Geophysical Prospecting

EAGE EUROPEAN ASSOCIATION OF GEOSCIENTISTS & ENGINEERS

Check for updates

Geophysical Prospecting, 2022, **70**, 1093–1115 doi: 10.1111/1365-2478.13192

Convolution neural network application for first-break picking for land seismic data

Georgy N. Loginov[1,2]*, Anton A. Duchkov[1,2], Dmitry A. Litvichenko[3] and Sergey A. Alyamkin[4]

[1]*Institute of Petroleum Geology and Geophysics SB RAS, Novosibirsk, 630090, Russia,* [2]*Novosibirsk State University, Novosibirsk, 630090, 2 Pirogova Street, Russia,* [3]*Gazpromneft-NTC, Tyumen, 625048, 14, 50 Let Oktyabrya, Russia, and* [4]*Expasoft LLC, Novosibirsk, 630090, Nikolaeva 12, 900, Russia*

Received June 2021, revision accepted February 2022

ABSTRACT

An automatic and robust algorithm for the first-break picking is necessary to build the near-surface velocity model. We propose the algorithm based on a convolution neural network. The introduced first-break picking strategy and neural network architecture are suited for processing large volumes of seismic exploration data with reasonable computational resources. To develop an optimal neural network topology and architecture, extensive testing was performed. We compared several architectures of neural networks, including one- and two-dimensional approaches. Our tests justify that the one-dimensional approach (trace-by-trace processing) provides the most reliable results in the case of first-break travel-time variations typical of complicated near-surface structures. This study demonstrates that the four-layered neural network trained on 5,000 traces is enough for robust first-break picking. The algorithm is evaluated on two land-acquisition field datasets from West Siberia with a total used size of about 7 million traces. The first dataset is used for training, and the second one is used only for testing. For both datasets, the error between the original and the predicted first breaks is not more than three samples for 95% of traces. The final evaluation is done by a comparison of seismic stacks to prove the benefits of the approach and its robustness for offsets over 600 m. Finally, the influence of choosing the locations for the training dataset is discussed, and a strategy for using the proposed approach in production work is introduced.

Key words: Automated classification, near surface, neural network, refraction, signal processing.

INTRODUCTION

Seismic acquisition on land often takes place in areas with complicated near-surface velocity structure including rough topography, presence of discontinuous high-velocity permafrost or basalt layers, low-velocity waveguides, etc. Such survey features produce kinematic anomalies and can strongly distort the resultant seismic image. Static corrections (constant time shifts in traces) are used to remove the influence of the near-surface structure (Yilmaz, 2001) with two steps. First, the near-surface velocity model is used to build the long-period component of the static correction. Second, the residual static analysis of the reflected waves is used to remove residual short-period corrections. Dense three-dimensional (3D) acquisition systems allow using the first-break picks for this purpose. After updating the near-surface model with the first-break picks, this model can be introduced into a bigger velocity model for further imaging and estimating the prior static corrections.

*E-mail: loginovgeorgy@gmail.com

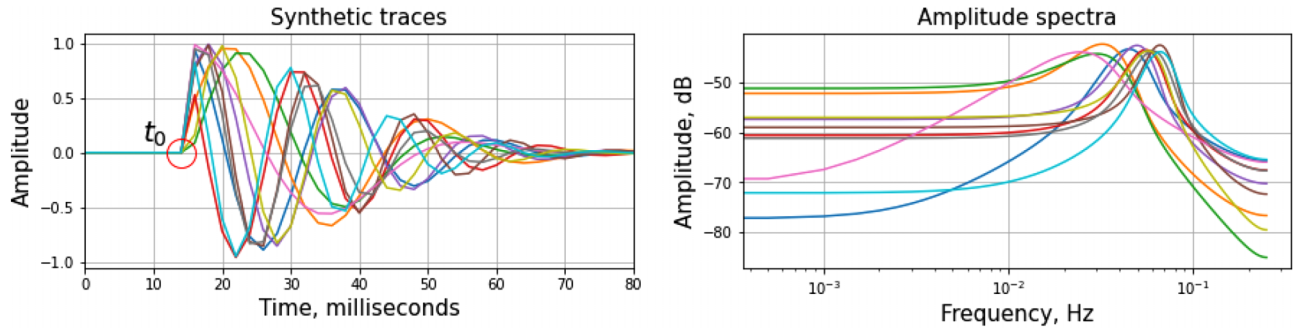

Figure 1 Synthetic data. Representative example of synthetic traces seeded at the same t_0 with the random parameters of signal shape. Left: traces; right: corresponding amplitude spectra.

The first-break picking is an important procedure in the processing of land seismic data and should be fast and automatic as seismic exploration surveys produce large volumes of data. Besides the land seismic data, the picking may be referred to as a more general travel-time picking problem, for example as the arrival-time picking of P- and S-waves in passive-seismic, seismological (Sabbione and Velis, 2010) and microseismic (Akram and Eaton, 2016) data processing. In this case, the problem becomes more complicated due to the complicated source signature and the necessity to pick multiple wave modes (P- and S-waves), especially for the anisotropic case (Yaskevich *et al.*, 2016).

Most travel-time picking algorithms utilize single-channel analysis, that is processing in a trace-by-trace manner. Most of the approaches perform signal analysis in a sliding window following ideas of the classic short term average (STA)/long term average (LTA) method (Peraldi and Clement, 1972; Allen, 1978). The methods based on kurtosis and high-order statistics of the signal (Sleeman and Van Eck, 1999; Akazawa, 2004) can be found as more accurate but require careful parameter tuning and are computationally more expensive. Tan and He (2016) used cross-correlation of neighbour traces, that is multi-trace analysis. Tselentis *et al.* (2012) suggested a classification of the existed travel-time picking methods: window-based analysis, envelope-based analysis, waveform correlation, maximum-likelihood analysis, autoregression, etc. Sabbione and Velis (2010) and Akram and Eaton (2016) performed thorough testing and comparison of different methods of arrival-time picking in synthetic and real data. They give recommendations on the optimal choice of the analysis window length and other picking parameters.

Several algorithms of travel-time picking based on machine learning have been presented in the literature. They make use of fuzzy logic (Hashemi *et al.*, 2008), pattern recognition (Joswig, 1990), singular value decomposition (Ursin and Zheng, 1985), support vector machines (Duan and Zhang, 2019), and principal component analysis (Hagen, 1982). The auxiliary problem of signal detection in continuous data records (triggering) was also considered as a problem of supervised (Qu *et al.*, 2019) or unsupervised machine learning (Huang, 2019). Mousa *et al.* (2011) considered the picking problem as image segmentation implemented by the method of projection onto convex sets. A huge comparison of machine learning techniques applied to the first-break picking problem was carried out by Ayub and Kaka (2021). It is important to mention an approach proposed in Turhan Taner *et al.* (1988), based on a supervised post-picking regression analysis, that incorporates reciprocity principles.

There is some history of using feed-forward artificial neural networks for automatic signal detection and first-break picking and other geophysical applications (Van der Baan and Jutten, 2000). Usually, it is formulated as a classification problem, that is one has to mark every sample as belonging to one of the two classes: first break or non-first break. Initial papers mostly used a single-layer fully connected neural network architecture (Murat and Rudman, 1992; McCormack *et al.*, 1993; Wang and Teng, 1995; Dai and MacBeth, 1997), followed by multi-layered fully connected neural networks (Zhao and Takano, 1999; Gentili and Michelini, 2006; Madureira and Ruano, 2009; Maity *et al.*, 2014; Mousavi *et al.*, 2016; Akram *et al.*, 2017). The main strategy was to use the seismic trace itself and auxiliary additional features as an input to a neural network: spectra of the traces, root mean square (RMS) values, envelope or other statistics of the seismic trace. The majority of those approaches implied using the detection functions computed by deterministic methods (STA/LTA and other window-based methods) as input to a neural network. The neural network architecture here used as a tool for easy combination of a priori assumptions about the signals that are used in classic methods. Unfortunately, in wide

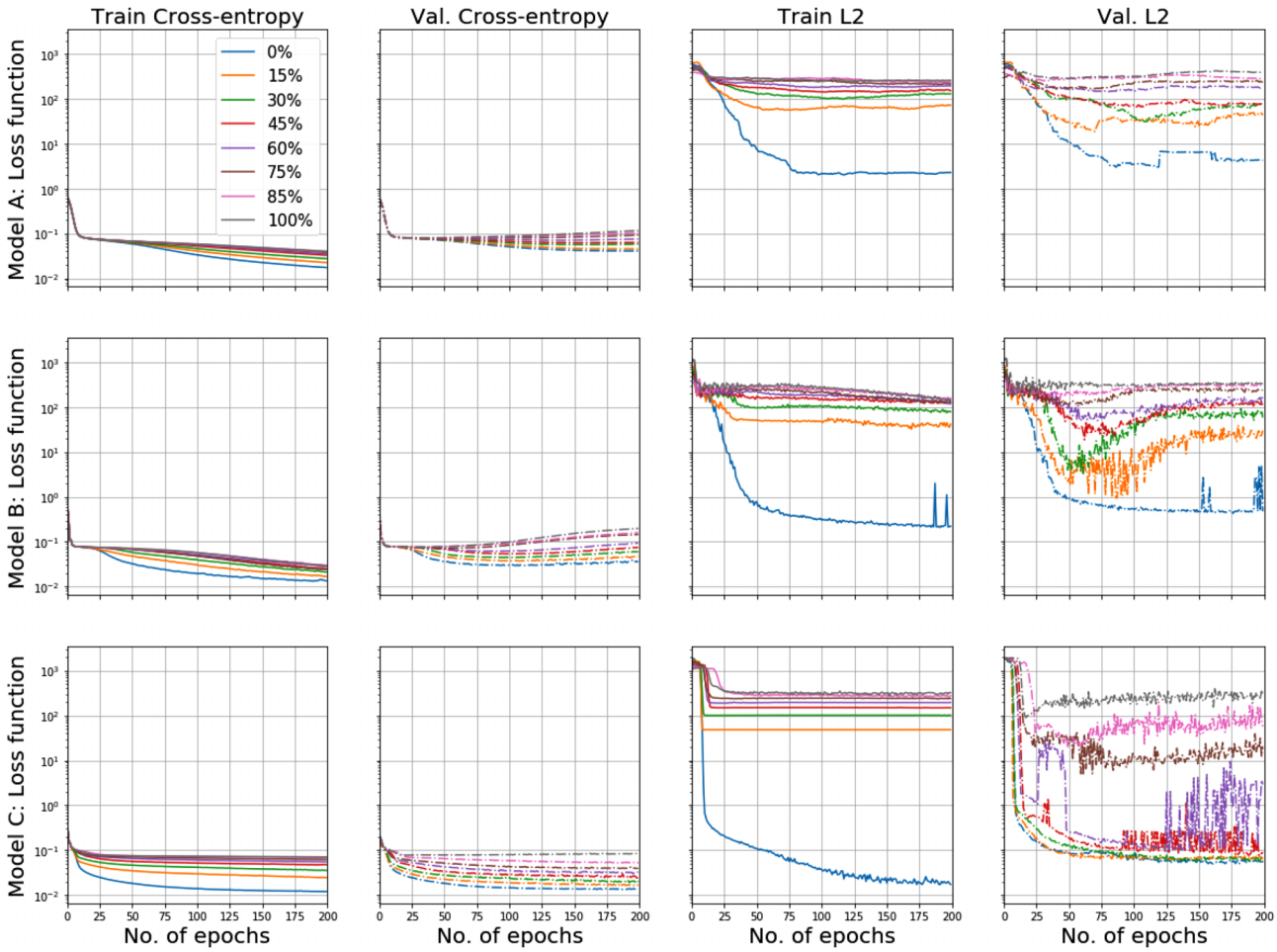

Figure 2 Synthetic data. Training curves for a synthetic dataset for three corresponding models. The colour of curves corresponds to the percentage of spoiled picks used for training.

practice, this strategy does not show a significant improvement of the results but have obvious drawbacks: the necessity of tuning parameters of classic methods and the increased computational burden of the whole algorithm.

