Artificial Intelligence in Geosciences 4 (2023) 84–94

Contents lists available at ScienceDirect

Artificial Intelligence in Geosciences

journal homepage: www.keaipublishing.com/en/journals/artificial-intelligence-in-geosciences

Original research articles

Blockly earthquake transformer: A deep learning platform for custom phase picking

Hao Mai [a,*], Pascal Audet [a], H.K. Claire Perry [b], S. Mostafa Mousavi [c], Quan Zhang [a]

[a] Department of Earth and Environmental Sciences, University of Ottawa, Ottawa, Canada, K1N 6N5
[b] Canadian Hazards Information Service, Natural Resources Canada, Ottawa, Canada, K1A 0E7
[c] Department of Geophysics, Stanford University, Stanford, 94305, USA

ARTICLE INFO

Keywords:
Earthquake detection
Seismic phase identification
Deep learning
Seismology

ABSTRACT

Deep-learning (DL) algorithms are increasingly used for routine seismic data processing tasks, including seismic event detection and phase arrival picking. Despite many examples of the remarkable performance of existing (i.e., pre-trained) deep-learning detector/picker models, there are still some cases where the direct applications of such models do not generalize well. In such cases, substantial effort is required to improve the performance by either developing a new model or fine-tuning an existing one. To address this challenge, we present Blockly Earthquake Transformer(BET), a deep-learning platform for efficient customization of deep-learning phase pickers. BET implements Earthquake Transformer as its baseline model, and offers transfer learning and fine-tuning extensions. BET provides an interactive dashboard to customize a model based on a particular dataset. Once the parameters are specified, BET executes the corresponding phase-picking task without direct user interaction with the base code. Within the transfer-learning module, BET extends the application of a deep-learning P and S phase picker to more specific phases (e.g., Pn, Pg, Sn and Sg phases). In the fine-tuning module, the model performance is enhanced by customizing the model architecture. This no-code platform is designed to quickly deploy reusable workflows, build customized models, visualize training processes, and produce publishable figures in a lightweight, interactive, and open-source Python toolbox.

1. Introduction

In the last five years, the global seismological community has witnessed the emergence of deep-learning (DL) as a promising candidate for undertaking large-scale seismic processing and modeling tasks in modern seismology (Mousavi and Beroza, 2022a). Among many other seismological applications, DL algorithms for earthquake signal detection and seismic phase picking have had huge success, thanks to the existence of large-scale and publicly-available archived waveforms and phase labels (Mousavi and Beroza, 2022b). DL models are designed to learn intricate patterns concealed in seismic waveforms, identify seismic signals and their relations with desired targets such as seismic wave arrival times (i.e., P-arrival, S-arrival, etc.) without human intervention. One obvious application of DL phase pickers is in the automation of seismic data processing in near real-time earthquake monitoring workflows that are routinely operated by regional, national and international network operators to improve the accuracy and completeness of earthquake and seismic event catalogues (Yeck et al., 2021). In addition to improving magnitude-frequency distributions, which are critical for seismic hazard assessment and forecasting (Beroza et al., 2021), DL phase pickers may have utility in verification seismology, specifically in the identification of P- and S-phases of very low-yield (e.g., $10^{-4} - 10^{-1}$ kT), low-magnitude (e.g., $m_b 1 - 3$) nuclear explosions. Such events are best recorded at local distances (<150–200 km) by regional seismic network operators (Koper, 2019). Algorithms based on DL phase pickers like the ones presented here can expedite the creation of reliable earthquake catalogs. Recent reviews of the state-of-the-art applications show that DL-based models outperform classical characteristic function-based models in many scenarios, such as fast detection of P and S arrivals in a global benchmark dataset, and for finding S-waves that are obscured by the coda of earlier phases (Mousavi et al., 2019a; Woollam et al., 2022; Münchmeyer et al., 2022; Ma et al., 2020, 2023).

Although DL applications have shown promising performance and resulted in interesting outcomes, the path to developing more effective DL models is not always clear and efforts have been uneven. The model development procedure is often fraught with difficulties with the consequence that any new DL project requires the investment of massive amounts of labor and research resources. DL-based approaches for seismic data processing face similar issues, but most of the steps

* Corresponding author.
E-mail address: hmai090@uottawa.ca (H. Mai).

https://doi.org/10.1016/j.aiig.2023.05.003
Received 14 March 2023; Received in revised form 12 May 2023; Accepted 26 May 2023
Available online 30 May 2023

have unique prerequisites that cannot be migrated from previous work. Standard DL workflows can be summarized in four steps:

1. Pre-processing: This step prepares the training and validation data, and includes data collection, cleaning, and labeling. Benchmark seismic datasets such as STEAD (Mousavi et al., 2019a), INSTANCE (Michelini et al., 2021), and DiTing (Zhao et al., 2022) provide high-quality labeled data for training and model building. However, the number of these benchmark datasets and their applications is limited. Alternatively, applying a custom seismic sample toolbox such as QuakeLabeler (Mai and Audet, 2022) can generate labeled datasets for training based on users' demands. The quality and distribution of the training data ingested by the DL algorithm have a large impact on the performance of the resulting model. Good training data should have qualities such as relevance, minimal labeling errors, wide distribution representing various real-world classes, few missing or repeated values, etc.
2. Training: In the training step, the pre-processed seismic data are passed to a specific DL architecture to find patterns within the seismic signals and learn to pick seismic phase arrival times. Many DL models have been proposed recently, including U-Net models such as BasicPhaseAE and PhaseNet (Woollam et al., 2019; Zhu and Beroza, 2019). Convolutional and Recurrent Neural Network (CNN and RNN, respectively) earthquake event detectors include CRED and DeepPhasePick (Mousavi et al., 2019; Soto and Schurr, 2021), as well as more complex architectures like Earthquake Transformer (EqT) that adds self-attention layers (Mousavi et al., 2020).
3. Validation: Before building a final phase picker model, the validation process allows testing the model against data that have not been used for training. In general, 20% of the entire dataset are retained and excluded from the training dataset. The performance of the model on this validation set represents how likely the model is to succeed in correctly predicting phases in the face of independent seismic data. A trained model could either be approved (i.e., deemed satisfactory), tuned, or rejected based on the validation results (Livieris et al., 2020). The validation stage should not be time-intensive; however, if the model is rejected, the project returns to the training stage to revise the architecture and parameters. This training and validation loop continues until a model is approved.
4. Deployment: The approved DL picker moves on to the model development stage and is then applied to different target regions/datasets, or at various scales ranging from local and regional scales to global scales, depending on the prescribed architecture. It is common that a performing DL model fails to yield satisfactory results using a new benchmark dataset or a new target region (e.g., Woollam et al., 2022), meaning the model has weak generalization. This may be due to different data distributions between the training dataset and the data collected from the regions of interest; DL models cannot guarantee performance if the new dataset is not statistically similar to the training dataset. Such generalization issues almost always arise, and there is always a limit for the performance of a DL model (Münchmeyer et al., 2022).

To shorten the development cycle of DL phase pickers, transfer learning (TL) has been shown to be an effective solution for efficient reproducibility. TL builds new DL picker models from pre-trained models using partially or fully pre-trained hyper-parameters (weights) and network architectures to reduce the training time. This is a rapidly growing field in DL research, which reduces the massive computation requirements and time resources, and improves upon data-deficient tasks. A few early attempts have proven the success of TL, especially benefiting research projects with too few data to train a full-scale model (Lapins et al., 2021; Zhu et al., 2022). For instance, TL was utilized in Italy to enhance the earthquake ground shaking predictions using only a limited amount of training events (a few hundreds) (Jozinović et al., 2022). Another example is the Phase-Net model that was re-trained using TL to augment the accuracy of mesoscale monitoring systems in seismic phase picking (Chai et al., 2020). Nevertheless, in seismic phase picking applications, TL only addresses specific situations, such as extending the model's application to other regions or seismic phase types (Zhu et al., 2022). On the other hand, if a need arises to re-train and deploy new models for specific data/regions, users need to consider all steps of the model-building cycle and their technical requirements, as outlined above. Although TL is an effective and flexible approach, its implementation is often not straightforward. Seismologists who are not familiar with neural networks and/or common programming languages used in machine learning (e.g. Python) often find it time-consuming and cumbersome to apply TL.

