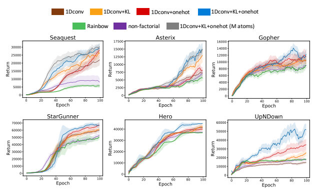
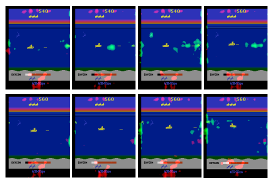
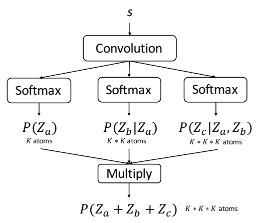


(a)



(b)



(c)

Figure 1: (a) training curves. (b) saliency maps. (c) model of full distribution (non-factorial) method.

General Response: Thanks a lot for your comments and suggestions! We will fix typos, add more details and re-organize the paper to improve clarity. Figure 1 shows the additional experiments. (**a**) *Ablative results to show the contribution of each component.* Meanwhile, we provide the results of non-factorial method in eq.3 as well as comparison with using M atoms for each distribution. (**b**) *Visualization using saliency maps.* We compute the absolute value of the Jacobian $|\nabla_x Q_i(x, \arg\max_{a'} Q(x,a'))|$ for each channel. We find that channel 1 (red region) focuses on refilling oxygen while channel 2 (green region) pays more attention to shoot sharks as well as the position where the sharks always appear. (**c**) *The model of non-factorial method.* (**d**) *Comparison with prior works.* As stated in our paper, most prior works on reward decomposition require either domain knowledge (Van Seijen et al. [2017]) or special environment (Grimm and Singh [2019] requires resettable environment). So it is impossible to fairly compare our algorithm which works on usual RL environments with theirs. Also the reported performances of those two works are quite limited: the reported performance of Grimm and Singh [2019] is only slightly better than DQN or even lower on some games, and the work of Van Seijen et al. [2017] only reports its performance on MsPacman. At the moment, it is difficult to reproduce results of Van Seijen et al. [2017] and Grimm and Singh [2019] since they need either domain knowledge or resettable environment. We will try our best to implement these prior methods and add all results in the new version.

To Reviewer 1: **Q1**: *Scale to high-D decomposition?* **A1**: In principle, our method can scale to high-D decomposition, and will work well as long as the reward in the environment can be decomposed to multiple (high-D) sources of sub-reward. In experiments we adopt only 2- or 3- decomposition because in most Atari Games the reward can be decomposed to 2 or 3 sources of sub-reward. **Q2**: *Comparison with prior work?* **A2**: Please refer to (**d**) in general response. **Q3**: *Why instantaneous reward rather than additive discounted reward in disentanglement loss?* **A3**: Note that $\mathcal{F}$ in loss 6 is the distribution of sub-returns instead of sub-rewards, so our disentanglement loss consists with that of Grimm and Singh [2019] in using additive discounted reward. The notations here might be a bit misleading; we will refine and make them clearer.

To Reviewer 2: **Q1**: *Using M atoms?* **A1**: As shown in Figure 1, the performance of using M atoms for each distribution is similar to (or slightly better than) results of using M/N atoms, which is due to larger network capacity. This suggests that the effectiveness of our method does not come from simply fitting an easier distribution with M/N atoms. **Q2**: *About state splitting.* **A2**: We do not use state splitting after using the one-hot embedding. **Q3**: *Situations when either sub-reward spikes but not the original one.* **A3**: We looked into those situations and found that the spikes are mainly due to over-estimations on Q. For example, returning large Q value when shark and submarine are on the same level instead of before the shark is going to get hit by projectiles. But this makes sense because the original sub-reward is a random variable and as stated in line 111-112, the pseudo reward is only used for visualization. **Q4**: *Failure cases with a single source of reward.* **A4**: We will run experiments on more Atari games and collect negative cases to see if there is anything in common. StarGunner in our paper is a game with a single source of reward; the experiments in Section 4 show that our method still out-performs Rainbow. We surprisingly found that in this game, one channel focuses on upper region of state while another channel focuses on lower region of state. We will include these results in the new version.

To Reviewer 3: **Q1**: *Ablative Analysis/Visualization using saliency maps/non factorial model.* **A1**: Please refer to (**a**)/(**b**)/(**c**) in general response. **Q2**: *Why took this approach?* **A2**: While learning a different policy for each component is intuitive, it is hard to combine those policies. Our approach uses a single policy and reaches even better score than Rainbow. The challenge of using a single policy is that it makes reward decomposition implicit and is hard to obtain meaningful sub-rewards; therefore, we derive a disentanglement loss to make the implicit sub-rewards as disentangled as possible. **Q3**: *Derivation of equation 4.* **A3**: The way in which we presented equation 4 might be a bit misleading; actually it is not difficult to derive the equation if we combine it with the distributional Bellman operator presented in Section 2.2. Due to space limitations, we cannot include the derivation here; we will add it into the new version. **Q4**: *Novelty of the projected Bellman loss.* **A4**: It was proposed in Bellemare et al. [2017], and implemented in the Rainbow architecture for distributional RL. In section 3.3. We combine our proposed KL loss with projected Bellman loss to form a final objective function. We will explicitly point out this in the new version. **Q5**: *Subscript i in eq. 7&8.* **A5**: The subscript in eq.7 refers to the probability of the i-th atom of the distribution; in eq.8 the full distribution (all atoms) is used to compute KL divergence, and so the subscript is omitted. **Q6**: *Hyper-parameters.* **A6**: We follow all the classic parameters of Rainbow except the hyper-parameters that induced by our method (number of channels N, atom number of each distribution and learning rate λ for KL loss). For the atom number, we choose M/N in order to keep the same network capacity as Rainbow. For λ, since KL loss is a regularization term, setting λ as a large value (e.g. 1 or 0.1) will hurt the agent performance. Therefore, we choose a relatively small value 0.001 and this learning rate is not sensitive.