Review:

1. Criticism: Although SAC++ provides a framework for temperature to automatically adjust, a target entropy still needs to be set. While the negative of the action space dimension is recommended, this may not be optimal and could still need some tuning.
2. Although SAC successfully automates the tuning of the temperature hyperparameter, this framework can still be sensitive to reward scaling. Since rewards vary greatly between tasks and even change dynamically during training, setting the reward scale appropriately remains a challenge for efficient learning. This issue can limit the generalizability of SAC across different domains without significant manual tuning of the reward scaling.

Rebuttal:

1. Although the target entropy requires adjustment for each task, the innovation here comes in the dynamic control of the exploration-exploitation tradeoff. This makes SAC more robust to new environments because the model still goes through various configurations of temperature to learn a better policy instead of staying fixed at one level of exploration.
2. Although the challenge of reward scaling persists, SAC offers a compelling solution to the exploration-exploitation tradeoff by maximizing entropy. This dynamic approach makes SAC more robust than traditional reinforcement learning algorithms, as it naturally adapts to uncertainty in action selection. Furthermore, automating the temperature adjustment significantly reduces the burden of hyperparameter tuning compared to other approaches that require manual adjustment for every environment, especially in dynamic and high-dimensional spaces.

Analysis:

Problem:

The primary problem faced by traditional RL methods like Q-learning and policy gradients is a difficulty in balancing exploration and exploitation, causing instability in performance during training. This leads to an overreliance on hyperparameter tuning. Additionally, these models face issues with a high sample complexity.

Motivation:

The motivation behind SAC is necessitated by the need to reduce instability in training, especially by being robust to various configurations of hyperparameters. Additionally, many real-world RL problems do not have the luxury of having large datasets of environment interactions or the ability to simulate/collect more data from the environment, so the research aims to reduce the sample complexity.

Although the original SAC paper performs well on benchmarks, the temperature hyperparameter (alpha), controlling the trade-off between exploration and exploitation, needs to be independently tuned for each task. This makes the original SAC algorithm less scalable and robust. The SAC++ paper introduces an automatic temperature tuning mechanism, making SAC more generalizable across environments.

Method:

SAC has a few key aspects in its methodology:

- A maximum entropy framework in the context of policy iteration, then applied as a Soft Actor Critic with function approximators for the Q-function and the policy. The model uses a specific

temperature parameter (alpha) to adjust the trade-off between reward and entropy. This is specifically the main novel contribution from the paper.
- Off-policy learning implemented via a replay buffer. While learning, the model randomly samples from the replay buffer.
- A stochastic policy, often represented via a Gaussian distribution. This complements the use of an off-policy replay buffer; if the model learns from a lot of older data, the stochasticity prevents the model from incorrectly converging to suboptimal actions.
- A twin Q-function that takes the minimum of each Q-value while training. Although not necessarily novel, this also complements the use of a replay buffer as it mitigates inaccuracies from older data in the buffer by choosing the more conservative estimate.

SAC++ implements the temperature tuning mechanism via a dual optimization problem. The update rule is gradient-descent based with the following equation:

$\nabla_{\alpha} J(\alpha) = E_{(s,a)\sim\pi}[-log\pi(a|s) - H]$

alpha will increase if the policy is becoming too deterministic (entropy is below the target), and alpha will decrease if the policy is too stochastic.

Experiments:

The SAC experiments evaluate the efficiency and stability across the OpenAI Gym benchmark tasks like Hopper, Walker2d, HalfCheetah, Ant, and Humanoid. SAC's performance on these tasks are benchmarked against DDPG, PPO, SQL, and TD3. The graphs provide insights into each model's convergence speed, final return, and overall stability. The authors draw a conclusion that SAC is close to baseline returns from other methods on easier tasks like Hopper, but tends to significantly outperform baseline methods on harder tasks like Humanoid and Ant.

SAC also performs ablation studies regarding stochasticity, reward scale, and the target smoothing coefficient. The authors conclude that across multiple seeds, the deterministic version of the policy had a much higher variance than the stochastic policy, indicating worse stability. For reward scaling, the authors argue that reward scaling is very important in determining the tradeoff between exploration and exploitation, and find that high reward scales lead to faster learning, but eventually become nearly deterministic. Thus, the authors conclude a "middle" reward scaling should be chosen for faster learning and better performance. Lastly, the authors use an exponential moving average with a smoothing constant to update the value network. They argue that large constant values lead to instability, but they find that there's a large range between 0.00001-0.01 as the constant that leads to stable performance.

In the SAC++ paper, the fixed vs learned temperature SAC methods are benchmarked across various OpenAI gym benchmarks. The authors conclude that SAC with a learned temperature performs much better on complex tasks like Humanoid-v2 while having similar performance for simpler tasks. The paper also shows an experiment where a Minitaur robot was trained with SAC to walk on flat terrain, but was able to generalize to unseen situations as well. Lastly, the authors also provide an example of an SAC "declaw" hand agent performing complex dexterous hand manipulation tasks, converging to a policy much faster than PPO.