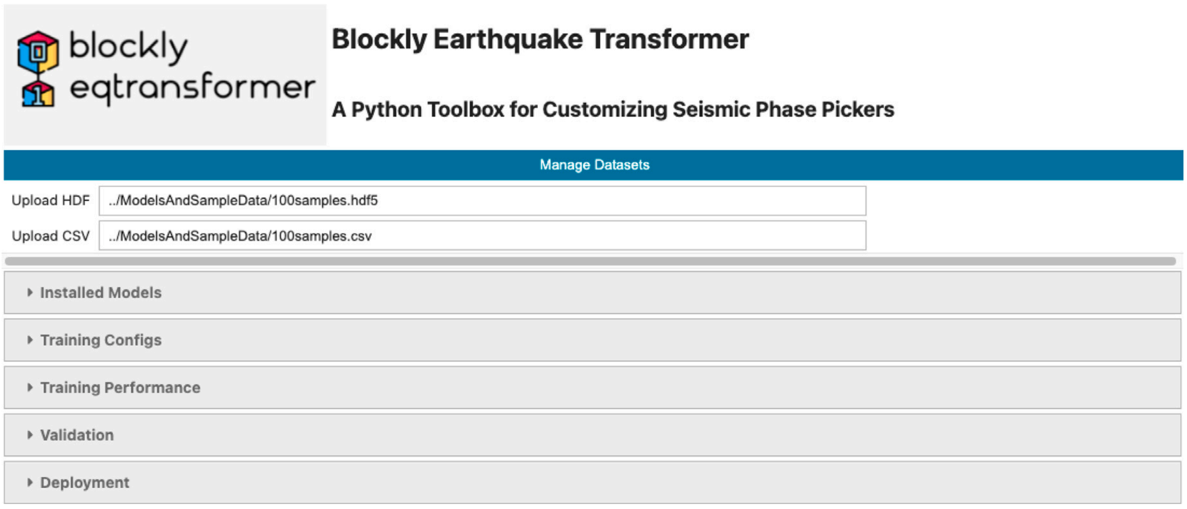
In other research fields that face similar challenges, the solution is often to use professional interactive platforms to design, train, and develop industry-level AI applications (Huang et al., 2017; Gibson et al., 2018; Ma et al., 2019). Interactive DL platforms accelerate related research, and reduce the number of repeated steps and duplicated efforts among different research groups. For instance, in natural language processing, notable projects include NVIDIA Transfer Learning Toolkit (Gopalakrishnan et al., 2017) and EasyTransfer (Wang et al., 2019). These platforms allow users to load existing models from built-in libraries, configure training settings, and perform training, validation and deployment with minimal programming operations. To the best of our knowledge, no such platform exists in the seismological community. The only python toolkit with similar functions is Seisbench, which implements previously published models that can be loaded to reproduce the built-in models and datasets (Woollam et al., 2022). However, if users want to use Seisbench to create new models or apply them to new data, code rewriting becomes an unavoidable part of the process; i.e., users need to learn the Seisbench API to write custom codes with similar PyTorch syntax as Seisbench's examples (Woollam et al., 2022).

To address these challenges and accelerate the adoption and development of DL pickers in seismology, we propose an easy-to-use, no-code, interactive platform – the Blockly Earthquake Transformer (BET) – for customizing DL pickers. BET uses Earthquake Transformer (Mousavi et al., 2020) as its base model. It provides a user-friendly interface to create new models and perform TF and fine-tuning using an interactive approach. BET's high flexibility and simplicity make it possible for users with a wide range of background knowledge and expertise to design and adjust DL models without in-depth DL knowledge. Blockly Earthquake Transformer is open source and is released under the MIT License. In the following section, we present the BET infrastructure and its possible configurations. Next, we provide three demos demonstrating BET's efficiency, effectiveness, and scalability. Finally, some potential applications of BET are discussed.

2. Methods

BET is entirely written in Python, combining Jupyter interactive widgets and Voilà to render a stand-alone web application (Silaparasetty, 2020). The graphical interface of BET is shown in Fig. 1. The BET dashboard allows users to design, analyze, and deploy industry-level phase-picking projects with little to no prior experience in Python or machine learning. For advanced users, built-in models and functions are easy to extend or rewrite through revisable APIs that can be found in the underlying Jupyter notebook.

Essentially, BET executes a tight pipeline of modules, which includes uploading datasets and pre-trained models (optional), setting up model configurations, training, validating and deploying. Additionally, built-in modules support automatic pre-processing of datasets, monitoring training performance, and saving the training history (which can be


Blockly Earthquake Transformer

A Python Toolbox for Customizing Seismic Phase Pickers

Manage Datasets

Upload HDF ../ModelsAndSampleData/100samples.hdf5

Upload CSV ../ModelsAndSampleData/100samples.csv

Fig. 1. Screen capture of the blockly earthquake transformer dashboard. BET provides a complete interactive workflow. Users can run BET's modules by defining their parameters interactively.

reloaded to continue training or to restore a previous model version) and diagnostic figures. Based on these utilities, users can either launch and apply pre-trained models to new datasets or design new DL or TL models in a short time frame.

2.1. Data management

Preparing a good quality dataset for neural network training and model building is a challenging and labor-intensive task. It is often complicated to set up a robust quality control procedure to eliminate erroneous labels and make the distribution of training data as diverse as possible. The BET data management module allows users to upload seismograms, which will be saved in a single Hierarchical Data Format version 5 (HDF5) archive file. The related information (metadata) should be supplied in a comma-separated values (CSV) file. If the file format of the uploaded dataset is compatible with STEAD (Mousavi et al., 2019a), BET can automatically parse it. However, if there are any complications, it is recommended to use QuakeLabeler (Mai and Audet, 2022) first to convert the dataset into the STEAD format. If the configuration of the uploaded data (e.g., sample length, input channel size) conflicts with the model configuration requested, BET's data management module will initialize the built-in pre-processing functions to attempt to fit the dataset into the model's prerequisites, if possible. For instance, if the length of raw seismic traces is less than the model's pre-defined channel length, the data augmentation module will be activated to adaptively duplicate a fragment of the trace away from the seismic signals to pad to the required sample length (see an example in Fig. 2); if the length of raw seismic traces is non-uniform and beyond the required input shape, a trimming method is applied to crop the trace where seismic signals exist (Fig. 2).

2.2. Model setup

Two types of model installation options are available in BET: (1) load a pre-trained model, and (2) create a new model. The first option will initialize a DL model based on the selected pre-trained network structure and hyper-parameters. The advantage of this option is that training data size can be minimal since the common seismic signal characteristics have been previously understood by the pre-trained models. Later, users can apply transfer learning, fine-tuning, and/or new model methods to customize models using a "warm start" (i.e., an optimal solution to a simpler optimization problem that sets initial values based on parameter information gained from a previous training dataset). The option to create new models is targeted to advanced users who wish to explore new architectures in addressing more complex tasks, for which current DL pickers face obstacles. In either case, users can use the BET dashboard to create model architectures without any coding involved (Fig. 3).

2.2.1. Transfer learning and fine-tuning

Transfer learning and fine-tuning options help users to optimize the performance of a pre-trained model to a new dataset (i.e., a new target region, new epicentral distance ranges, or phase types) with minimum effort. They minimize the model's revision procedures, but often produce high-quality models that result in performance enhancement. In transfer learning and fine-tuning, most of the hidden layers are set as frozen layers, which means that their hyper-parameters will not be updated during training; frozen layers will not cost further training time. Therefore, most of the computation resources are instead dedicated to the learning of new unique features representing a particular task or dataset at the very bottom hidden layer (where more high-resolution features are extracted from the data).

Note that the transfer learning and fine-tuning modes request users to select one of BET's supported pre-trained models (hyper-parameters combined) in the model setup dashboard. This is the most convenient way to start a new seismic signal detection/picker project when data and/or computation resources are limited. The default transfer learning mode is the easiest way to start a new project because it only requests users to design input/output (I/O) channels and set training strategies. If the transfer learning mode cannot capture the user's performance requirements, BET offers the fine-tuning mode to activate more frozen layers and make small and precise adjustments of the model's hyper-parameters. This results in improved models but at the cost of an increase in training time.

2.2.2. Creating new models

BET allows designing new DL models based on Earthquake Transformer(EqT)'s network architecture (Mousavi et al., 2020), which has been one of the most successful deep learning approaches, both for seismic signal detection and phase-picking tasks. In recent years, many transfer learning applications have shown that EqT is well suited for the migration to different phase-picking tasks after re-training or light-weight model revisions (Xiao et al., 2021; Zhang et al., 2022; Jiang et al., 2021). Fig. 4 shows the general EqT framework, which is used by default by BET. Users can revise the framework by specifying different arguments via this interactive form, e.g., numbers of filters, kernel size, padding function, activation method, amount of 1D convolutional neural network (CNN) blocks and bi-directional and uni-directional