A recent trend is to use convolution neural networks (CNNs) for solving various problems in seismic processing. Perol *et al.* (2018) used the feed-forward CNN for seismological record classification and approximate location. Yu *et al.* (2019) gave a detailed overview on using CNN models for noise suppression in seismic data (also see Dong *et al.*, 2019, for low-frequency de-noising). Sun *et al.* (2018) performed an analysis of seismic gathers to identify if the shingling effect was present in the first-arrival of waveforms. Wu *et al.* (2019) used the CNN architecture designed for two-dimensional (2D) image analysis for travel-time picking and considered it as a segmentation problem. Zhu and Beroza (2018) suggested the use of the CNN layers for attribute computation while the last dense layer of the neural network performed the classification. Zheng *et al.* (2017) used recurrent neural networks to detect sequences of events, in particular, the S-wave arrival that follows the P-wave arrival. Duan and Zhang (2020) used a residual 2D convolution network trained on about 2.5 million of traces and achieved an accuracy of about 95–97% for different cases. In Hollander *et al.* (2018), the 2D CNN model was incorporated with the classic envelop calculation approach. In Cova *et al.* (2020), a constrained poling was used to regularize 2D CNN based on the UNet (Ronneberger *et al.*, 2015) architecture, where constrain was guided by effective velocity estimation. A semi-supervised learning for 2D CNN model is introduced in Tsai *et al.* (2019). Zwartjes *et al.* (2020) considered comparison of different convolution-based models on real data and concluded that the best results are achieved with UNet architecture consisting of seven convolution blocks. In Xie *et al.* (2019), authors used fully convolution 2D CNN

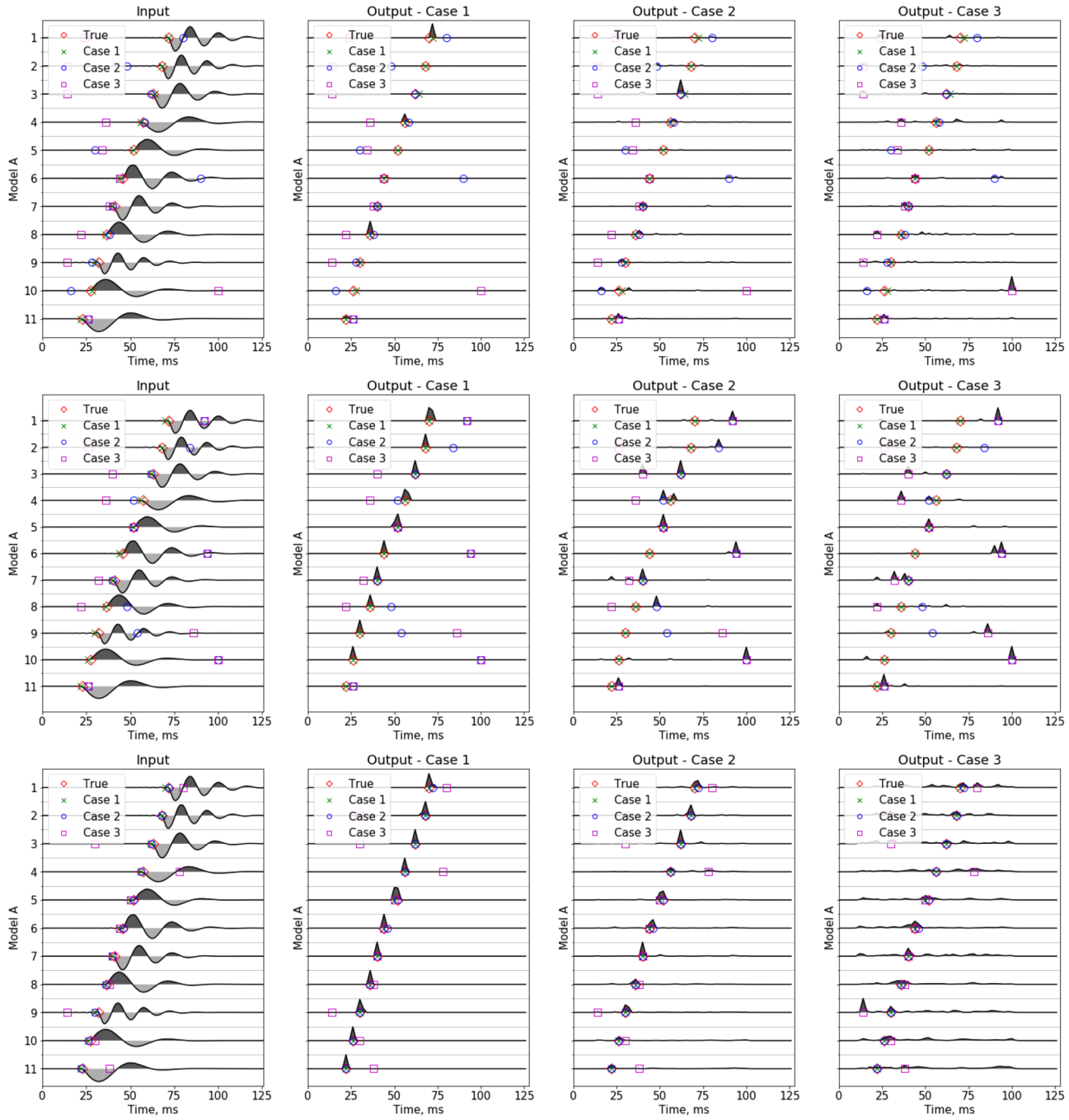

Figure 3 Synthetic data. Compare applications of Models A, B and C for *Cases 1*, *2* and *3*, respectively. Horizontal panels are correspondingly represent models. First column refers to input traces; second column refers to the first-break probability for *Case 1*, third column for *Case 2* and fourth column for *Case 3*.

and applied the transfer learning technique. A semantic segmentation approach for noisy data is suggested in Xu *et al.* (2021) that incorporates a UNet-based model, which incorporated some conventional attributes as input in addition to seismic trace.

For field data, the common practice is to test several methods on a subset of data. After the choice of preferable parameters and algorithms, the picking is performed for the entire dataset. The quality control (QC) of the first break picking can be used to choose the algorithms and parameters. The QC

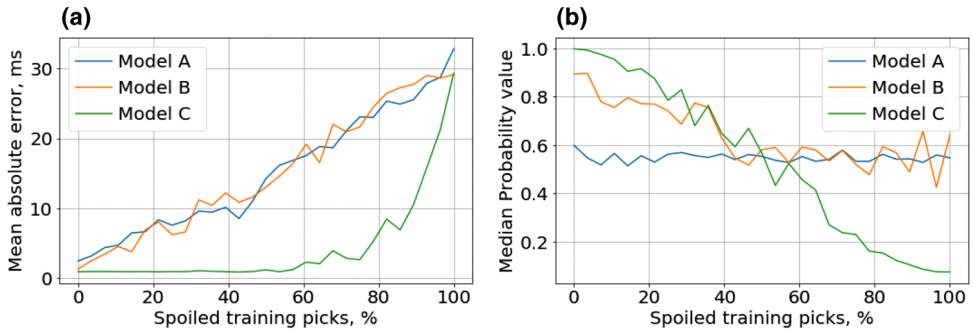

Figure 4 Synthetic data. Evaluation of Models A, B and C for different percent of spoiled picks used for training. Left: mean absolute error between true and predicted first-break picks (milliseconds); right: median probability value of the predicted first-break pick.

usually combines the statistical analysis of the first-break picks (including the checking of correlation of first-break picks vs. the signal to noise ratio, central frequency and other wavefield attributes) and resultant seismic imaging, which is often characterized as empirical. The empirical estimation implies checking the continuity of primary events, their amplitude behaviour over the section and consistency with geological expectations. The choice of the optimal workflow for the specific dataset always includes an empirical estimation (e.g. the quality of pre- or/and post-migration stack) and may use some weighted combination of picking results derived in different ways. The real data will always require close consideration of different aspects, but some general choice of architecture can be done in an attempt to apply to an arbitrary dataset. We do not expect that the CNN model must be trained once and applied to any data but the choice of CNN architecture can be fixed to train 'from scratch'. With such a statement, the next question is how big should be the size and variability of the training dataset and ways to sustain it while training.

In this paper, we focus on the development of a neural network–based technique for fast and accurate picking of first breaks for field land seismic data. We are pursuing the strategy for first-break picking, which can be reproduced for arbitrary field data in three steps: (1) generate training picks from target data; (2) train the neural network; (3) predict first breaks for an entire volume. On the other hand, for neighbouring geological regions we show that a pre-trained CNN model performed with sufficient quality. The paper is organized in the following manner. First, we explain the theory of neural networks and show some synthetic tests. After that, the algorithm is evaluated on two field datasets from the land acquisition in West Siberia with a total size of about 7 million traces. Finally, we discuss the first-break picking strategy. There are the following issues, which we especially have focused on while developing the workflow:

1. the field data are characterized by a strong variation of signal characteristics that bring a strong curvature of pick profiles;
2. the training first-break picks (markup) can be non-reliable due to the complexity of a region, miss-picking, noise and a low capacity of standard methods used to conduct the training picks,
3. as the picking of training first breaks in field data will still be a problem, it is necessary to make the estimates of which training size is sufficient, the smaller the training set, the faster it can be examined for reliability.

CONVOLUTION NEURAL NETWORK MODEL DEVELOPMENT

In this study, we use supervised learning with a convolution neural network (CNN) for automatic first-break picking in land seismic data. The CNN architecture seems to be natural in this problem because of the classic model of a seismic trace as a convolution of the source signal with the reflectivity function. In the real data that the first-arrival waveforms may vary considerably from trace to trace and can be complicated by noise. Thus we focus on addressing the problem of the first-break picking for each trace independently. Group analysis of the travel-time picks can be done as a next post-processing step, where it is more natural to take into account the reciprocity theory and acquisition geometry specification (post-picking regression, correlation and tomography).

In this section, we describe the main development steps of our first-break picking algorithm: (1) problem formulation, (2) steps of CNN construction, (3) strategies of CNN training, and (4) QC of the results. There are three primitive architectures tested on synthetic data to conduct the baseline for the choice of architecture blocks for field data.

