Quickstart

To launch a **fault-tolerant** job, run the following on all nodes.

```
torchrun
   --nnodes=NUM_NODES
   --nproc_per_node=TRAINERS_PER_NODE
   --max_restarts=NUM_ALLOWED_FAILURES
   --rdzv_id=JOB_ID
   --rdzv_backend=c10d
   --rdzv_endpoint=HOST_NODE_ADDR
   YOUR_TRAINING_SCRIPT.py (--arg1 ... train script args...)
```

To launch an **elastic** job, run the following on at least `MIN_SIZE` nodes and at most `MAX_SIZE` nodes.

```
torchrun
    --nnodes=MIN_SIZE:MAX_SIZE
    --nproc_per_node=TRAINERS_PER_NODE
    --max_restarts=NUM_ALLOWED_FAILURES_OR_MEMBERSHIP_CHANGES
    --rdzv_id=JOB_ID
    --rdzv_backend=c10d
    --rdzv_endpoint=HOST_NODE_ADDR
    YOUR_TRAINING_SCRIPT.py (--arg1 ... train script args...)
```

> **Note**
>
> TorchElastic models failures as membership changes. When a node fails, this is treated as a "scale down" event. When the failed node is replaced by the scheduler, it is a "scale up" event. Hence for both fault tolerant and elastic jobs, `--max_restarts` is used to control the total number of restarts before giving up, regardless of whether the restart was caused due to a failure or a scaling event.

`HOST_NODE_ADDR`, in form <host>[:<port>] (e.g. node1.example.com:29400), specifies the node and the port on which the C10d rendezvous backend should be instantiated and hosted. It can be any node in your training cluster, but ideally you should pick a node that has a high bandwidth.

> **Note**
>
> If no port number is specified `HOST_NODE_ADDR` defaults to 29400.

> **Note**
>
> The `--standalone` option can be passed to launch a single node job with a sidecar rendezvous backend. You donâ€™t have to pass `--rdzv_id`, `--rdzv_endpoint`, and `--rdzv_backend` when the `--standalone` option is used.

> **Note**
>
> Learn more about writing your distributed training script here.

If `torchrun` does not meet your requirements you may use our APIs directly for more powerful customization. Start by taking a look at the elastic agent API.