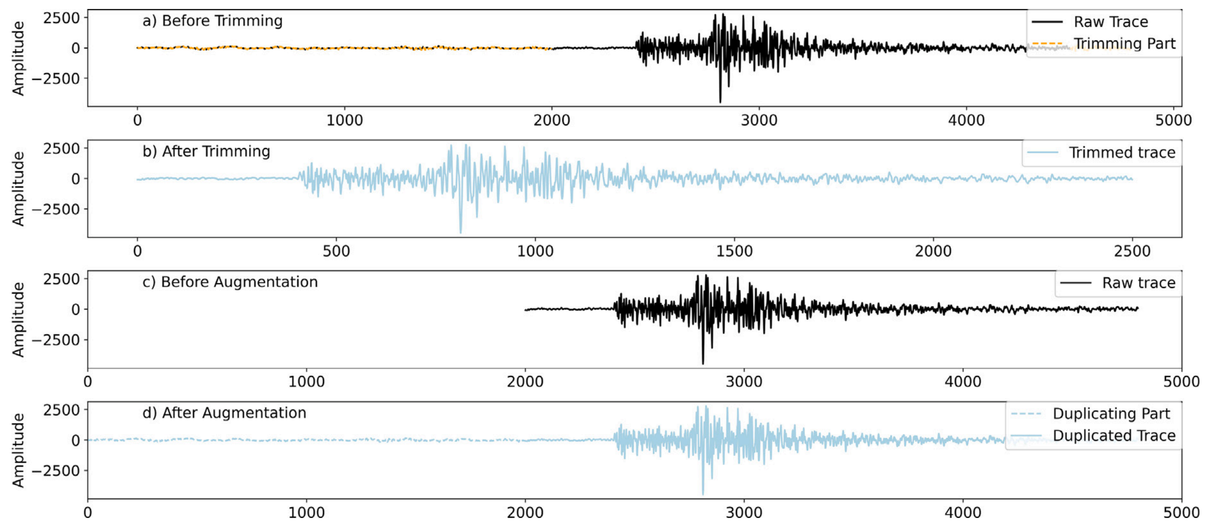


Fig. 2. Example of trimming and data augmentation. In 2a, the model's input size requires 2500 sample points, but the raw data has 4500 sample points per trace. The BET trimming module adaptively trims 2000 non-signal sample points from both ends of the raw trace. 2b is the trimmed trace which fits the input size. In 2c, another model's input size requires 4800 sample points, but the raw trace only has 2500 sample points per trace. The BET data augmentation module automatically adaptively duplicates 2300 non-signal sample points that are pre-appended to the raw trace. 2d shows the trace after data augmentation.

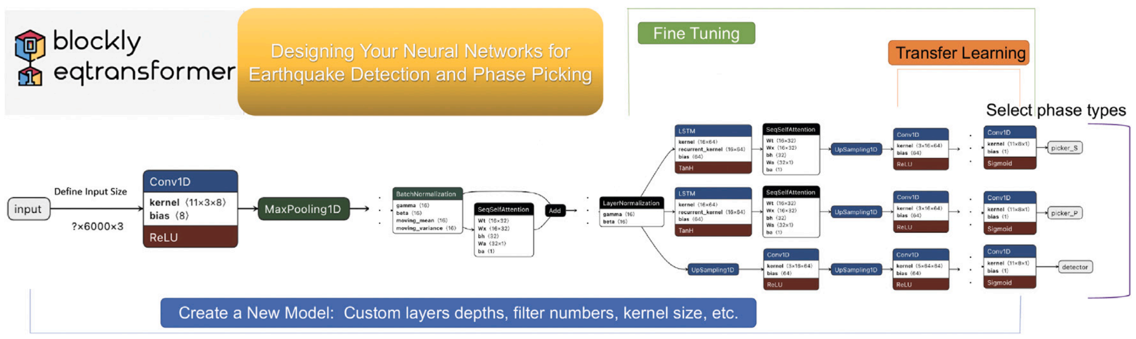


Fig. 3. Schematic diagram of BET's model setup framework. BET provides a visual interface for building and training neural networks. By default, the 'Create a New Model' module allows users to select the number of layers, the number of units in each layer, etc., to fully design a new model. 'Transfer Learning' and 'Fine Tuning' modules can load pre-trained model architectures and weights, then select to re-train part of the layers to improve the model's accuracy in a short time.

long-short-term memory blocks. These arguments control the framework of the newly created DL model, which will generate an EqT-like DL picker. This customization can be done via the BET dashboard, which means that no programming is required. Furthermore, BET also provides the complete API hidden in its Jupyter notebook, giving advanced users the possibility to rewrite or extend the other parts of the network architecture, e.g., adding new types of neural network units.

2.3. Training configuration

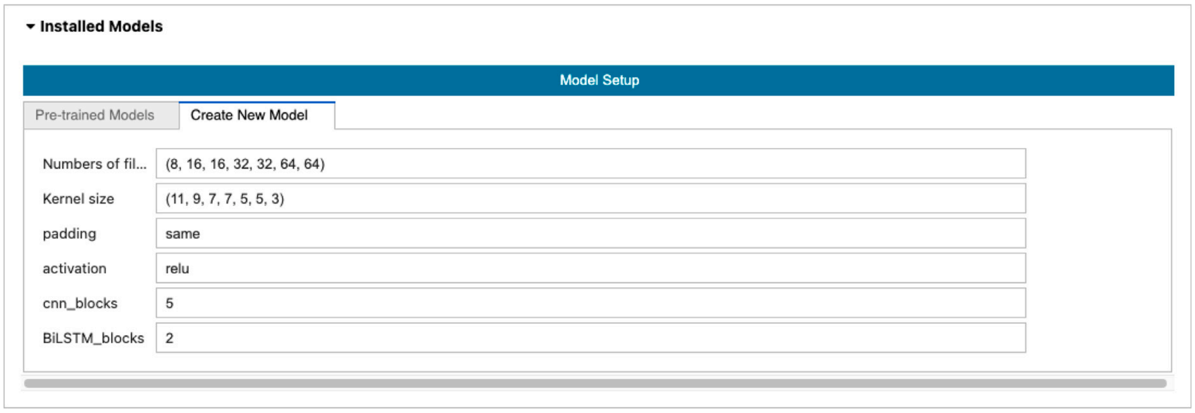
Designing a training configuration is the last step before training a DL model, and can be done interactively through the BET dashboard ("Training Configs"; see Fig. 5). Most of these arguments are straightforward, and their descriptions are shown in Table 1. The most important part of the training configuration is the designing of input/output (I/O) channels. Here we give a brief overview to help users understand the mechanism of I/O channels in a DL picker.

2.3.1. Input channel

In a DL project, prepared datasets must be compatible with I/O channels of the pre-trained model, e.g., the EqT model receives 3-component seismic traces with 6000 sample points each ($6000 * 3$). If the new datasets do not satisfy the model's input shape, errors will be raised during training unless the data have been correctly trimmed or augmented (Mousavi et al., 2020). This always hinders the quick application of pre-trained DL models to new data; fortunately, BET allows users to customize the input data shape. For instance, a single trace input (e.g., $5000 * 1$) or multiple input channels (e.g., $6000 * 5$) are allowed. Once users complete the parameter input form, BET will automatically rewrite the input data format. To offer more flexibility, BET will also check the compatibility between the training data and the model's input shape, and (if possible) automatically pre-process the training data to ensure compatibility. Various input data sampling rates are acceptable, making it possible to utilize a variety of datasets in the DL models.

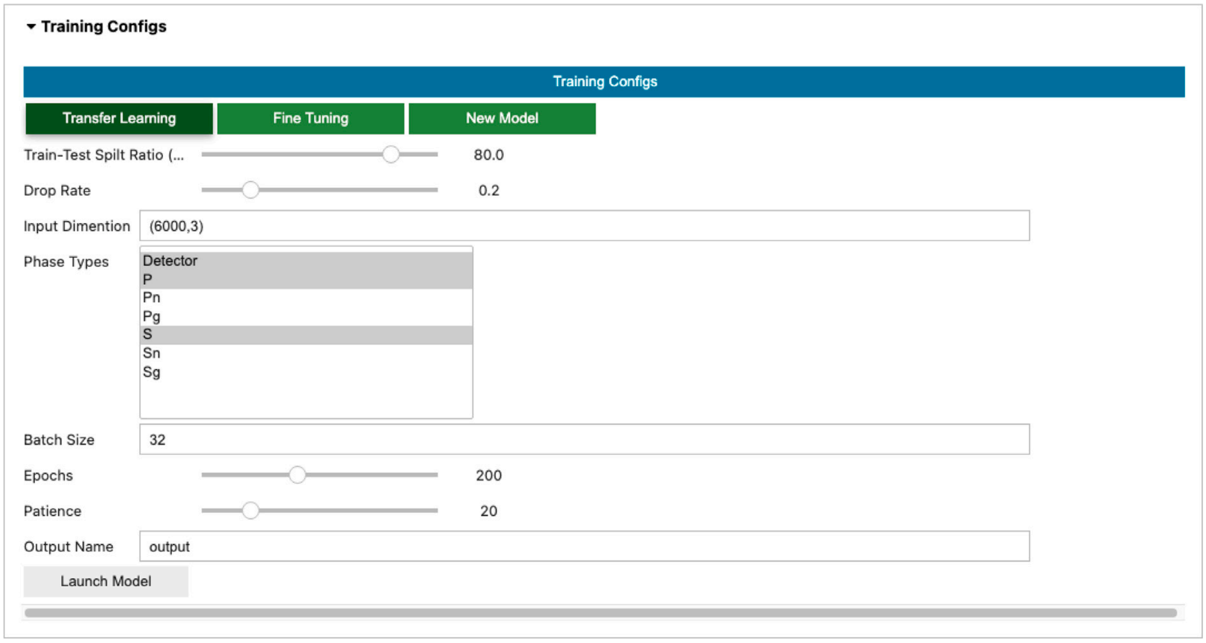
2.3.2. Output channel

The output channels represent the prediction probabilities of each identified seismic signal or phase existing at each sample point. BET defines output channels from a pre-defined list of phase types (or labels) shown in Fig. 5. Based on the supported labels, various output combinations can be formed. As a default, BET offers three major categories of output channels: seismic signal detector, P-phase picker and S-phase picker. Seismic signal detector distinguishes between seismic signals and noise. Note that, in contrast to the original EqT model output that annotates the full length of the seismic signal on the seismograms, the seismic signal detector channel in BET annotates a very limited time window (about 1–2 s), which is similar to the phase picker channels.