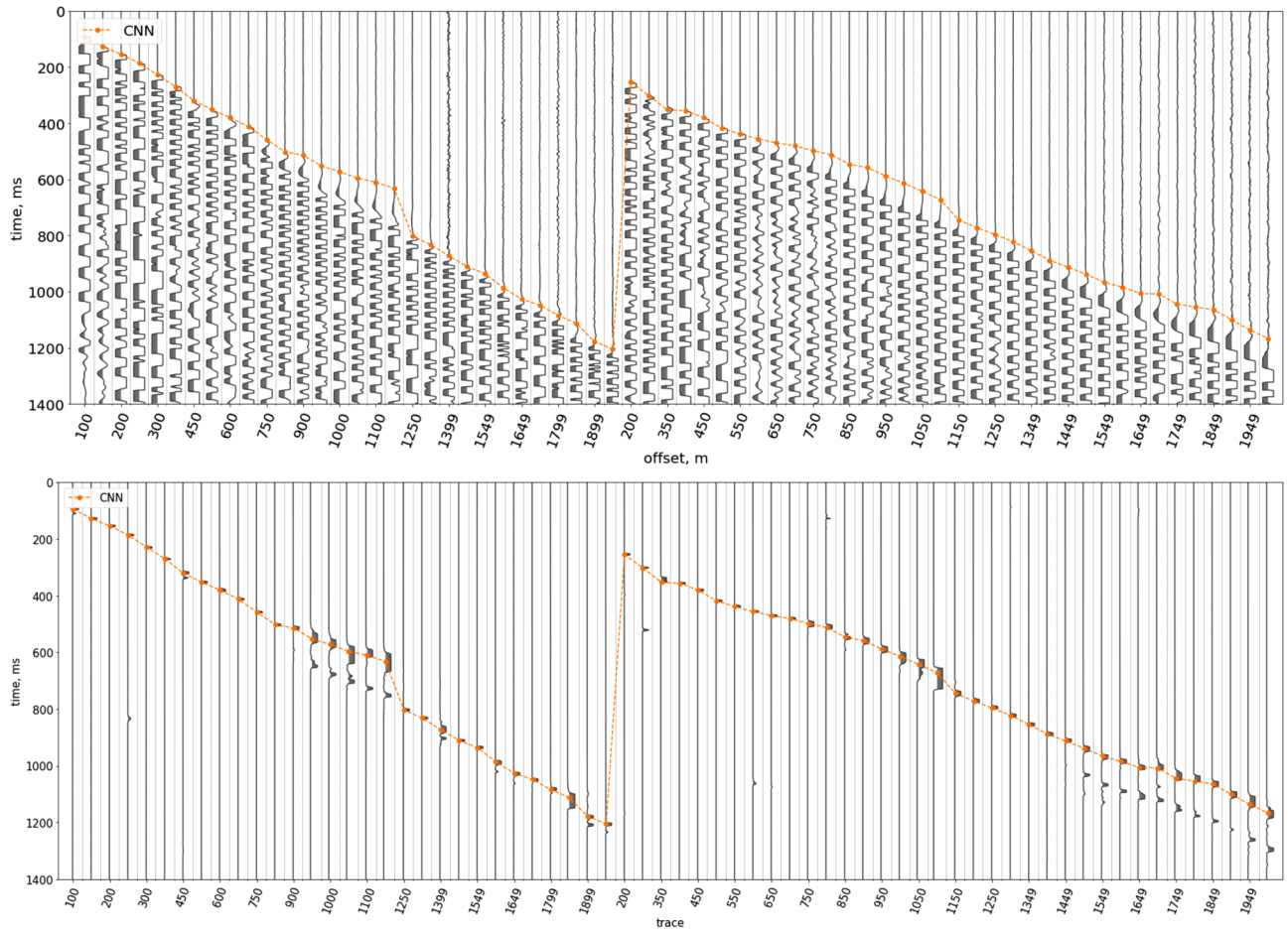

Figure 5 Dataset 1. Examples of first-break picking results for gathers with the presence of low-frequency waves in first arrivals. Top panel: gathers; bottom panel: CNN model output (detection function); orange: picks predicted by the CNN model.

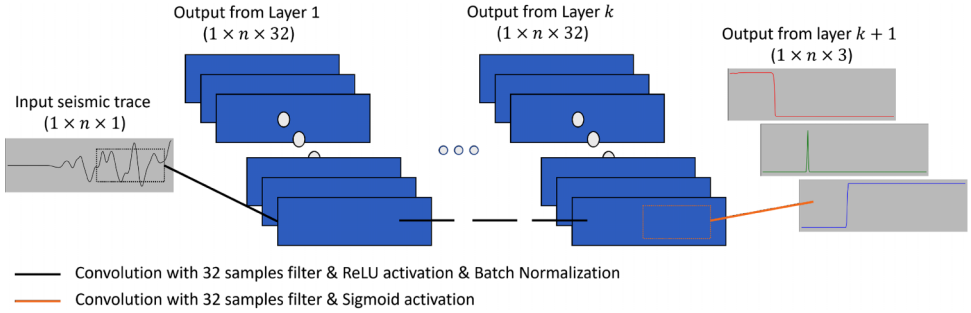

Figure 6 Schematic visualization of the proposed CNN architecture.

First-break picking as a classification problem

The standard machine-learning approach to the first-break picking is to formulate it as a classification problem. For each input trace, we get an output in the form of the markup matrix with the number of columns equal to the number of time samples in the seismic trace and the number of rows equal to the number of classes. The element of this matrix is equal to 1 if the trace sample belongs to the corresponding class, and it is equal to 0 otherwise. For the first-break picking problem,

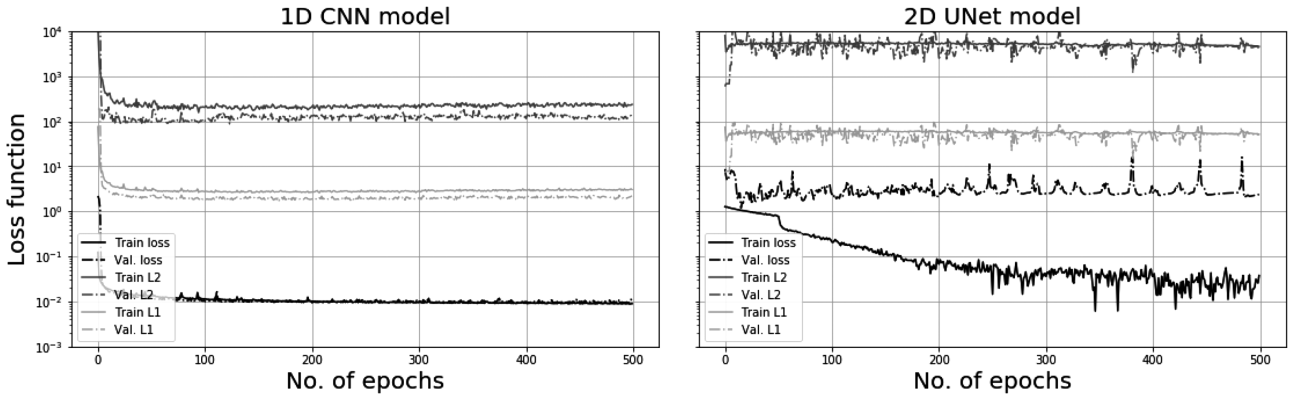

Figure 7 Field data. Comparison of training curves between 1D CNN (left) and 2D UNet (right) models

it is used to introduce two classes: 'first break' and 'non-first break'. Such classification brings a strong imbalance between classes as there should be only one sample belonging to the class 'first break', while all other samples should belong to the class 'non-first break'. One way to mitigate this problem is to use weights for the classes in the loss function during the CNN training. But it requires tuning the weights, that is introducing additional parameters. We propose a new approach and use three classes instead:

- 'noise';
- 'signal';
- 'first break'.

Let us assume that all samples preceding the first break should belong to the class 'noise', while all samples following the first break should belong to the class 'signal'. Then two classes, 'noise' and 'signal', appear to be well balanced, and one can use a loss function without the weighting of classes. Our tests show that this new approach provides a more robust picking of the first breaks. We discuss the benefits of three-class markup in the section of *Field data CNN tuning*, and some general recommendations for imbalanced datasets can be found in He and Ma (2013) and Fernández *et al*. (2018).

Convolution neural network architecture

By the CNN architecture, we mean the number of hidden layers (k) and their structure. Each convolution layer contains the following parameters (corresponding hyper-parameters are listed in parentheses):

- convolution layer (number of filters, filter size, padding type);
- fully connected layer (number of units);
- activation function to introduce non-linearity (*sigmoid*, *softmax*, *rectified linear unit*);
- batch normalization and mean value removal for improving the performance of the next layer;
- dropout to reduce over-fitting (rate).

Plenty of testing can be done by searching for optimal hyper-parameters of convolution layers (number of filters and their size). There are many approaches to search for optimal hyper-parameters of the neural network. These experiments are used to find such hyper-parameters that would allow to achieve the minimal error and best performance metrics. It can be done in an automatic manner using algorithms such as grid search, random, generic (Liashchynskyi and Liashchynskyi, 2019), population-based algorithm (Nalçakan and Ensari, 2018) or a specific algorithm for CNNs (Cui and Bai, 2019). In our research, we have also performed internal testing of different hyper-parameters (mainly regarding the filter length and amount per convolution layer). After the testing, we decided to use probably the most popular setup, which is 32 filters with a length of 32 samples per layer. Referring to the common sampling step of 2 ms of seismic traces, the filter of length 32 gives a resolution of 15 Hz. We chose the same parameters for our CNN as our tests demonstrated that the use of longer filters does not improve the CNN performance, but shorter filters result in an under-fitting problem. According to the internal tests, the rectified linear unit activation function provides better performance, which is in agreement with the results of Krizhevsky *et al*. (2012). Batch normalization (mean-average removal and normalization) was used to improve the training performance (cf. Ioffe and Szegedy, 2015). Finally, the dropout procedure was used to reduce over-fitting with the default rate of 0.5.

