**Summary**

**Contributions**
- Standardizing and benchmarking differential evolution (DE) as a search strategy for neural architecture search (NAS).
- We demonstrate that our DE yields state-of-the-art performance for NAS, comparing favorably to regularized evolution (RE) and Bayesian optimization.

**Observations**
- DE yields improved and more robust results for 13 tabular NAS benchmarks based on NAS-Bench-101, NAS-Bench1Shot1, NAS-Bench-201 and NAS-HPO bench.
- DE shows strong final performance, compared to RE, BOHB.
- DE appears to be robust to high-dimensional spaces and handle mixed data types adeptly.

**Canonical DE**

DE is an evolutionary algorithm that is based on four steps:

- **Initialization:** Initialize a population space of \( NP \) individuals \( \{x_{1}, x_{2},..., x_{NP}\} \) where the integer parameters are additionally rounded.
- **Mutation:** A new child/offspring is produced
- **Crossover:** Combine target and mutant to generate a trial
- **Selection:** Evaluates trial and compares to keep or discard

**Parameter space mapping**

- Integer and float parameters: \( x \in [a, b] \) are retrieved as: \( a + (b - a) \cdot U \), where the integer parameters are additionally rounded.
- Ordinal and categorical parameters \( x \in \{x_1, ..., x_n\} \): the range \([0, 1]\) is divided uniformly into \( n \) bins.

**Search space visualization**

Multi-dimensional scaling (MDS) plots show the correspondence of the search trajectories between the DE space of \([0, 1]\) and the original NAS parameter space.

**DE for NAS**

- The figure below shows the general framework of our DE implementation for NAS.
- We scale all NAS parameters to \([0, 1]\) to let DE work on individuals from a uniform, continuous space.
- We found the best way of applying DE when parameters are discrete or categorical is to keep the population in a continuous space, perform canonical DE, and only discretize copies of individuals to evaluate them.
- The mutation strategy selected was rand\(_1\) and binomial crossover were selected as the DE strategies for this work.

**Experiments**

**Benchmark Results**

For NAS-101, DE is able to exploit high-dimensional spaces well and handle mixed-types better.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOHB</td>
<td>0.8363</td>
<td>0.8167</td>
</tr>
<tr>
<td>RE</td>
<td>0.8376</td>
<td>0.8178</td>
</tr>
<tr>
<td>DE</td>
<td>0.8385</td>
<td>0.8187</td>
</tr>
</tbody>
</table>

For NAS-1Shot1: For search space 3, the most complex, largest space, DE performs best and converges fastest.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Search space 1</th>
<th>Search space 2</th>
<th>Search space 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOHB</td>
<td>0.6058</td>
<td>0.6058</td>
<td>0.6072</td>
</tr>
<tr>
<td>RE</td>
<td>0.6099</td>
<td>0.6099</td>
<td>0.6099</td>
</tr>
<tr>
<td>DE</td>
<td>0.6099</td>
<td>0.6099</td>
<td>0.6099</td>
</tr>
</tbody>
</table>

**Robustness**

- For NAS-Bench-101, DE is robust in solving CIFAR and CIFAR100 while RE is better in solving CIFAR100.
- For NAS-Bench-1Shot1, DE is more robust to solve the three search spaces while we can say that RE is competitive in search space 2.
- For NAS-201, RE is more robust than DE in ImageNet while DE is competitively robust to RE in CIFAR10 and CIFAR100.
- For NAS-HPO, DE shows more robust performance in Slice and Parkinson datasets. For Protein and Naval datasets, DE is competitively robust to RE.

**Implementation Publicly Available:** [https://github.com/automl/DE-NAS](https://github.com/automl/DE-NAS)