Numbers of fil... (8, 16, 16, 32, 32, 64, 64)
Kernel size (11, 9, 7, 7, 5, 5, 3)
padding same
activation relu
cnn_blocks 5
BiLSTM_blocks 2

Fig. 4. Screen capture of BET's model setup dashboard, with the "Create New Model" tab activated. Users can customize the model architecture via this dashboard to build new models at different scales.



Train-Test Spilt Ratio (... 80.0
Drop Rate 0.2
Input Dimention (6000,3)
Phase Types: Detector, P, Pn, Pg, S, Sn, Sg (selected: Detector, P, S)
Batch Size 32
Epochs 200
Patience 20
Output Name output

Fig. 5. Screen capture of BET's training configuration dashboard. In this dashboard, users can select different training options, i.e., TL, fine-tuning or training a new model, and other training related arguments.

Table 1
Training configuration.

Argument name	Description
Train–Test Split Ratio	Retained data used to test model performance.
Drop Rate	Dropping out rates (to avoid over-fitting).
Batch Size	Amount of training examples utilized in one iteration.
Epoch	One training iteration.
Patience	The number of epochs to wait before an early stop if there is no progress.
Output Name	Folder name of the generated DL/TL model.

Experimental results showed that, in this way, convergence of the model is faster during training. For the phase picker channels, users can select the default P and S pickers, or additional channel identifiers, i.e., P, Pg, Pn, S, Sg, Sn, etc, if those labels are available in the training metadata (i.e., catalog picks). BET will automatically rewrite the output channels based on these selections. Alternatively, users can use the default P and S labels as surrogate for other binary labels, for example, blasts and natural earthquakes, to start training a new model for the discrimination of anthropogenic and natural events.

2.4. From performance to deployment

2.4.1. Training performance

BET has a built-in visualization tool to display the training performance and monitor the loss function of all defined phase types. The loss function is a method to evaluate the difference between predicted seismic arrival times and the ground truth labels (authentic phase labels), where loss is proportional to error rate (Christoffersen and Jacobs, 2004). For example, in Fig. 6, the loss function decreases to a very low rate and remains stable, which means a successful training procedure is complete and the trained model has a very low probability of making an erroneous prediction in arrival time. We note that not all training procedures will be successful and generate ideal loss functions like the ones shown in Fig. 6. For a well-trained dataset, generally 25 training iterations (epochs) or greater are required to minimize loss for P- and S-wave pickers and detectors. It is common to encounter loss functions that remain high or fluctuate. In these cases, users need to consider redesigning the network architecture and retraining it.

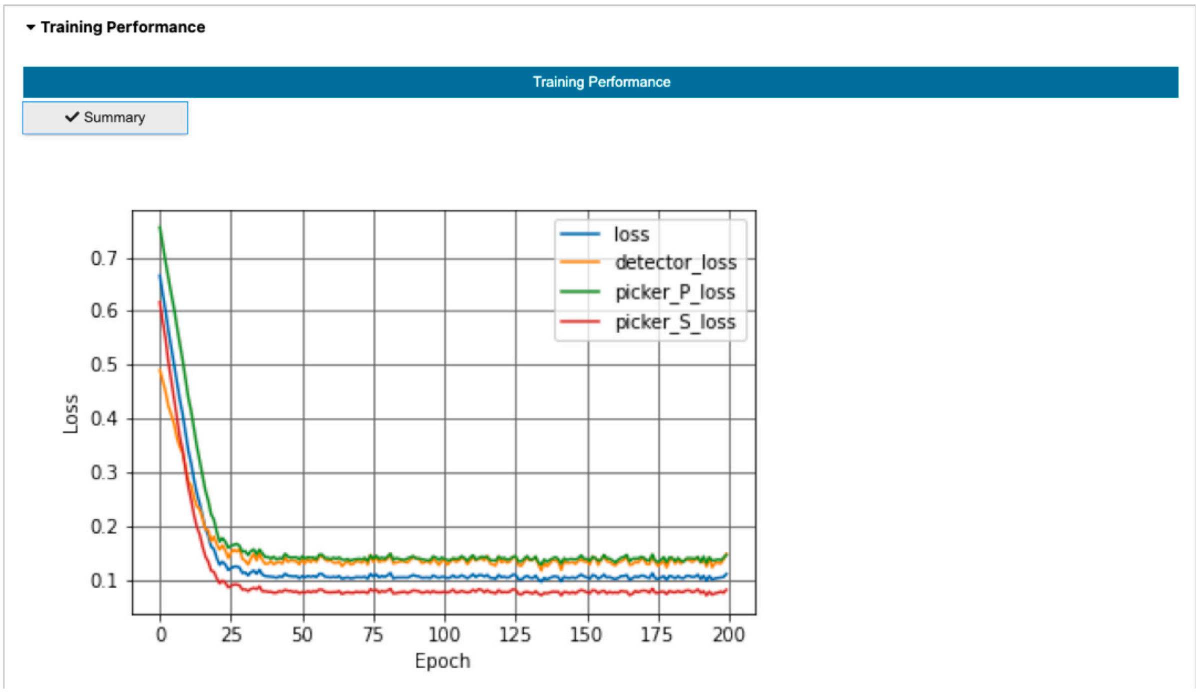


Fig. 6. Screen capture of training performance. The loss function for each output channel and the overall loss decrease linearly with epoch until they reach a stable value, where loss no longer decreases.

2.4.2. *Validation and deployment*

Validating the newly trained DL model is essential to ensure its performance and accuracy (Eelbode et al., 2021). During validation, a sufficient quantity of samples in proportion to the overall size of the dataset, but not considered in the training set, is used to validate the model's performance. These samples are preserved from the split dataset. If the validation results are as good as in the training dataset, the model has not been over-fitted and the new picker is ready to be deployed. Although there are no standards for evaluating a model, BET provides three types of supporting files to help users evaluate their trained models. In the specified output folder, users can find detailed test results to evaluate the model's performance, i.e., a summary table (*.CSV) containing all the picked results, a text file (*.TXT) reporting the configuration parameters for prediction and model performance, and a figure folder containing the picked arrivals (default: 10 examples of picks). BET also offers a deployment dashboard to quickly apply newly trained pickers to large-scale datasets (see Fig. 7). When the deployment is complete, BET exports several sample graphs and human-readable documents to help users visualize and understand the model predictions (see Fig. 13). These extensions provide ways to visualize and understand model predictions, avoid repetitive labor and let users focus on analyzing performance.

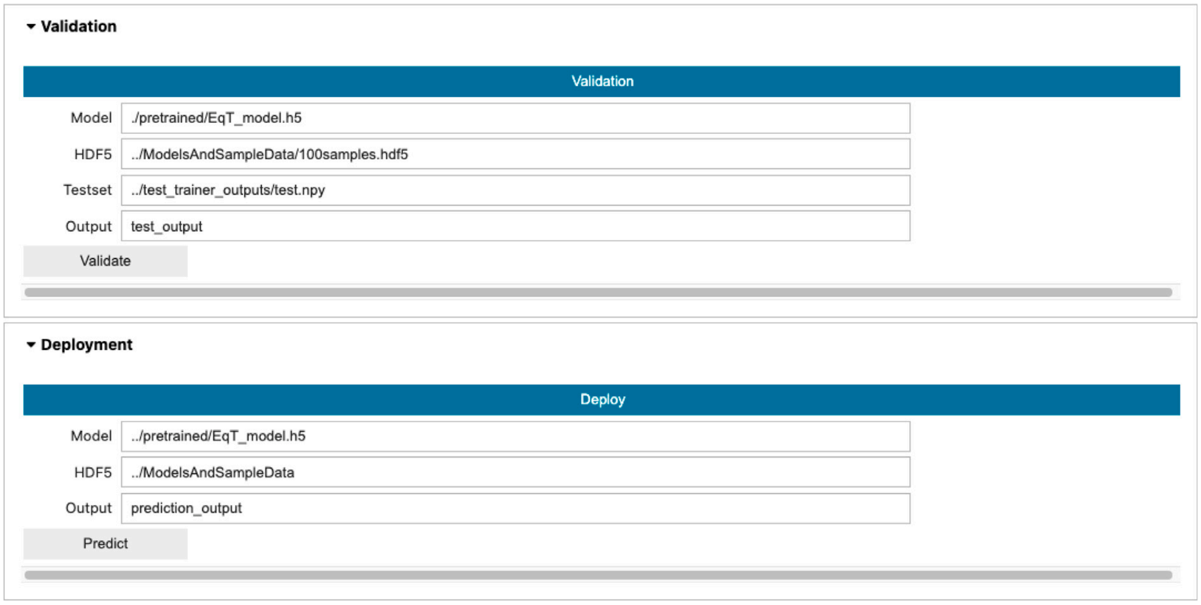
3. Prospective applications

In this section, we present three examples that cater to novice, intermediate and advanced users, respectively, to illustrate the flexibility and applicability of BET for phase-picking projects. For novice users, BET provides a convenient way to understand how deep learning models can be deployed in earthquake detection and phase-picking projects. For intermediate users, BET can be used to train new deep learning and/or transfer learning models to address specialized phase-picking tasks. For advanced users, they can fully customize deep-learning models to create novel deep-learning pickers. The focus of this section is to give users a quick overview of BET's capabilities, rather than to showcase the accuracy of any particular trained model. We will focus on various case studies in future work.