It is popular to use the pooling procedure in the CNN layers. It provides down-sampling of the signal, which is believed to improve the CNN performance in the problems of object detection in images (Goodfellow *et al*., 2016). We chose

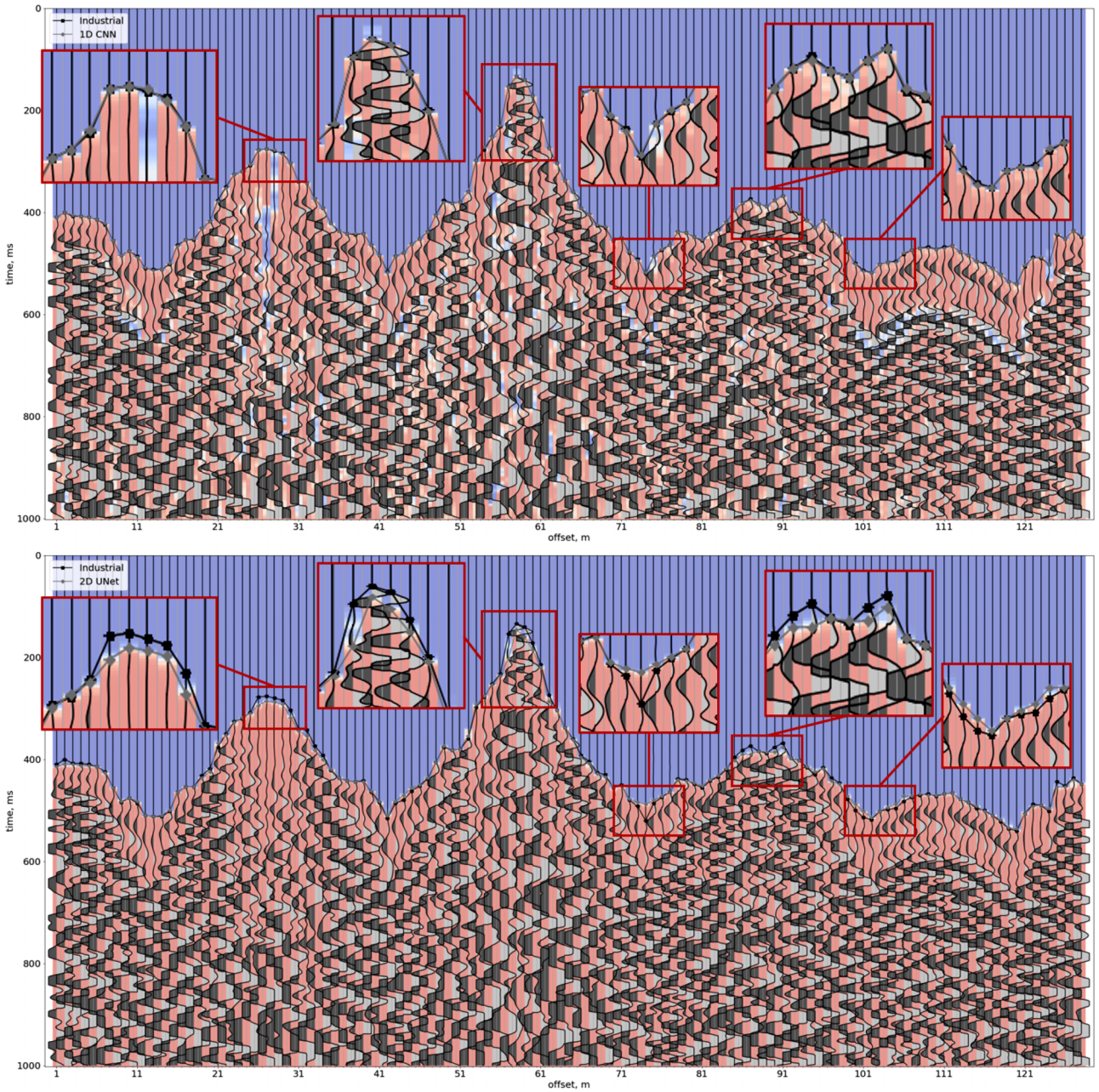

Figure 8 Field data. Picking examples by 1D CNN (up) and 2D UNet (bottom) models, background colour is representing a predicted probability for class 'signal'. The background colour map is in blue-white-red, where blue represents that the probability is zero, white indicates that the probability is 0.5 and red indicates that the probability is 1.

to use neither pooling nor dense output layers. Without pooling, we get the same length of the CNN output as the length of its input. Also avoiding dense layers, we get a fully convolution CNN model which can be applied to traces of different lengths (number of samples) not equal to the trace length in the training set. In addition, while using only convolution layers we stay consistent with the convolution model of a seismic trace.

The CNN output is a matrix of the same size as the original markup matrix. It has three columns of the length of the input trace. Each column contains the probability (value between 0 and 1) of the trace sample that belongs to one of the

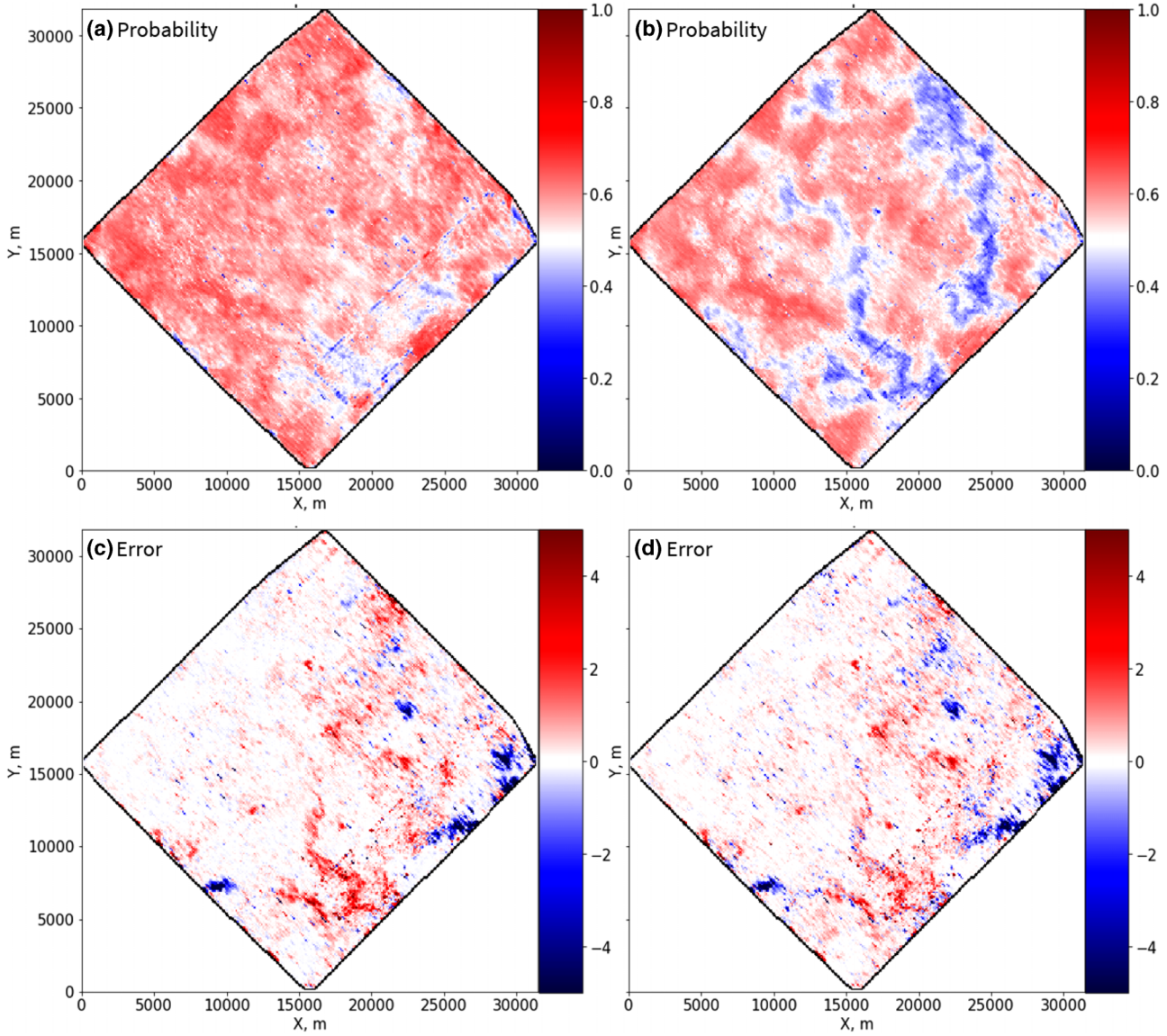

Figure 9 Quality control of picking results by using 1D CNN. The left column - three-class markup (a, c) and the right one is for for the binary markup (b, d). The top row - probability value distribution (the predicted probability value of the first-break sample on seismic traces). The bottom row - mean squared error.

three classes: 'noise', 'signal', and 'first break'. Then we choose the position of the maximum value in the third column as the first-break pick. Thus we get the first-break pick and the estimation of the picking reliability (probability).

Training procedure

Let us mention some most important hyper-parameters of the CNN training (learning) process: batch size, type of loss function, optimization algorithm, learning rate and number of epochs. We compute the loss (misfit) function as the cross-entropy between original and predicted markup matrices. While training the neural network, the loss function is computed for a subset of training data called a batch. Zhang *et al.* (2017) provided recommendations on how to choose the batch size. We used a batch size of 64 traces. The smaller batch size may lead to an under-fitting problem, and the larger values did not improve the prediction accuracy. For training, we used the Adam optimization algorithm (Kingma and Ba, 2014) with a learning rate of 0.005. The number of epochs

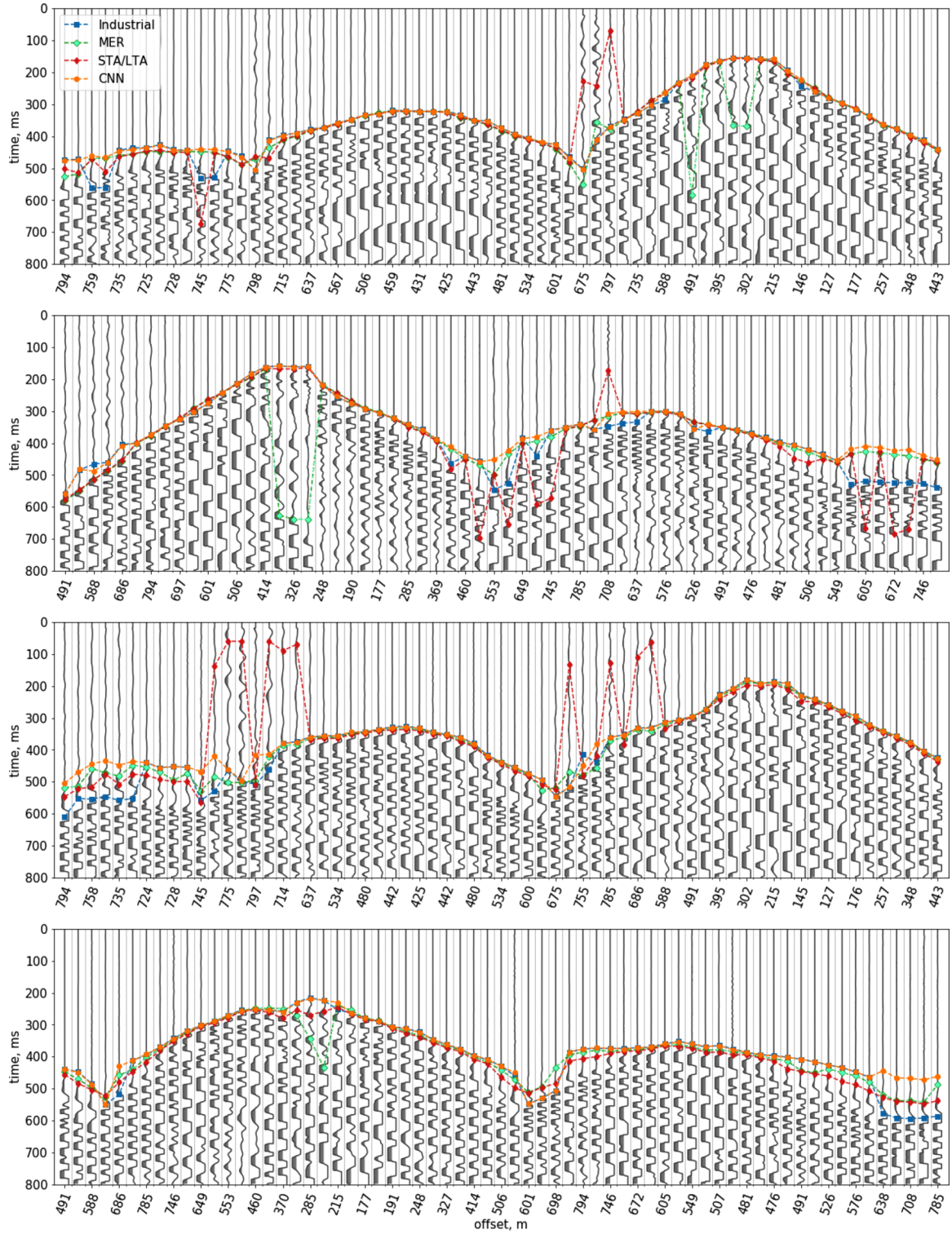

Figure 10 Dataset 1. Comparison of first-break picking algorithms for common-shot gathers: blue, original (industrial) picks; orange, our CNN prediction; green, MER prediction; red, STA/LTA prediction.

