviation over 5 independent runs.

On askscience, we try multiple settings with $N = 10, K = 2, 3, 4, 5$ and the results are shown in Table 4. Both DRRN-Sum and DRRN-BiLSTM consistently outperform baseline methods. The DRRN-BiLSTM performs better with larger K, probably due to the greater chance of redundancy in combining more sub-actions.

We also perform online training and test across different subreddits. With $N = 10, K = 3$, the test performance gains over the linear baseline are shown in Figure 4. Again, the test performance is

K	Linear	PA-DQN	DRRN	DRRN-Sum	DRRN-BiLSTM
2	553.3 (2.8)	556.8 (14.5)	553.0 (17.5)	569.6 (18.4)	**573.2 (12.9)**
3	656.2 (22.5)	668.3 (19.9)	694.9 (15.5)	704.3 (20.1)	**711.1 (8.7)**
4	812.5 (23.4)	818.0 (29.9)	828.2 (27.5)	829.9 (13.2)	**854.7 (16.0)**
5	861.6 (28.3)	884.3 (11.4)	921.8 (10.7)	942.3 (19.1)	**980.9 (21.1)**

Table 4: On askscience, average karma scores and standard deviation of baselines and proposed methods (with $N = 10$)
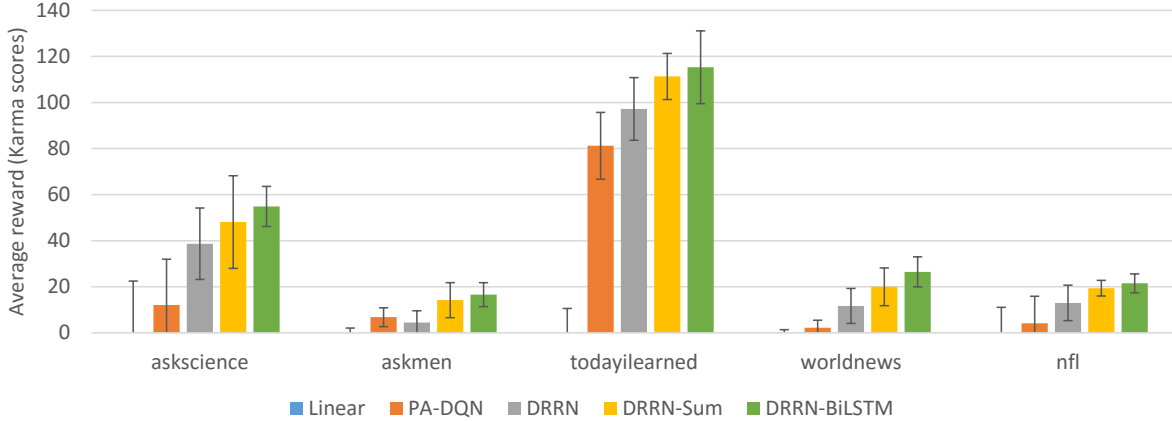

Figure 4: Average karma score gains over the linear baseline and standard deviation across different subreddits (with $N = 10, K = 3$).

K	DRRN-Sum	DRRN-BiLSTM
2	538.5 (18.9)	**551.2 (10.5)**
4	819.1 (14.7)	**829.9 (11.1)**
5	921.6 (15.6)	**951.3 (15.7)**

Table 5: On askscience, average karma scores and standard deviation of proposed methods trained with $K = 3$ and test with different K's

averaged over 1000 episodes, and we report mean and standard deviation over 5 independent runs. The findings are consistent with those for askscience. Since different subreddits may have very different karma scores distributions and language style, this suggests the algorithms apply to different text genres.

In actual model deployment, a possible scenario is that users may have different requests. For example, a user may ask the agent to provide $K = 2$ discussion threads on one day, due to limited reading time, and ask the agent to provide $K = 5$ discussion threads on the other day. For the baseline models (Linear, PA-DQN, DRRN), we will need to train separate models for different K's. The proposed methods (DRRN-Sum and DRRN-BiLSTM), on the other hand, can easily handle a varying K. To test whether the performance indeed generalizes well,

we train proposed models on askscience with $N = 10, K = 3$ and test them with $N = 10, K \in 2, 4, 5$, as shown in Table 5. Compared to the proposed models that are specifically trained for these K's (Table 4), the generalized test performance indeed degrades, as expected. However, in many cases, our proposed methods still outperform all three baselines (Linear, PA-DQN and DRRN) that are trained specifically for these K's. This shows that the proposed methods can generalize to varying K's even if it is trained on a particular value of K.