3.1. *Deploying pre-trained models for earthquake detection and phase picking*

Using a pre-trained model can be a good way to get started with deep learning, especially for novice users with little experience in DL model training. To deploy a pre-trained EqT model on a new dataset, the user may skip the training and validation steps, and start with the deployment module. BET provides built-in, pre-trained models and associated files, such as weights. Alternatively, users can download pre-trained models, obtained from other transfer learning applications using the EqT framework. The user then defines the model's file path on the BET deployment interface (see Fig. 8), and BET will load the model, the associated weights and the configuration. Other than choosing a pre-trained model, users only need to pass the new dataset through the model and use the probability output to make predictions. On the user's end, BET requires the prepared dataset directory be defined (see Fig. 8). The input dataset should be similar to the STEAD dataset (Mousavi et al., 2019a) to avoid running into errors. Building a dataset for deep learning may be challenging for novice users. To remediate this, BET provides a tutorial notebook to convert seismic data to a STEAD-like dataset. Alternatively, users can use QuakeLabeler to generate a dataset that matches the STEAD format (Mai and Audet, 2022). As described previously, BET has an auto-preprocessing module to adjust the input dataset (e.g., reshape the input size, drop data with insufficient information, etc.) to avoid errors during deployment.

Here we use a built-in dataset in STEAD format to showcase how to use the pre-trained EqT model to pick events. This dataset is the same example dataset as the original Earthquake Transformer package (Mousavi et al., 2020); it contains 99 human-reviewed local earthquake events. To try this example, users can skip to the deployment interface and press the 'Predict' button. BET will then load the pre-trained EqT model automatically (see Fig. 8). Typically, it takes 1–2 min to load the pre-trained model in the backend, depending on the available computational resources. In the second step, BET will auto-extract waveforms from the user input dataset directory's HDF5 file and send them to the model's input channel. The pre-trained EqT model predicts the probabilities of the P-phase and S-phase at each data point. In the final step, all detected events are saved in a CSV file with some example visuals stored in a separate folder.



▾ Validation

Validation

Model ./pretrained/EqT_model.h5
HDF5 ../ModelsAndSampleData/100samples.hdf5
Testset ../test_trainer_outputs/test.npy
Output test_output

Validate

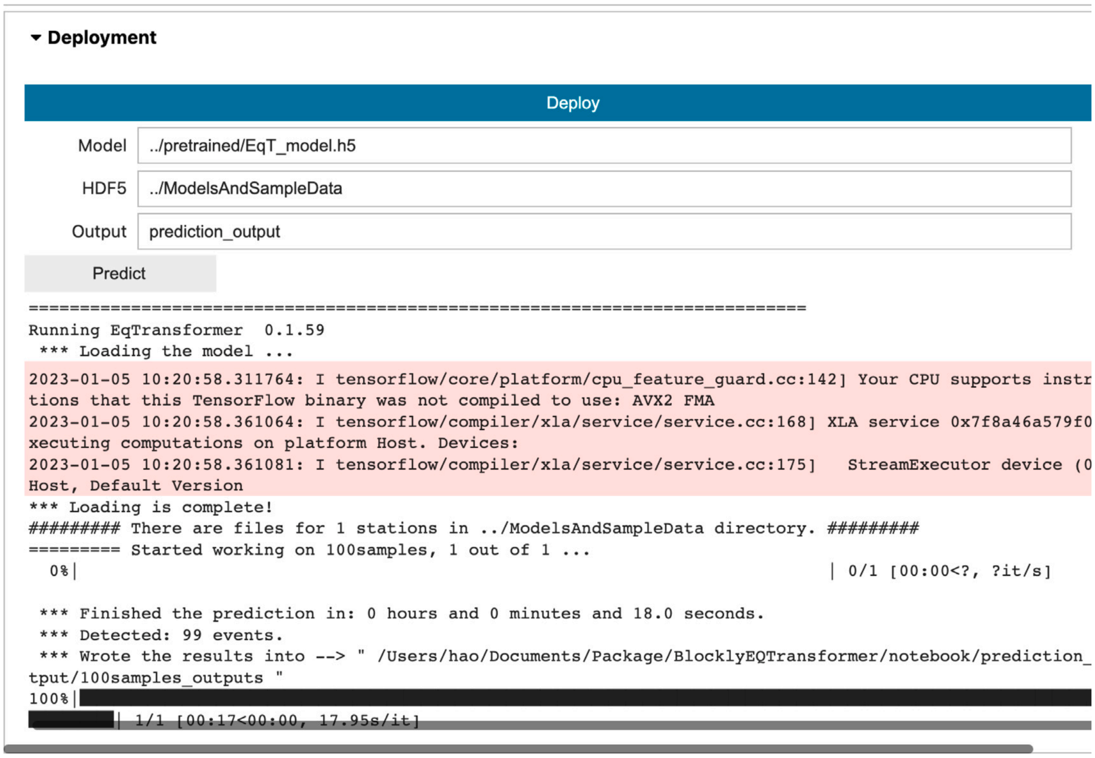
▾ Deployment

Deploy

Model ../pretrained/EqT_model.h5
HDF5 ../ModelsAndSampleData
Output prediction_output

Predict

Fig. 7. Screen capture of validation and deployment dashboard. When the user finishes a training process, the validation and/or deployment dashboard can be activated based on training results, e.g., input training output folder and other user-defined arguments.



▾ Deployment

Deploy

Model ../pretrained/EqT_model.h5
HDF5 ../ModelsAndSampleData
Output prediction_output

Predict

```
==============================================================================
Running EqTransformer  0.1.59
 *** Loading the model ...
2023-01-05 10:20:58.311764: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instr
tions that this TensorFlow binary was not compiled to use: AVX2 FMA
2023-01-05 10:20:58.361064: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x7f8a46a579f0
xecuting computations on platform Host. Devices:
2023-01-05 10:20:58.361081: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0
Host, Default Version
*** Loading is complete!
######### There are files for 1 stations in ../ModelsAndSampleData directory. #########
========= Started working on 100samples, 1 out of 1 ...
  0%|                                                                          | 0/1 [00:00<?, ?it/s]

 *** Finished the prediction in: 0 hours and 0 minutes and 18.0 seconds.
 *** Detected: 99 events.
 *** Wrote the results into --> " /Users/hao/Documents/Package/BlocklyEQTransformer/notebook/prediction_
tput/100samples_outputs "
100%|██████████████████████████████████████████████████████████████████████████████████████████████████
█████████| 1/1 [00:17<00:00, 17.95s/it]
```

Fig. 8. Screenshot of using a pre-trained model to detect and pick earthquake events. Users can use BET's built-in pre-trained models (and weights) to predict earthquake events. Following the procedure outlined in this paper, the user defines the file path and selects the 'Predict' button. BET automatically detects the earthquake events in the dataset and generates results and example visuals in the output folder. The deployment stage generally takes a few minutes, depending on the size of the dataset and computational resources.

3.2. Training models on new datasets

Deploying an AI picker in a new scenario (e.g., different input channel requirements, specific regions with no prior training data, etc.) may require the laborious adaptation of previously published DL or TL models. BET is designed to facilitate the training of new deep-learning models on user-defined datasets. The first step for training a new model is dataset preparation in the STEAD format. We illustrate the training functionality in BET by collecting training data waveforms from the Canadian National Seismograph Network, CNSN (Natural Resources Canada, 1975) and events from the National Earthquake DataBase, NEDB (Natural Resources Canada, 1985) for Cascadia Subduction Zone seismicity between 2015-2019 (see Data and Resources). For the training dataset, a subset of 726 earthquakes, including 27,219 manual picks of magnitude ≥ 2 (see Fig. 9) are extracted from the NEDB between