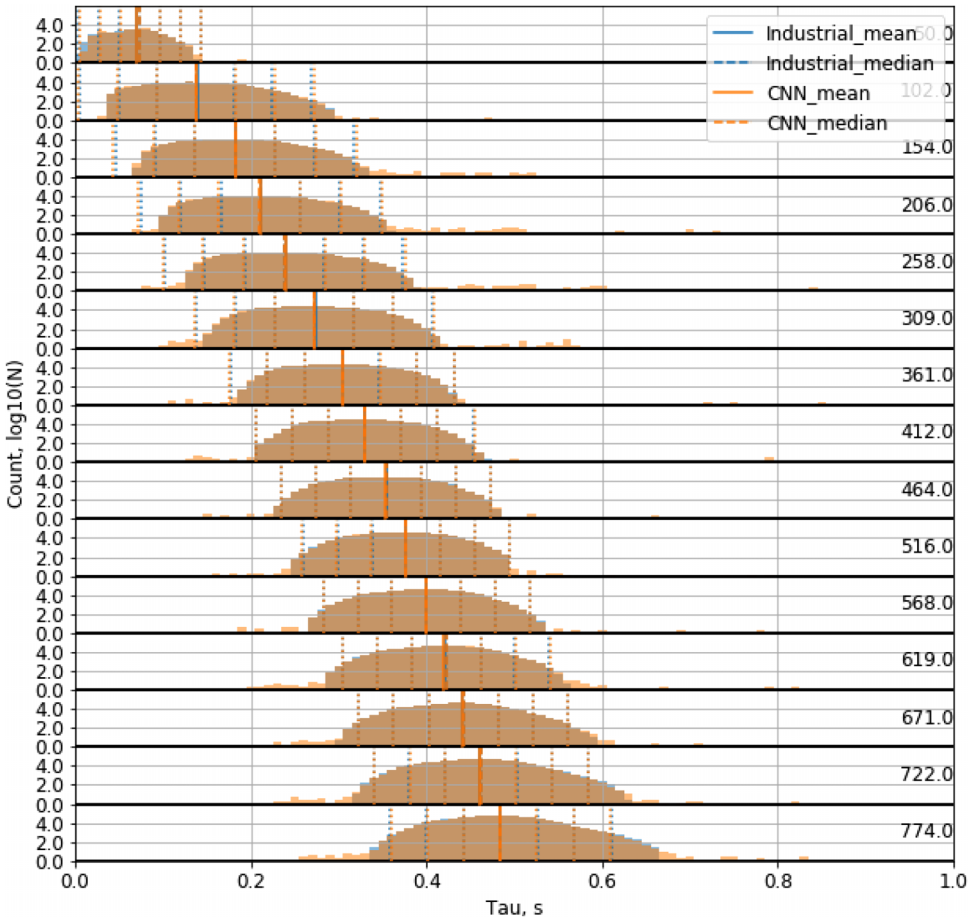

Figure 11 Dataset 1. Histograms of the first-breaks distribution for offset bins (in logarithmic scale); original/industrial travel times: blue, CNN predicted: orange; solid vertical lines: mean value, dashed lines: median value; dotted lines correspond to refer to 4 σ, 3 and 4 σ intervals. The histograms are stacked by vertical axes, and the offset value is marked on the right.

(one pass over full training dataset) is optimized by analysing the loss-function decrease. It should be large enough to overcome under-fitting and stop before running into the problem of over-fitting.

Synthetic tests

There is a big variety of approaches for neural network development, we conduct our tests using common practices of machine learning and observations of field data. According to the specifics of field datasets, which we consider in this study, the key problem of first-break picking is a strong variability of signal characteristics: lateral changes of velocity resulted in a complicated profile of first-breaks, signature, amplitude, central frequency, phase characteristics and interference. To mimic the variations of first breaks and estimate the capacity of performance of neural networks, we conducted tests on primitive architectures of noise-free synthetic data, but with strong changes in signature.

We created a synthetic dataset of 1,000 traces with a length of 128 ms and a sampling rate of 2 ms. The signal's signature was created by the random uniform distribution of its parameters. The 700 traces are used for training, and 300 traces are used for validation. The seismic traces are generated by Chirplet function (1) (e.g. Boßmann and Ma, 2015), where t is time, $t_0$is the first-break time, α is the bandwidth factor, $\beta$is the symmetry coefficient, $\omega$is the central frequency and γ is attenuation. Figure 1 shows the example of synthetic traces generation. As a demonstration, we created all the traces with the same first-break time ($t_0 = 15$ ms) to highlight how flexible can be the Chirplet model to mimic the variations in the seismic signal. Note that all the created synthetic traces have an obvious first-break time, shown as a trough that is marked with a red circle in the left plot in Figure 1. The first-break

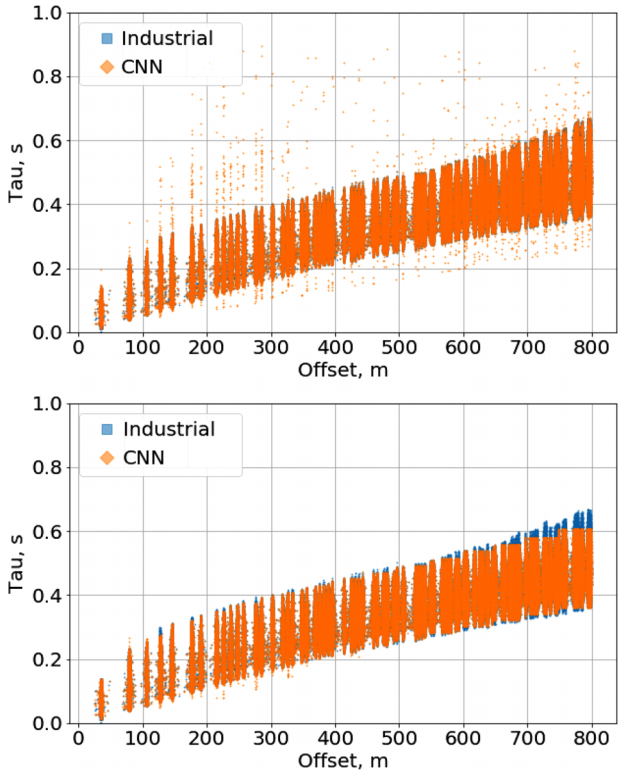

Figure 12 Dataset 1. Comparison of tau-offset cross-plots between the original/industrial (blue) and CNN-predicted (orange) first breaks; top panel: CNN-predictions before post-processing; bottom panel: after post-processing and outlier removal.

time is generated from a random uniform distribution in a range from 12 to 100 ms.

$$g(t) = \sin(2\pi\omega(t - t_0) + \gamma(t - t_0)^2),$$
$$h(t) = \exp(-\alpha(1 - \beta\tanh(t - t_0))(t - t_0)^2), \quad (1)$$
$$f(t) = g(t)h(t).$$

The common practice of neural network development implies the usage of fully connected (dense) layers on the last layers of the architecture. This practice is used to be ranked as a regularization technique and applied to one-dimensional (1D) and two-dimensional (2D) models, including such popular models as UNet (Ronneberger *et al.*, 2015), ResNet (He *et al.*, 2016), Inception (Szegedy *et al.*, 2015), etc. Our primitive test shows that this practice is not well applicable to the situations where the training datasets are non-reliable and lead to over-fitting. As far as it can be judged from the seismic processing practice, picking for field datasets is often ambiguous. To mimic a field-data scenario of first-break picking, we test how the correctness of a markup may influence training results. The synthetic data are complicated only by signal shape

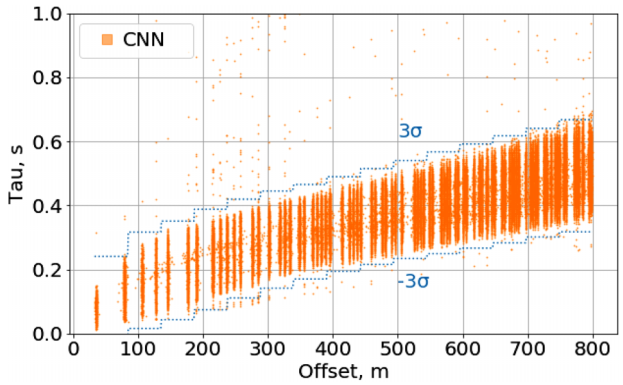

Figure 13 Dataset 2. Tau-offset cross-plot of CNN-predicted first breaks; dashed lines: 3σ interval for outlier removal.

variations, and no interference or noise is considered. In such a way, we want to make a decision on which architecture is more robust to be used for field data.

We consider three 1D neural network models shown in Table 1. *Model A* is an example of a classic fully connected neural network with three layers, *Model B* is an example of two convolution layers followed by two fully connected and *Model C* consists of only three convolution layers. All the models were trained with the same learning rate of 0.001 over 200 epochs by the Adam optimization algorithm. The target of training was a mask of three classes: noise, first break, and signal. The loss function is categorical cross-entropy:

$$\text{loss} = \sum -y_i \log(\hat{y}_i), \quad (2)$$

where y is a training mask, $\hat{y}$ is model output and i is an index of a class (noise, signal, first break). As a metric to quality control (QC), the training results we used the mean squared error (L2), which was calculated in between actual first-break time t_0 and prediction, where k is an index of first-break pick class:

$$L2 = \sum (t_0 - \text{argmax}(\hat{y}_k))^2, \quad (3)$$

In Table 1, the architectures of models are described. For the *fully connected* layers, we used 194 units per each layer, and for *convolution* we used 32 filters with 32 samples each. All the hidden layers are activated with the rectified linear unit function and the final with *sigmoid*. The total number of parameters is also listed in Table 1.

