# Running AI Models

One aspect of using AI algorithms in research is to use already designed and trained machine learning models. For those new to the world of AI, we'll provide step-by-step guidance on how to get started with pre-trained models, including running already installed application available on the Platform as well as setting up your local environment to run your first AI-powered predictions. This guide will help you to set up your environment on an HPC cluster to run AI models on large data sets.


## On Episodes Platform

The EpisodesPlatform.eu provides easy access to run AI models on data selected by the user. The models can be run as a regular application in My Workspace. For details see the following documentation:

- P and S Waves Detection Tool user guide

## On user local machines or HPC clusters

The AI tools are available as an ordinary Python packages. We recommend for installation to use **Anaconda** or **Miniforge**, but it should be possible to use any other Python installations or distributions.

### Miniforge installation

Miniforge is a Python distribution based on Conda. This is a preferred way to run scripts and notebooks distributed on our Platform. It is possible to install and run on other python distributions, but we provide support only for Miniforge/Anaconda.

The installation starts with downloading the Miniforge binary from the official project site for your platform. Then follow the instructions in the official guide.

We prepared a Conda environment with all the AI tools installed. To create the environment please:

1. Download the `epos-ai-tools.yml` from our repository.
2. If necessary activate the Miniforge environment.
3. Run the installation

```
mamba env create -f epos-ai-tools.yml
```

4. Then to activate the environment it is necessary to run for each new shell session:

```
conda activate epos-ai-tools
```

To check if you have enabled the environment correctly you should see the name of the environment in the shell prompt.

### Available Tools

The installation comes with the following applications:

- **gpd_tool** - an application for automatic detection of the first arrival time of the P and S waves based on Generalized Seismic Phase Detection with Deep Learning by Ross, Z. E., Meier, M.-A., Hauksson, E., and T. H. Heaton (2018)
- **phasenet_tool** - an application for automatic detection of the first arrival time of the P and S waves based on PhaseNet: a deep-neural-network-based seismic arrival-time picking method by Zhu, W., & Beroza, G. C. (2019)

and official packages such as:

- Seisbench - an open-source python toolbox for machine learning in seismology,
- ObsPy - an open-source project dedicated to provide a Python framework for processing seismological data,
- PyTorch - an optimized tensor library for deep learning using GPUs and CPUs

### Tools for picking P and S waves

To run application which are responsible for picking P and S waves please use one of our applications: **gpd_tool** and **phasenet_tool**. To see all possible options please use option `--help`. Both applications have pick command which is used to detect phases in streams of data. To use one of the application provide the names of stream files as arguments. This command is designed for detecting phases in these specified streams using the specified configuration options e.g.:

```
phasenet_tool pick --threshold-p 0.4 --threshold-p 0.3 input_stream.mseed
```

The input for this command can be any data streams that can be read by ObsPy. The AI models used for phase detection require that each station's stream contains a minimum of three channels, and the signal must have a duration longer than 30 seconds.

The application generates and saves the probabilities of P, S, or Noise occurrences at each data point in MSEED format. Additionally, it stores the list of detected events in a QuakeML file, and in a custom JSON format.

Back to top

# Related Documents

- EPOS AI Workbench