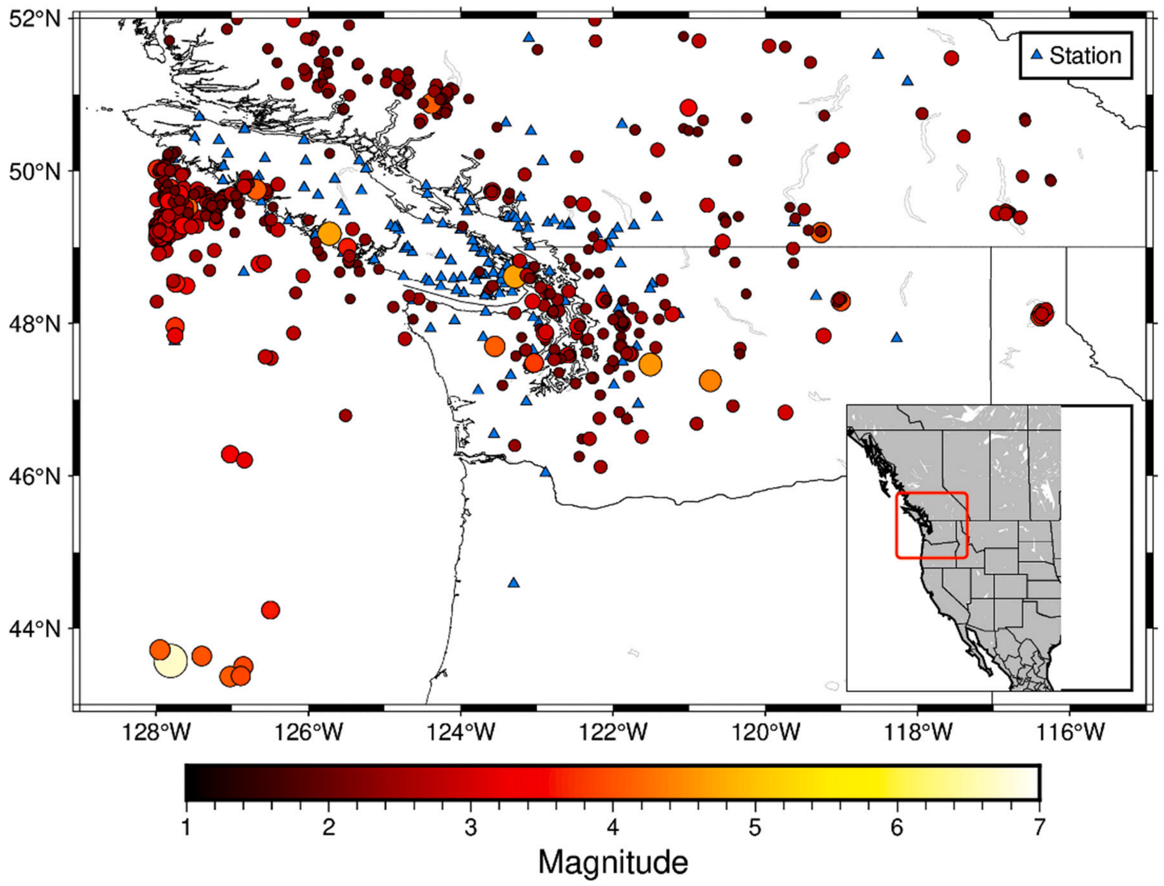


Fig. 9. Geographic data distribution of earthquakes of magnitude M ≥ 2, from the training dataset extracted from the NEDB catalog (Natural Resources Canada, 1985) from 2015-01-01 to 2019-12-31. Blue triangles represent the seismic stations. Circles represent earthquakes, with size and color scaled to event magnitudes. The NEDB dataset contains single trace P- and S-labeled, manually-picked phases and is used in this study as a training dataset to introduce BET's training module.

2015-01-01 and 2019-12-31. A magnitude threshold of 2 was chosen based on previous studies (Yeck et al., 2021). We first use this dataset to evaluate the pre-trained EqT model for this new region (Cascadia) and use transfer learning to improve the model.

The second step is to define the architecture of the model. Here we select the 'Transfer Learning' function for illustration. In the new dataset, each sample uses one channel of input trace; the vertical component for P arrivals and one horizontal component for S arrivals. This input structure differs from most published DL models, making direct application of the pre-trained EqT model challenging. However, for transfer learning on the EqT model, the user needs only to specify that the input shape is that of the original EqT model (i.e., 6000 ∗ 3) and set up other arguments as shown in Table 2. The new dataset will be automatically formatted as the EqT model, and fed into training in BET's backend. In this case, EqT pre-trained model weights are acceptable as a warm start to accelerate model convergence.

The transfer learning and fine-tuning options are suitable for users who do not have access to arrays of GPUs for computation. A single desktop-level GPU (e.g., GTX 1080I) or workstation with CPU cores (e.g., 8 cores of Intel i7) can complete a training step in a few hours. When training is complete, the next step is to test the accuracy of the trained picker in the validation dashboard. We use the validation dataset (2721 samples), which gets automatically split when loading the dataset, to test BET's prediction accuracy. Performance comparison of the original EQTransformer (V1.59) model and the BET newly-trained transfer-learning model (Table 3), demonstrates that the new picker has improved detection ability. The original EqT model's low accuracy suggests that the NEDB data distribution differs from the STEAD dataset. Fig. 10 shows a predicted result from the newly trained picker. This example illustrates a promising method for improving picking in a region where the DL model has never been applied.

Table 2
Model configuration for Cascadia picker.

Model configuration	
Input Dimension	(6000, 3)
Train–Test Spilt Ratio	0.8
Training Mode	Transfer learning
Drop Rate	0.2
Phase Types	P, S
Batch Size	100
Epochs	100
Patience	20

Table 3
Model performance on the CNSN/NEDB validation dataset.

Model	P-phase accuracy	S-phase accuracy
EQTransformer (V1.59)	73.9%	66.6%
BET New Trained Picker	92.3%	88.6%

3.3. Beyond P- and S-phase picking

Here we present a case where BET is used to train a comprehensive AI phase picker to extend the detectability of P and S phase arrivals and further include direct crustal phases (Pg and Sg) and/or Moho refracted phases (Pn and Sn). We use BET's fine-tuning mode to create the model. In this example, we select all phase types in the configuration dashboard (i.e., P, Pg, Pn, S, Sg, Sn), define the input channel as

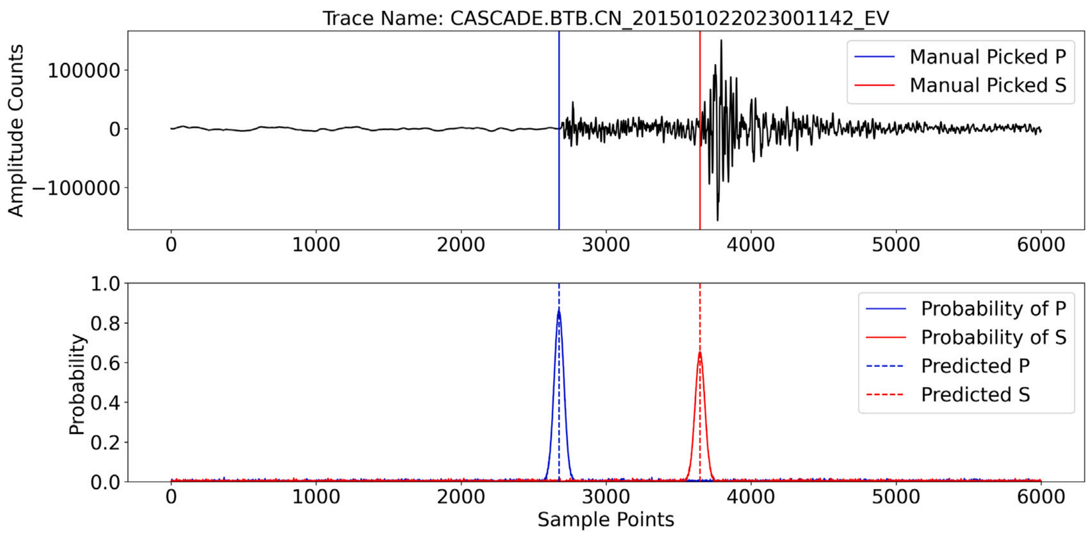


Fig. 10. Representative predicted P and S arrival in Cascadia test dataset. The trained picker correctly identifies P and S phase from a N-component seismic trace. The human-reviewed P and S arrival times are taken from the NEDB catalog at station CN.BTB for a magnitude 2.0 event on 2015-01-02 at an epicentral distance of 78 km. The picker predicts a high probability (>0.6) at the correct sample position.

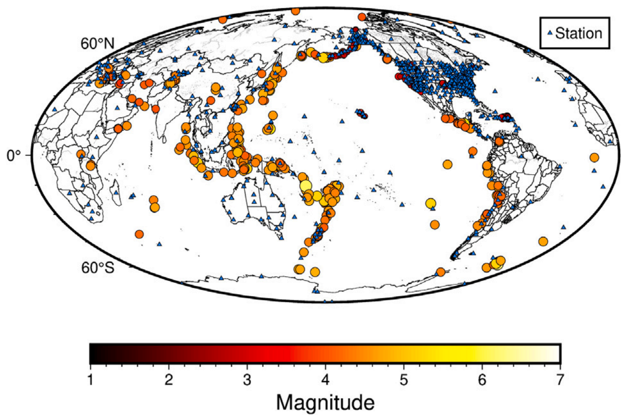


Fig. 11. Geographic data distribution of the USGS dataset (Cole and Yeck, 2022) used in this application. The blue triangles represent the available seismic stations. The circles represent earthquakes color-coded by magnitude. The USGS dataset contains human-reviewed P, Pn, Pg, S, Sn and Sg labeled samples, and is used here as a training dataset to test BET's applicability.