To mimic the field-data scenario, we have randomly generated the first-break times t_0 and corresponding synthetic traces (1). After, before the training we spoil the t_0 by introducing a random error. The error is introduced in a subset of the training data, such as the percentage of the t_0 can be called

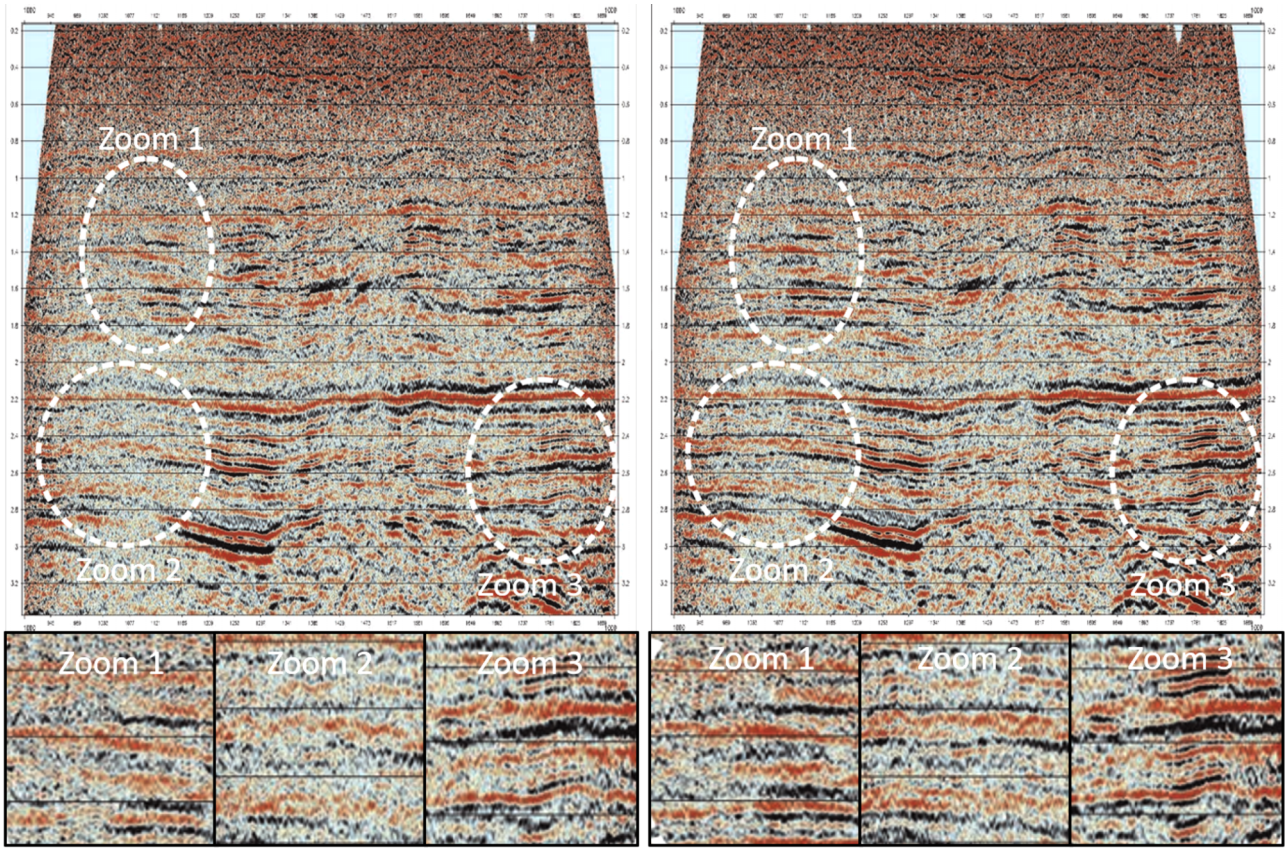

Figure 14 Dataset 1. Seismic stacks obtained with different a priori static correction estimated from original (industrial) first-break travel-times (left) and CNN-based travel-times (right). White ellipses highlight regions with the most improvement.

Table 1 Neural network models used for synthetic testing of potential capacity

Layer	Model A	Model B	Model C
1	Fully connected	Convolution	Convolution
2	Fully connected	Convolution	Convolution
3	Fully connected	Fully connected	Convolution (three filters)
4		Fully connected	
Number of parameters	83,906	88,270	36,931

spoiled. Therefore, we consider 0–100% percentage of spoiled picks used for training. Three experiments are highlighted to demonstrate the picking results for different models: *Case 1* with 100% true first breaks used as a markup; *Case 2* with 85% true and 15% false (random, spoiled); *Case 3* with 5% true and 95% spoiled picks used for training. In other words, for *Cases 2* and *3* for the corresponding percentage of spoiled picks we use wrong t_0 as a markup for training. These spoiled first breaks were not involved in generating the traces. The validation set was the same for all the experiments and did not contain the spoiled picks. For all scenarios, the traces were generated using random uniform distribution for parameters defined in function (1) and remained unchanged.

In Figure 2, we show the training curves for the experiments with different percentages of spoiled picks. The dashed lines are for validation and solid lines for testing progress over 200 epochs. The columns are representing curves for loss function per epochs: training and validation cross-entropy; training and validation mean squared error between true and predicted picks (*L2*). We have here performed a comparison of the progress of models training on 0–100% of spoiled picks. In such a way, we mimic the field data scenario when the

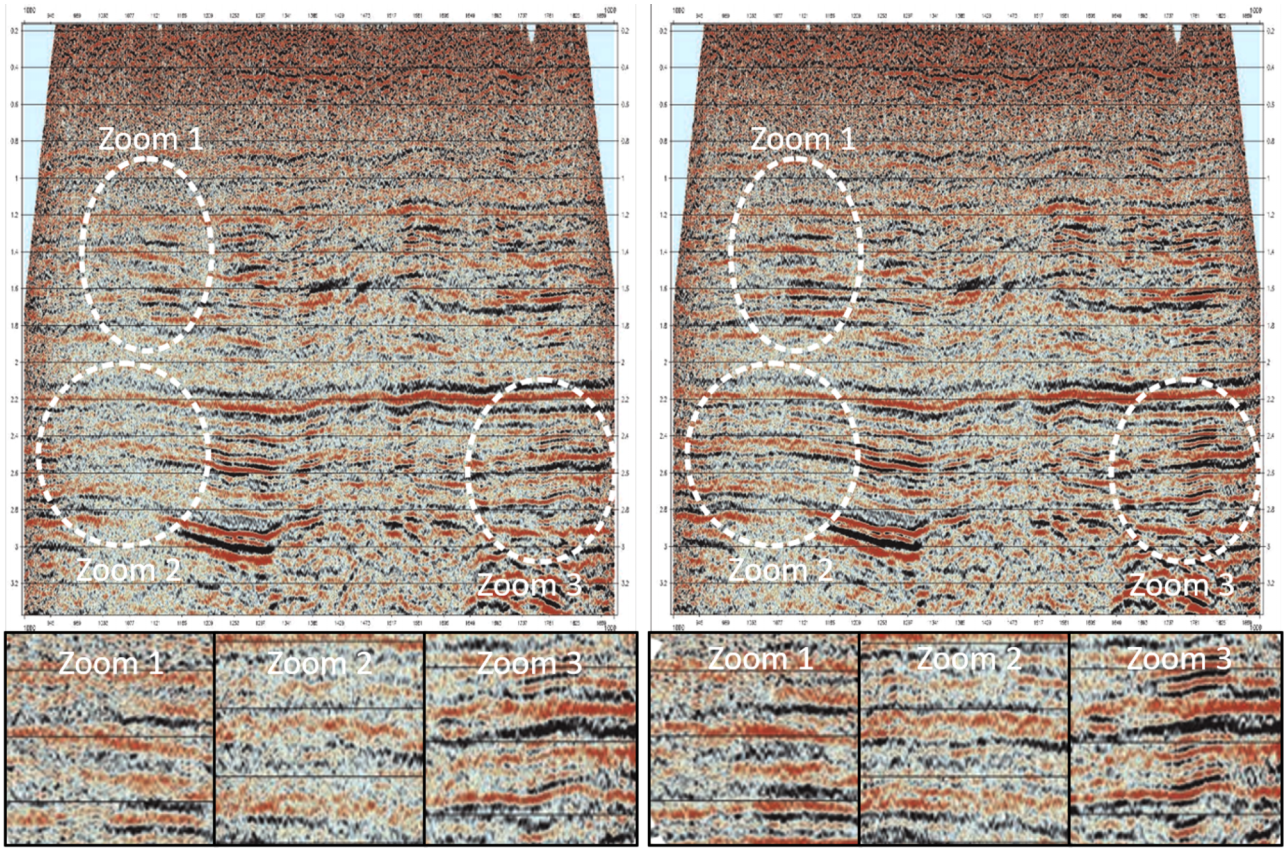

Figure 15 Dataset 2. Seismic stacks obtained with different a priori static correction estimated from original (industrial) first-break travel-times (left) and CNN-based travel-times (right). White ellipses highlight regions with the most improvement.

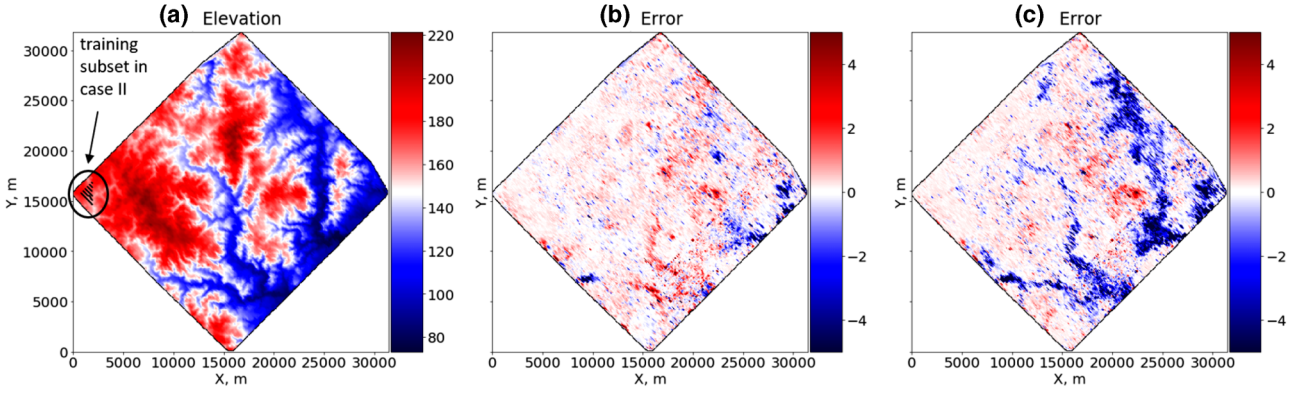

Figure 16 Dataset 1. (a) receivers elevation map (meters), black dots indicate the location of the examples of training subset for case 2); (b) and (c) relative mean error distribution (time samples); (b)random choice of the training subset; (c) training subset location (case 2).

conventional picking for training is not 100% reliable. Here we do not restrict the misfit between true and false picks as the overall data length is small, and no additional complications are introduced. One can see that while training progress Model C behaves more robustly on both metrics (2 and 3) and gives a smaller error. Models A and B are getting over-fitted at certain epochs and stop showing progress on the validation set. Though Model C is not showing progress after about 100 epochs, it is still performed with the smallest errors on validation. In Figure 3, we show an example of picking results

(left set of plots) of a gather of synthetic traces and the prediction masks for *Cases 1*, *2* and *3* are shown in corresponding columns. Each trace, including the prediction mask, is normalized on its own maximum value for viewing purposes. The vertical panels are addressed to models A, B, and C correspondingly. One can see that for *Case 1*, all the models provide a reasonable result, but the prediction mask is more reliable for models B and C. In *Case 2* (third column), it is the only Model C, which still provides correct picking of first breaks. In *Case 3* (fourth column), it is showing an extreme scenario when only 5% of training picks was true.