In Table 6, we show an anecdotal example with state and sub-actions. The two sub-actions are strongly correlated and have redundant information. By combining the second sub-action compared to choosing just the first sub-action alone, DRRN-Sum and DRRN-BiLSTM predict 86% and 26% relative increase in action-value, respectively. Since these two sub-actions are highly redundant, we hypothesize DRRN-BiLSTM is better than DRRN-Sum at capturing interdependency between sub-actions.

6 Conclusion

In this paper we introduce a new reinforcement learning task associated with predicting and tracking popular threads on Reddit. The states and actions

State text (partially shown)
Are there any cosmological phenomena that we strongly suspect will occur, but the universe just isn't old enough for them to have happened yet?
Comments (sub-actions) (partially shown)
[1] White dwarf stars will eventually stop emitting light and become black dwarfs. [2] Yes, there are quite a few, such as: White dwarfs will cool down to black dwarfs.

Table 6: An example state and its sub-actions

are all described by natural language so the task is useful for language studies. We then develop novel deep Q-learning architectures to better model the state-action value function with a combinatorial action space. The proposed DRRN-BiLSTM method not only performs better across different experimental configurations and domains, but it also generalizes well for scenarios where the user can request changes in the number tracked.

This work represents a first step towards addressing the popularity prediction and tracking problem. While performance of the system beats several baselines, it still falls far short of the oracle result. Prior work has shown that timing is an important factor in predicting popularity (Lampe and Resnick, 2004; Jaech et al., 2015), and all the proposed models would benefit from incorporating this information. Another variant might consider short-term reactions to a comment, if any, in the update window. It would also be of interest to explore implementations of backtracking in the sub-action space (incurring a cost), in order to recommend comments that were not selected earlier but have become highly popular. Lastly, it will be important to study principled solutions for handling the computational complexity of the combinatorial action space.

References

J. Allan. 2012. *Topic detection and tracking: event-based information organization*, volume 12. Springer Science & Business Media.

Roja Bandari, Sitaram Asur, and Bernardo Huberman. 2012. The pulse of news in social media: forecasting popularity. In *Proc. Int. AAAI Conf. Web and Social Media (ICWSM)*.

S.R.K. Branavan, H. Chen, L. Zettlemoyer, and R. Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th IJCNLP*, pages 82–90, August.

S.R.K. Branavan, D. Silver, and R. Barzilay. 2011. Learning to win by reading manuals in a Monte-Carlo framework. In *Proc. of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 268–277. Association for Computational Linguistics.

Justin Cheng, Lada Adamic, P. Alex Dow, Jon Kleinberg, and Jure Leskovec. 2014. Can cascades be predicted? In *Proc. Int. Conf. World Wide Web (WWW)*.

L. Deng and D. Yu. 2014. Deep learning: Methods and applications. *Foundations and Trends in Signal Processing*, 7(3–4):197–387.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. 2016. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, and J. Hunt. 2016. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*.

A. Graves and J. Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610.

C. Guestrin, D. Koller, and R. Parr. 2001. Multiagent planning with factored MDPs. In *NIPS*, volume 1, pages 1523–1530.

M. Hausknecht and P. Stone. 2016. Deep reinforcement learning in parameterized action space. In *International Conference on Learning Representations*.

J. He, J. Chen, X. He, J. Gao, L. Li, L. Deng, and M. Ostendorf. 2016. Deep reinforcement learning with a natural language action space. In *Proc. Annu. Meeting Assoc. for Computational Linguistics (ACL)*.

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.*, 29(6):82–97.

Liangjie Hong, Ovidiu Dan, and Brian Davison. 2011. Predicting popular messages in Twitter. In *Proc. Int. Conf. World Wide Web (WWW)*, pages 57–58.