$6000 * 3$ and use default values for all training settings. We download three months (i.e., from June to September 2013) of data from the U.S. Geological Survey (USGS) machine learning dataset (Cole and Yeck, 2022) and convert it into STEAD format. This USGS dataset is publicly available and contains a wide range of human-reviewed samples of the above 6 phase types. In this project, we set 0.8 as the training-test-split ratio to produce 31,254 training samples and 7814 validation samples (the data distribution map is shown in Fig. 11). In the USGS dataset, each phase is centered in the trace, i.e., the original waveform is sampled at 40 Hz with a start time 60 seconds before the first P-wave arrival, so the arrival time is at the 2400 sample point. Therefore, the original ground truth labels are fixed at a static position, which will cause problems for the model in understanding the spatial features of the picking tasks. However, once the data loading function is activated, BET will automatically pre-process the dataset in its backend, augment sample length to the input shape and randomize arrival positions to avoid biased training.

In this example, training reaches its best performance after only 7 epochs, at which point the training is stopped automatically (after 2 h 20 min of training time). The pre-trained model (EqT version 1.59 model) provides the basic network structure and a warm start. In this training solution, the total number of hyper-parameters is 378,667, but there are only 4871 trainable hyper-parameters. The 373,796 non-trainable hyper-parameters are frozen layer units. This significantly reduces the training time and required data size. The training is mainly limited to fine-tuning the new TL model to capture the characteristics of new seismic phases (i.e., Pg, Pn, Sg, and Sn) by updating the learned features at the final layers of the EqT baseline model, enabling the model to distinguish between these phases. According to Table 4, the last three epochs of loss value are relatively low (below 0.0420855), indicating a successful training despite a low number of iterations and training samples. The accuracy in the validation set also indicates that this newly trained phase picker can correctly detect P, Pg, Pn, S, Sg, Sn phases in the new data in a similar way as the usual P and S phases. The output probability channel predicts relatively high values (i.e., Pn, Pg channel > 0.8; Sn, Sg channel > 0.6) at the location of seismic arrivals for the correct phase output channels (see Fig. 12), with other channels correctly predicting very low probabilities (see Fig. 13).

Table 4
Transfer learning performance in USGS dataset.

Phase type	Last value of loss function	Accuracy in validation
P	0.0032257	99.5%
Pg	0.0100786	97.8%
Pn	0.0058863	98.2%
S	0.0113646	93.6%
Sg	0.0269472	94.5%
Sn	0.0420855	92.4%

4. Extensibility

BET allows users to customize various types of DL pickers. Based on different types of datasets, BET can train models that encompass regional to teleseismic events and identify signal types such as natural earthquakes, explosions, volcanic or even vehicle tremors. In phase-picking tasks, phase types are no longer limited to only P and S phases, and can be extended to P, Pg, Pn, S, Sg, and Sn. Furthermore, advanced users can implement other phase types (i.e., to distinguish between natural earthquakes and blasts) by expanding BET's output channels at the scripting level. BET's Jupyter notebook can be used to replace the **P** and **S** types with **Natural Earthquakes** and **Blasts**, respectively. Then, by running voila (Silaparasetty, 2020) to reinitialize BET, these 2

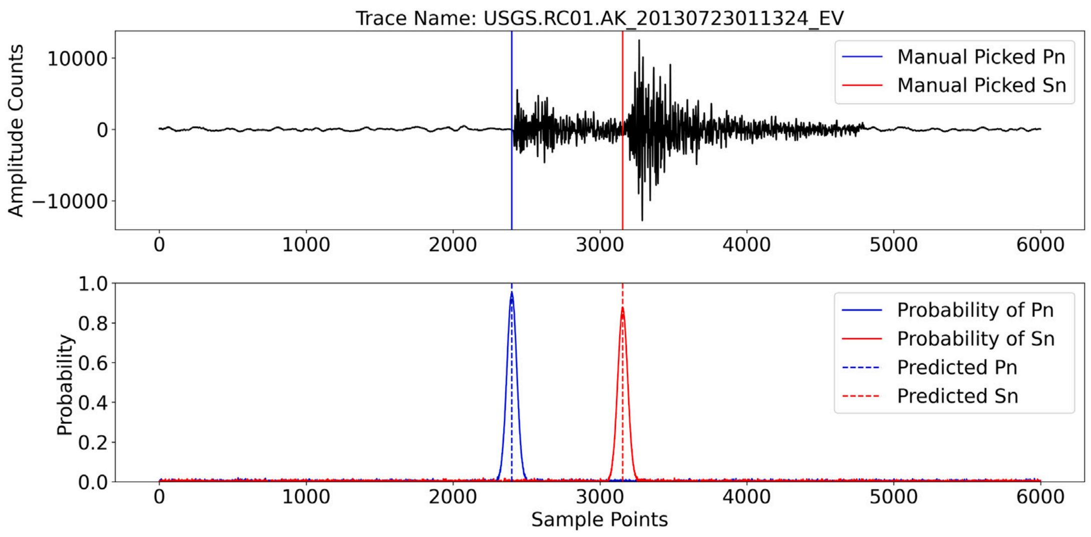


Fig. 12. Representative predicted Pn and Sn arrivals in the USGS dataset (Cole and Yeck, 2022).The fine-tuned picker successfully detects a Pn-arrival and a Sn-arrival at station AK.RC01 from the USGS validation dataset. This event is human-reviewed with a magnitude of 2.9 recorded at a distance of 161 km from the epicenter.

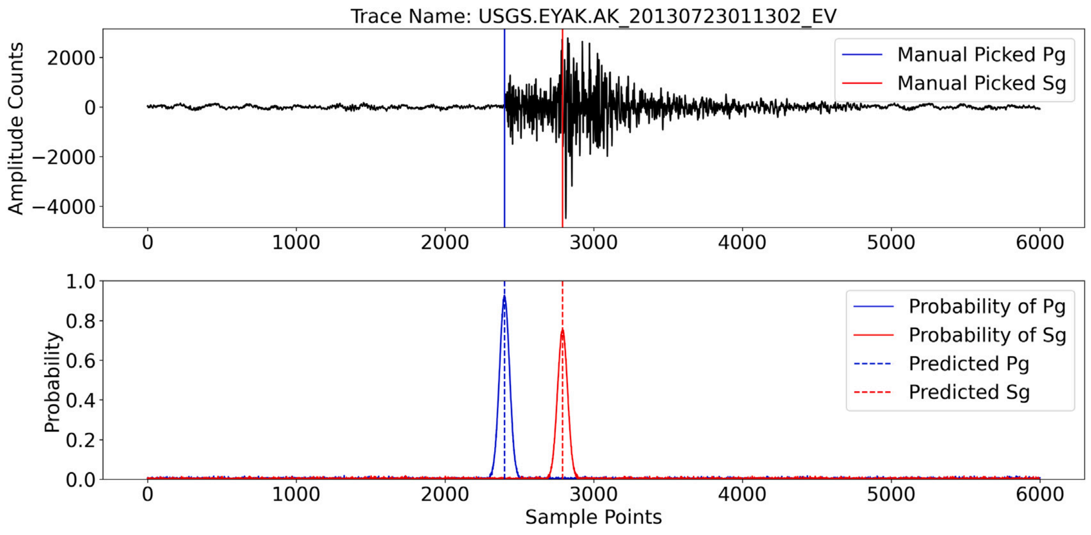


Fig. 13. Representative predicted Pg and Sg arrivals in the USGS dataset (Cole and Yeck, 2022).The custom BET picker successfully detects a Pg-arrival and a Sg-arrival at station AK.EYAK from the USGS validation dataset. This event is human-reviewed with a magnitude of 2.9 recorded at a distance of 65 km from the epicenter.

new channels can be utilized for training. More complex extensions are also allowable in BET's scripting level and outlined in the user manual.

5. Conclusion

The application and development of DL models are steadily growing in seismology. BET provides a user-friendly platform for engaging a broader range of users, including those without machine learning or coding experience. BET allows researchers to explore DL pickers without having to worry about in-house computational power and other technical concerns. BET will prove useful in improving model performance and lowering the magnitude of completeness of existing seismic catalogues. Similar packages can be developed for other seismological tasks, including those in exploration seismology, where fine-tuning pre-trained models by synthetic data is even more common due to the scarcity of large-scale labeled training datasets. Other applications of BET are in the discrimination between natural earthquakes and anthropogenic events, where separate training datasets may be developed for each event types. This package will be both practical for students to understand deep learning applications and convenient for research groups to deploy large-scale phase-picking tasks. The BET package is available as a GitHub repository at https://github.com/maihao14/BlocklyEQTransformer. We encourage users to contribute trained models to enrich the library of BET models.

6. Data and resources

The (Canadian) National Earthquake Database may be found at https://www.earthquakescanada.nrcan.gc.ca/stndon/NEDB-BNDS/bulletin-en.php (last accessed June 2023). At this time, the online database is searchable from 1985 to present and returns only the solutions. Solutions and phase information for older earthquakes may be obtained by contacting http://nrcan.earthquakeinfo-infoseisme.rncan@canada.ca. Waveform data and/or station metadata from the CNSN, may be retrieved via FDSN dataselect webservice https://earthquakescanada.nrcan.gc.ca/fdsnws/dataselect/1/, via FDSN station webservice https://earthquakescanada.nrcan.gc.ca/fdsnws/station/1/, or via Station Book https://earthquakescanada.nrcan.gc.ca/stndon/CNSN-RNSC/stnbook-cahierstn/index-en.php.