In Figure 4, we demonstrate the dependency between the percentage of spoiled picks and (a) mean absolute error (MAE) and (b) median probability value (MPV). One can see that the performance of Models A and B is starting to decay strongly, with the increase in the percentage of spoiled picks. On the other hand, Model C is showing acceptable MSE even up to 80% of the spoiled picks, which gives a straight argument for the usage of the purely CNN model for first-break picking. The curves in Figure 4(b) can be interpreted as the confidence of the neural network. Model A keeps almost the same probability value for all the scenarios, whereas Models B and C become less confident over the percentage of spoiled picks and Model B does it faster. We find that it is an interesting observation, as the common practice of neural networks is to include dense layers. The presented example shows that the pure convolution network may easily indicate the inconsistency of a dataset and provide predictions based on data features, rather than driven only by the markup as for purely dense neural networks. It is worth to consider a study of this observation with another research. In our current study, we will use the observations regarding purely convolution Model C to verify further tests on field data. For the given example, the threshold for percentage of spoiled picks is about 60% for Model C. Below this threshold, the model provides an acceptable mean absolute error (MAE) about 2 ms. It is wrong to conclude that such observation can be accepted for any dataset as the error in picks maybe not random, signal characteristics may strongly vary and no noise level was involved.

The behaviour of a purely convolution model may be explained as the target signal which is already a prominent feature in comparison to the background value of the seismic trace, which is zero in this experiment. Therefore, the CNN is aimed to recover a deconvolution filter to transfer the given input to a delta function, with delta at t_0. As the CNN contains many filters and two layers, it has an efficient capacity to perform a well-regularized deconvolution (roughly speaking, a specific filter for each specific signature of first-arrival waveform). Model C contains only convolution layers, and there is no obvious bias given in the spoiled picks; therefore, its random nature does not allow the CNN filters to ignore (or: avoid matching/mislead) the clear signature of the synthetics. Further tests could be done to incorporate the way of training on spoiled picks and the noise level. It is also interesting to consider the case when some shot–receiver effects are introduced in the synthetics or some clear constant or variable shift (e.g. 10 ms shift).

Thus, we can conclude that in case of high ambiguity of training markup the fully connected layers are forcing the neural network model to fit the non-reliable picks and lead to over-fitting. On the other hand, the pure convolution model is flexible enough to focus on signal characteristics that can be interpreted in terms of the de-convolution (de-signature) process, which is a conventional method for seismic data analysis. In the following, we use a pure convolution model to develop the architecture for field data.

FIELD DATA CONVOLUTION NEURAL NETWORK TUNING

We consider two real datasets, which we will further called as *Dataset 1* and *Dataset 2*. The distance between the corner points of Datasets 1 and 2 is about 500 km, and the acquisition was done in different years. These real three-dimensional exploration-seismic datasets were acquired in the north of West Siberia using an explosive source. The survey area was characterized by a heterogeneous near-surface structure. In particular, there is discontinuous permafrost in the area. In Figure 5, one can see the typical case of the presence of low-frequency wave at the first arrivals, occurring due to a velocity anomaly. One can identify up to three branches with different apparent velocities in the shot gathers (implying a three-layered near-surface model). The dataset contained about 4.5 million traces for over 30,000 sources with offsets up to 800 m (time sampling is 2 ms). *Dataset 2* was also acquired in the north of West Siberia using an explosive source and contained about 2 million traces for over 10,000 sources with offsets up to 800 m (time sampling is 2 ms). The industrial picking from *Dataset 1* was used as a training markup and was not ideal as contained obvious (visually observable), and systematic errors, that could be seen especially on offsets above 600 m. *Dataset 2* was used as a 'blind' test, that is available first-break picks were never used for additional training. Instead, we used pre-trained CNN for the automatic processing of this dataset. The first-break travel times used for training were obtained by using automatic industrial processing software, which was conducted with the following steps:

Table 2 The architecture of 1D convolution neural network, where n is the number of input trace samples and k is the number of hidden layers

Layer	Procedure	Input Size	Output Size
	Input	$1 \times n \times 1$	
	Convolution (32 filters, 32 samples, 'same' padding)		
	Activation function (*ReLU*)		
1…k	Batch normalization	$1 \times n \times 1$	$1 \times n \times 32$
	Dropout (rate of 0.5)		
k+1	Convolution (three filters, 32 samples, 'same' padding)	$1 \times n \times 32$	$1 \times n \times 3$
	Activation function (*sigmoid*)		
	Output		$1 \times n \times 3$

- preconditioning (linear move-out, band-pass filtering, gain correction in sliding window, divergence correction);
- picking using a combination of detection functions based on envelope function, cross-correlation of near traces and method of modified energy ration (4);
- post-processing by cross-validation of near traces in accordance with acquisition consistency.

The modified energy ratio (MER) function is based on the calculation of the squared average of the input signal (f) in two sliding windows, followed by each other:

$$s_i = \sum_{j=i}^{i+w} f_j^2 / \left(\sum_{j=i-w}^{i} f_j^2 + \beta \right), \qquad \mathrm{MER}_i = (|f_i| s_i)^3, \tag{4}$$

where β is a stabilizing constant and w is a window length. We are not presenting the exact chosen internal parameters of the MER function as the detection function is a combination of several algorithms and characterized by internal software parameters.

Convolution neural network models

In this section, we provide the results of comparison between one-dimensional (1D) and two-dimensional (2D) CNN models and testing of hyper-parameters (number of layers and size of training dataset). While comparing 1D and 2D CNN models, we are mainly trying to understand whether the 2D CNN is able to boost significant accuracy and other performance metrics. It is rather hard to compare the 1D and 2D models in terms of number of layers and parameters as the 2D. Thus we focus on the dimensional capacity of 2D in comparison to 1D. The 1D CNN model architecture is presented in Table 2. For the 2D model, we used a popular UNet architecture, which is described in Ronneberger *et al.* (2015).

In Figure 6, we illustrate a schematic visualization of the CNN architecture presented in Table 2. The figure shows how the input seismic trace is transformed to 32 traces after application of the first layer of CNN, which included convolution, activation by a rectified linear unit (*ReLU*), batch normalization and dropout while training. Each of 32 channels of the first-layer output is derived by corresponding filters of 32 sample length. We have conducted testing of a different number of layers, but the last one always contains three filters for three target classes correspondingly: noise, first break and signal. As we do not apply any down-sampling function, the number of samples for the last layer is the same as for input.

The comparison between 1D CNN and 2D UNet should also consider the difference of shape of training data and the ways of its compilation. In the case of the 1D model, the training examples are independent from each other and each training example is a single seismic trace with the corresponding mask; both have a length of 501 time samples. The selection process can be focused on choosing the location of traces or other auxiliary attributes. In the case of 2D, it is necessary to consider the way of creating the 2D subset of traces (following the common receiver, shot or common depth point (CDP) point) and the number of traces per example. For our tests, we formed the subset for UNet training by a set of 3,000 2D common-shot gathers with the size of 256 traces and 501 time samples per training example. For the UNet training, we also use transposed convolution layers for the decoding part (Szegedy *et al.*, 2015; Dumoulin and Visin, 2016) and skip-connections (Ronneberger *et al.*, 2015; He *et al.*, 2016) to pass the residuals from the encoding to decoding part. To support UNet training with additional examples, we have used data-augmentation techniques, including polarity flipping, temporal frequency band-pass, adding noise, flipping the axes and chopping samples into different patches. It is hard to establish a precise comparison between 1D and 2D, in terms of the number of parameters, data dimension, and estimation of

effective batch size and amount of epochs. For this reason, we have a limited number of epochs, that is 500, the learning rate is set at 0.001 and the batch size is 128. We have conducted several experiments to reach the best UNet result in current circumstances (like varying kernel size, number of passes for the encoding and decoding part, adaptive learning rate). The best result was still achieved with the default settings, proposed by Ronneberger *et al*. (2015). For the 1D CNN, we used 100,000 examples.

In Figure 7, we show the comparison of training curves between 1D CNN and 2D UNet models. One can see that for the same amount of epochs the 1D model reaches a lower limit of metrics than the UNet. Note that as we have shown in the previous section, due to the ambiguity of markup picks, we cannot judge the quality of training, relying solely on the training statistics. In Figure 5, one can see the examples of the industrial markup and areas of potential problems. In Figure 8, we show the picking results obtained by 1D CNN (on top) and UNet (at the bottom). The UNet provides a smoother mask and is not as well generalized as the 1D model. After the training, we have evaluated the UNet performance on a full dataset (4.5 million traces) and calculated the distribution of errors, where about 90% with no more than three sample shifts, 7% with no more than 10 samples and other 3% are above.

Thus, we did not find the UNet helpful for this particular dataset as it is not improving the fidelity of picking in the complicated areas and requires much more resources for training. The 1D CNN model provides about 96% of the solution (will be discussed in the next section) and much cheaper in computation: 2 million training parameters for 2D model versus 100 thousand for 1D; the number of traces required for training is equal to a batch size for 1D and 2D, it is bigger in 256 times (as training example is compiled of 256 traces). We think that such argumentation is reasonable enough to continue further tests only with the 1D architecture. Though we are not saying that UNet or another 2D architecture cannot be trained or tuned in a way that will provide a higher score. For example, following our results, the fully convolution 2D model can be considered to reduce the influence of 'spoiled' training picks and provide not an over-smoothed solution, which we see from UNet application.

Testing one-dimensional convolution neural network parameters

The strong variability of the first-arrival waveforms makes the problem of robust first-break picking difficult and labour intensive. This may require the involvement of the specialist for some manual data markup, but then it needs to be minimal. We also anticipate that in the future one may need to perform additional CNN training for some particular datasets. Thus during the CNN tuning, we were trying to achieve two main goals: to get the simplest CNN model and identify the minimal size of training set required for reliable first-break picking in such a way that it can be easily checked manually for reliability. The CNN was trained on a small fraction of *Dataset 1* (see correspondingly in Table 3) and then evaluated the remaining part of this dataset. Following this strategy, we performed a series of tests to optimize the following parameters:

- the number of hidden CNN layers ($k = 1, 2, \ldots, 7$);
- the size of the training dataset (5,000, 10,000, 25,000, 50,000, 100,000 traces).

The testing results for the four-layered CNN and different sizes of the training set are shown in Table 3. We estimate the performance of the CNN by comparing the original and predicted first-break picks as shown in Table 3 for the four-layered CNN and different sizes of the training set. The CNN accuracy is shown as a percentage of traces falling below the specific level of misfit between the original and predicted first-break picks: picks are equal (first row), the misfit is less or equal to three samples (second row), the misfit is more than three samples (third row). Columns show the results for different sizes of the training dataset. For further evaluation, we consider the misfit less or equal to three time samples (≤ 6 ms) as a reasonable accuracy of the first-break picking (this choice is motivated by discovered errors in the original first-break picks). From Table 3, we conclude that even for the smallest training dataset (5,000 traces) our CNN provides reasonable accuracy of 94.3% for the problem of the first-break picking.