A. Jaech, V. Zayats, H. Fang, M. Ostendorf, and H. Hajishirzi. 2015. Talking to the crowd: What do people react to in online discussions? In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 2026–2031, September.

A. Krizhevsky, I. Sutskever, and G. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105.

Himabindu Lakkaraju, Julian McAuley, and Jure Leskovec. 2013. What's in a name? Understanding the interplay between titles, content, and communities in social media. In *Proc. Int. AAAI Conf. Web and Social Media (ICWSM)*.

C. Lampe and P. Resnick. 2004. Slash(dot) and burn: distributed moderation in a large online conversation space. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 543–550.

Y. LeCun, Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

T. P Lillicrap, J. J Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. 2016. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*.

L-J Lin. 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321.

Michal Lukasik, Trevor Cohn, and Kalina Bontcheva. 2015. Point process modelling of rumour dynamics in social media. In *Proc. Annu. Meeting Assoc. for Computational Linguistics (ACL)*.

M. Mathioudakis and N. Koudas. 2010. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A Rusu, J. Veness, M. G Bellemare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

K. Narasimhan, T. Kulkarni, and R. Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, September.

K. Narasimhan, A. Yala, and R. Barzilay. 2016. Improving information extraction by acquiring external evidence with reinforcement learning. *arXiv preprint arXiv:1603.07954*.

R. Nogueira and K. Cho. 2016. Webnav: A new large-scale task for natural language based sequential decision making. *arXiv preprint arXiv:1602.02261*.

B. Sallans and G. E Hinton. 2004. Reinforcement learning with factored states and actions. *The Journal of Machine Learning Research*, 5:1063–1088.

K. Scheffler and S. Young. 2002. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In *Proc. of the second International Conference on Human Language Technology Research*, pages 12–19.

D. Silver, A. Huang, C. J Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

S. P Singh, M. J Kearns, D. J Litman, and M. A Walker. 1999. Reinforcement learning for spoken dialogue systems. In *NIPS*, pages 956–962.

A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT 2015*.

B. Suh, L. Hong, P. Pirolli, and E. H. Chi. 2010. Want to be retweeted? Large scale analytics on factors impacting retweet in twitter network. In *Proc. IEEE Inter. Conf. on Social Computing (SocialCom)*, pages 177–184.

Chenhao Tan, Lillian Lee, and Bo Pang. 2014. The effect of wording on message propagation: Topic- and author-controlled natural experiments on Twitter. In *Proc. Annu. Meeting Assoc. for Computational Linguistics (ACL)*, pages 175–186.

Manos Tasgkias, Wouter Weerkamp, and Maarten de Rijke. 2009. Predicting the volume of comments on online news stories. In *Proc. CIKM*, pages 1765–1768.

Alexandru Tatar, Jeremie Leguay, Panayotis Antoniadis, Arnaud Limbourg, Marcelo Dias de Amorim, and Serge Fdida. 2011. Predicting the polularity of online articles based on user comments. In *Proc. Inter. Conf. on Web Intelligence, Mining and Semantics (WIMS)*, pages 67:1–67:8.

G. Tesauro. 1995. Temporal difference learning and TD-gammon. *Communications of the ACM*, 38(3):58–68.

C. JCH Watkins and P. Dayan. 1992. Q-learning. *Machine learning*, 8(3-4):279–292.

T.-H. Wen, M. Gasic, N. Mrksic, L. M Rojas-Barahona, P.-H. Su, S. Ultes, D. Vandyke, and S. Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*.

T. Weninger, X. A. Zhu, and J. Han. 2013. An exploration of discussion threads in social news sites: A case study of the reddit community. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 579–583. IEEE.

Tae Yano and Noah A. Smith. 2010. What's worthy of comment? Content and comment volume in political blogs. In *Proc. Int. AAAI Conf. Weblogs and Social Media (ICWSM)*.

Y. Yue and C. Guestrin. 2011. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*, pages 2483–2491.

1847

Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. 2015. SEISMIC: A self-exciting point process model for predicting Tweet popularity. In *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining*.