The Global Earthquake Machine Learning Dataset: Machine Learning Asset Aggregation of the PDE (MLAAPDE) from USGS may be found at https://www.sciencebase.gov/catalog/item/6127b30fd34e40dd9c05094c (last accessed December 2022). USGS dataset provides 6 types of human-reviewed phases, i.e., P, Pg, Pn, S, Sg, Sn.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was funded by a Discovery Grant (RGPIN-2018-03752) from the Natural Science and Engineering Research Council of Canada (PA). The authors thank Stephen Crane, Laurel Sinclair, two anonymous reviewers and the editor for their reviews of this manuscript. This is NRCan publication number 20220610.

References

Beroza, G.C., Segou, M., Mostafa Mousavi, S., 2021. Machine learning and earthquake forecasting—next steps. Nature Commun. 12 (1), 1–3.

Chai, C., Maceira, M., Santos-Villalobos, H.J., Venkatakrishnan, S.V., Schoenball, M., Zhu, W., Beroza, G.C., Thurber, C., Team, E.C., 2020. Using a deep neural network and transfer learning to bridge scales for seismic phase picking. Geophys. Res. Lett. 47 (16), e2020GL088651.

Christoffersen, P., Jacobs, K., 2004. The importance of the loss function in option valuation. J. Financ. Econ. 72 (2), 291–318.

Cole, H., Yeck, W.L., 2022. Global Earthquake Machine Learning Dataset: Machine Learning Asset Aggregation of the PDE (MLAAPDE). US Geological Survey, URL https://www.sciencebase.gov/catalog/item/6127b30fd34e40dd9c05094c.

Eelbode, T., Sinonquel, P., Maes, F., Bisschops, R., 2021. Pitfalls in training and validation of deep learning systems. Best Pract. Res. Clin. Gastroenterol. 52, 101712.

Gibson, E., Li, W., Sudre, C., Fidon, L., Shakir, D.I., Wang, G., Eaton-Rosen, Z., Gray, R., Doel, T., Hu, Y., et al., 2018. NiftyNet: a deep-learning platform for medical imaging. Comput. Methods Programs Biomed. 158, 113–122.

Gopalakrishnan, K., Khaitan, S.K., Choudhary, A., Agrawal, A., 2017. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. Constr. Build. Mater. 157, 322–330.

Huang, L., Dong, X., Clee, T.E., 2017. A scalable deep learning platform for identifying geologic features from seismic attributes. Lead. Edge 36 (3), 249–256.

Jiang, C., Fang, L., Fan, L., Li, B., 2021. Comparison of the earthquake detection abilities of PhaseNet and EQTransformer with the Yangbi and Maduo earthquakes. Earthq. Sci. 34 (5), 425–435.

Jozinović, D., Lomax, A., Štajduhar, I., Michelini, A., 2022. Transfer learning: Improving neural network based prediction of earthquake ground shaking for an area with insufficient training data. Geophys. J. Int. 229 (1), 704–718.

Koper, K.D., 2019. The importance of regional seismic networks in monitoring nuclear test-ban treaties. In: AGU Fall Meeting Abstracts. Vol. 2019. pp. S14B–05.

Lapins, S., Goitom, B., Kendall, J.-M., Werner, M.J., Cashman, K.V., Hammond, J.O., 2021. A little data goes a long way: Automating seismic phase arrival picking at Nabro volcano with transfer learning. J. Geophys. Res. Solid Earth 126 (7), e2021JB021910.

Livieris, I.E., Stavroyiannis, S., Pintelas, E., Pintelas, P., 2020. A novel validation framework to enhance deep learning models in time-series forecasting. Neural Comput. Appl. 32 (23), 17149–17167.

Ma, Y., Cao, S., Rector, J.W., Zhang, Z., 2020. Automated arrival-time picking using a pixel-level network. Geophysics 85 (5), V415–V423.

Ma, Y., Eaton, D., Igonin, N., Wang, C., 2023. Machine learning-assisted processing workflow for multi-fiber DAS microseismic data. Front. Earth Sci. 11.

Ma, Y., Yu, D., Wu, T., Wang, H., 2019. PaddlePaddle: An open-source deep learning platform from industrial practice. Front. Data Domput. 1 (1), 105–115.

Mai, H., Audet, P., 2022. QuakeLabeler: A fast seismic data set creation and annotation toolbox for AI applications. Seismol. Soc. Am. 93 (2A), 997–1010.

Michelini, A., Cianetti, S., Gaviano, S., Giunchi, C., Jozinovic, D., Lauciani, V., 2021. INSTANCE–the Italian seismic dataset for machine learning. Earth Syst. Sci. Data Discuss. 1–47.

Mousavi, S.M., Beroza, G.C., 2022a. Deep-learning seismology. Science 377 (6607), eabm4470.

Mousavi, S.M., Beroza, G.C., 2022b. Machine learning in earthquake seismology. Ann. Rev. Earth Planet. Sci. 51, 2023.

Mousavi, S.M., Ellsworth, W.L., Zhu, W., Chuang, L.Y., Beroza, G.C., 2020. Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. Nature Commun. 11 (1), 1–12.

Mousavi, S.M., Sheng, Y., Zhu, W., Beroza, G.C., 2019a. Stanford earthquake dataset (STEAD): A global data set of seismic signals for AI. IEEE Access 7, 179464–179476.

Mousavi, S.M., Zhu, W., Sheng, Y., Beroza, G.C., 2019. CRED: A deep residual network of convolutional and recurrent units for earthquake signal detection. Sci. Rep. 9 (1), 1–14.

Münchmeyer, J., Woollam, J., Rietbrock, A., Tilmann, F., Lange, D., Bornstein, T., Diehl, T., Giunchi, C., Haslinger, F., Jozinović, D., et al., 2022. Which picker fits my data? A quantitative evaluation of deep learning based seismic pickers. J. Geophys. Res. Solid Earth 127 (1), e2021JB023499.

Natural Resources Canada, 1975. Canadian National Seismograph Network [dataset]. Int. Fed. Digit. Seismogr. Netw. http://dx.doi.org/10.7914/SN/CN.

Natural Resources Canada, 1985. Canadian National Earthquake Database [dataset]. Can. Hazards Inf. Serv. http://dx.doi.org/10.17616/R3TD24.

Silaparasetty, N., 2020. Introduction to jupyter notebook. In: Machine Learning Concepts with Python and the Jupyter Notebook Environment. Springer, pp. 91–118.

Soto, H., Schurr, B., 2021. DeepPhasePick: a method for detecting and picking seismic phases from local earthquakes based on highly optimized convolutional and recurrent deep neural networks. Geophys. J. Int. 227 (2), 1268–1294.

Wang, J., Chen, Y., Yu, H., Huang, M., Yang, Q., 2019. Easy transfer learning by exploiting intra-domain structures. In: 2019 IEEE International Conference on Multimedia and Expo. ICME, IEEE, pp. 1210–1215.

Woollam, J., Münchmeyer, J., Tilmann, F., Rietbrock, A., Lange, D., Bornstein, T., Diehl, T., Giunchi, C., Haslinger, F., Jozinović, D., et al., 2022. SeisBench—A toolbox for machine learning in seismology. Seismol. Soc. Am. 93 (3), 1695–1709.

Woollam, J., Rietbrock, A., Bueno, A., De Angelis, S., 2019. Convolutional neural network for seismic phase classification, performance demonstration over a local seismic network. Seismol. Res. Lett. 90 (2A), 491–502.

Xiao, Z., Wang, J., Liu, C., Li, J., Zhao, L., Yao, Z., 2021. Siamese earthquake transformer: A pair-input deep-learning model for earthquake detection and phase picking on a seismic array. J. Geophys. Res. Solid Earth 126 (5), e2020JB021444.

Yeck, W.L., Patton, J.M., Ross, Z.E., Hayes, G.P., Guy, M.R., Ambruz, N.B., Shelly, D.R., Benz, H.M., Earle, P.S., 2021. Leveraging deep learning in global 24/7 real-time earthquake monitoring at the National Earthquake Information Center. Seismol. Soc. Am. 92 (1), 469–480.

Zhang, M., Liu, M., Feng, T., Wang, R., Zhu, W., 2022. LOC-FLOW: An end-to-end machine learning-based high-precision earthquake location workflow. Seismol. Soc. Am. 93 (5), 2426–2438.

Zhao, M., Xiao, Z., Chen, S., Fang, L., 2022. DiTing: A large-scale Chinese seismic benchmark dataset for artificial intelligence in seismology. Earthq. Sci. 35, 1–11.

Zhu, W., Beroza, G.C., 2019. PhaseNet: a deep-neural-network-based seismic arrival-time picking method. Geophys. J. Int. 216 (1), 261–273.

Zhu, J., Fang, L., Miao, F., Fan, L., Zhang, J., Li, Z., 2022. Deep learning and transfer learning of earthquake and quarry-blast discrimination: Applications to southern California and eastern Kentucky.