We further made several tests for CNN with a different number of layers and using different sizes of the training dataset. Each time we evaluated the trained CNN on the full *Dataset 1* (about 4.5 million traces). In Table 4, we show the percentage of traces for which the misfit between the CNN predicted and the original picks was less or equal to three samples (≤ 6 ms). One can see that starting from four hidden layers the CNN accuracy remains almost the same being higher than 94%. Thus we suggest using the CNN with four hidden layers as an optimal architecture and recommend 5,000 traces as an optimal size of the training dataset. It should be noted that such a conclusion we have obtained for this particular dataset would have high fidelity only for similar datasets. As a general recommendation, we conclude that the proposed architecture is an effective baseline for the first-break picking problem but does not necessarily give the best result for some

Table 3 Dataset 1. Accuracy (percentage of the validation dataset) for the CNN model with four hidden layers trained on datasets of different sizes (columns) and the interval of misfit between original and predicted travel-times (rows)

	Size of Training Set					
Misfit between original and predicted first breaks (samples)	5,000	10,000	15,000	25,000	50,000	100,000
0	26.4	28.5	29.6	31.1	31.3	32.7
1	61.3	62.4	61.4	61.0	61.1	61.0
2	5.6	4.1	3.8	2.6	2.8	2.5
3	1.0	0.5	0.5	0.6	0.4	0.4
> 3	5.7	4.5	4.7	4.7	4.4	4.0
≤ 3	94.3	95.5	95.3	95.3	95.6	96.0

Table 4 Dataset 1. Accuracy (percentage of the validation dataset) of the CNN first-break picking (misfit between predicted and original travel-times ≤ 3 time samples) for different sizes of the training set and the number of CNN hidden layers

	Size of Training Set				
Number of CNN Layers	5,000	10,000	25,000	50,000	100,000
1	83.5	83.6	N/A	N/A	N/A
2	91.0	91.3	N/A	N/A	N/A
3	93.0	94.7	N/A	N/A	N/A
4	94.3	95.0	95.3	95.6	96.0
5	94.5	95.7	95.7	95.9	96.1
6	94.1	95.8	95.7	96.0	96.0
7	95.3	95.1	95.7	95.9	95.8

arbitrary dataset. In the case of this study, such a threshold is chosen because the root mean square (RMS) of tomography results is not changing whether we include or not the errors between 4 and 6 ms. On the other hand, the errors above 6 ms may lead to some minor changes in the tomography RMS, but no noticeable errors in the resultant velocity model to make a change in a misfit between observed and synthetic first-break travel times.

EVALUATION OF DATASETS

In this section, we first perform the visual comparison of the convolution neural network (CNN)-predicted and the original first-break travel times used for training. We mark these original first-break travel times as 'industrial' because they were obtained by conventional industrial processing. Our visual analysis revealed that the original first-break picks are not always correct and contain errors.

Thus we used an alternative way of checking the quality of the first-break picking – we used them for estimating the a priori static correction and further use them in the processing graph to get the final seismic stack. Then we can compare the resultant seismic stacks to illustrate the performance of the proposed CNN-based first-break picking. Note that for using the first-break travel times for further processing we propose the procedure for automatic post-processing of the CNN-predicted travel times for removing outliers. This procedure is also further described in this section.

In this section, we would like to show the comparison between the binary classification of the time sample (markup for two classes: first break and not first break) and three classes (noise, first break, signal). We have used the four-layered CNN model, trained on 5,000 examples under the same conditions, but with the different target markup (two and three classes output). In Figure 9, we show the comparison of picking results for the entire survey. The map is built per shot location, and the plotted attributes (errors and probability) are calculated as the median value per common-shot is gathered. There are about 280 receivers (corresponding first-break picks), and they were used to calculate a median value per shot location. The top row of the maps illustrates the probability value distribution (the predicted probability value of the first-break sample on seismic traces): (a) and (b); the bottom one is the mean squared error: (c) and (d). The left column is the results for the three-class markup: (a) and (c) and the right one is for for the binary markup. For both markup cases, we measure the probability of the first-break class. One can see that there is a mild difference between the error maps, but the probability distribution is more stable for the three-class markup. This observation motivated us to continue to use the three-class markup in our further tests. We have interpreted this result with higher confidence based on the CNN model, as shown by the proposed approach.

Comparison of the first-break picks

In Figure 10, we show the comparison of original (industrial) picks and our CNN-predicted results for *Dataset 1* (the CNN was trained on 5,000 traces from this dataset). One can see fractions of several common-shot gathers (not used for the CNN training). One can see that the quality of the CNN travel-time picking is overall comparable to the original (industrial) picking and in some cases it looks more accurate – see two lower panels in the figure. We also added the results of using classical first-break picking algorithms: modified energy ratio (MER) (Wong *et al*., 2009) and STA/LTA (Allen, 1978). For classical algorithms, we performed manual tuning of internal parameters (length of the sliding windows, detection threshold). In Figure 10, one can see that the results of these classical algorithms are unstable despite several rounds of parameter tuning.

In Figure 5, we show the picking results for the test dataset (*Dataset 2*), which was made in a 'blind' manner. The top picture in Figure 5 shows gathers with the presence of high variance in signal characteristics (frequency, decay, interference, etc.). One can see the overlapping of two waves on the offset of about 1,000 m. One wave has a higher velocity and lower frequency and the other one with lower velocity and higher frequency. The observed data can be interpreted as the shingling effect, where the high-speed velocity cannot be continuously observed through the offsets. At the bottom of Figure 5, we show the detection function built by the CNN output. One can see that the interval of the waves overlapping can be detected by the CNN, which provides more than a single focus of first-break probability. The orange curve shows the detected first-break times. According to the presented observations, we can propose that the CNN model can provide more than one first-break pick to be used for further specification by multi-trace analysis or tomography inversion.

Automatic post-processing for outlier removal

The tau-offset cross-plot is a standard tool for analysing the distribution of the first-break travel times and their QC. We suggest using it for automatic post-processing of the CNN-predicted travel times. For this, we divide all these travel times into 50-m bins in the offset coordinate. In Figure 11, we show histograms of the travel-time distributions for *Dataset 1*. We show the histograms for each offset bin – labels on the right show offset values for the bin centres – horizontal axis – first-break travel-time, vertical axis – the decimal logarithm of the number of travel-time picks. Histograms for the original (industrial) travel-time picks are shown in blue, and for CNN predicted travel-time picks are shown in orange. One can see that there are some outliers in the CNN-predicted travel times (orange histogram columns for tau values are greater than 300 ms). For each offset bin, we compute mean/median values and standard deviation values σ. In Figure 11, we show mean values and error bars (for σ, 2σ, 3σ and 4σ) by vertical ticks.

Motivated by Figure 11, we propose the following automatic approach to post-processing of the CNN-predicted first-break picks (a '3σ rule'):

- group picks from offset bins;
- compute the mean value and standard deviation σ in each bin;
- discard travel-time picks outside the 3σ interval around the mean.

In Figure 12, we show the cross-plot comparing the industrial (blue) and the CNN-predicted (orange) travel-time picks for *Dataset 1*. The top panel shows the CNN-predicted travel-time picks before our automatic post-processing. The bottom panel shows the CNN-predicted travel-time picks after our automatic post-processing. One can see that most of the obvious outliers have been removed.

For further evaluation, we used the CNN trained on the subset from *Dataset 1* to process *Dataset 2* as a 'blind' test, that is without additional training. In Figure 13, we show the cross-plot comparing the industrial (blue) and the CNN-predicted (orange) picks for *Dataset 2*. The CNN-predicted picks are shown after the proposed automatic post-processing. The 3σ interval for outlier removal is shown as the dashed line in the figure. Note that there were less than 1% of outliers in this case. Again we see that the CNN-predicted picks do not look more scattered for offsets over 600 m. This observation we consider as evidence of the stability of the proposed CNN model.

Comparison of seismic stacks

We use an alternative way of checking the quality of the first-break picking. We compare the seismic stacks calculated after a priori static corrections calculated using (1) original (industrial) and (2) the CNN-predicted first-break travel times. After the static corrections, the identical CDP processing is used to get the two seismic stacks.

In Figure 14, we show seismic stacks for *Dataset 1* obtained with a different a priori static correction estimated from the original (industrial) first-break travel-times (left) and the CNN-based travel-times (right). A similar comparison of seismic stacks for *Dataset 2* is shown in Figure 15. Both

examples show that the CNN-based stack is of superior quality compared to the original stack (processing with standard industrial software). White elliptic frames highlight regions with the most improvement of the CNN-based seismic image.

DISCUSSION

We present an interesting result that an imperfect training dataset obtained from automatic industrial processing procedures can be used for training the convolution neural network (CNN)-based picking algorithm. The CNN model was able to partially mitigate these errors and predict more stable first-break picks than the given industrial picks. We see three reasons, explaining this observation. First, the synthetic test proved that the CNN model is able to neglect the imperfectness of a markup (spoiled picks) and perform with an acceptable accuracy in case of a 'poor' consistency of training data. Second, the industrial picks are obtained as compromise of a fast and simple solution to get a reasonable first breaks for production work purposes. Finally, such robustness of the CNN models that applied for imperfect training datasets has been observed in other applications, for example automatic fault detection in seismic images (Li *et al.*, 2019). In this study, the robustness of the CNN model was confirmed by the comparison of seismic stacks obtained using a priori static corrections from the original (industrial) and the CNN-predicted first-break picks.

One can think of several directions for improving the CNN-model performance. In particular, one important result of this paper is that even a small marked dataset of about 5,000 traces is enough for training the CNN. In this case, one can think of manual correction of the travel-time picks in the training dataset to boost the CNN-model performance.

Another important question is a proper choice of the training dataset. The survey area may contain several distinct types of terrain or near-surface structure. Then it may be important to choose the training set carefully and preferably make this choice automatic. Let us consider our *Dataset 1* and two different cases for choosing the training subset (the size of the training subset is 5,000 traces as usual):

1. randomly chosen traces over the whole survey area;
2. close-by traces in a certain region of the survey.

In Figure 16(a), we show the elevation map of a survey in meters, where the black dots shows the location of traces chosen for a training subset in case 2). In Figure 16(b,c) , we provide maps of the distribution of mean error between the original and predicted first breaks for receiver points. The error is calculated in the steps of the time sample. Figure 16(b) shows the error distribution in the case of the random choice of training examples. Figure 16(c) shows the error distribution in the case of choosing the training subset only from the restricted area of the survey. One can see that for case 2 the distribution of errors strongly correlates with the peculiarities of the elevation. It can be noted that the greatest discrepancies arise in areas of low elevation, which corresponds to the floodplains of the rivers. Otherwise, the random choice of the training subset provides well-balanced distribution of the shot-gather conditions.

According to the results, in the first-break picking process, we suggest using a random distribution of examples for the cases of strong peculiarities of shot-gather conditions. According to the findings of this study, the strategy for first-break picking should consist of the following steps: choose the subset of seismic data (e.g. the set of common shot point (CSP) gathers), according to production criteria of data quality; provide initial picking with a conventional algorithm or pre-trained CNN; normalize each trace independently; train and evaluate the CNN model on the desirable subset; reject outliers; provide transfer learning (Pan and Yang, 2010) or fine-tuning and repeat evaluation.

CONCLUSIONS

We proposed a strategy for first-break picking in land seismic data with a relatively light one-dimensional convolution neural network (CNN) model. The set of synthetic tests was conducted to sustain the assumption of redundancy of fully connected layers in the presence of non-reliable markup. The three-class based markup was proposed as a strategy for balancing the class distribution. The comparison of the two-dimensional UNet model shows that the near-surface velocity features do not allow to account for the heterogeneity and lead to a smooth solution. It is shown that imperfect training datasets obtained from automatic industrial processing procedures can be used for training the CNN-based picking algorithm, which can be more robust than the conventional flow. Also, the CNN model provides stable estimates of the first-break travel times not only for small but also for large offsets with a low signal-to-noise ratio. Our results show that even small marked datasets of about 5,000 traces allow for training the CNN. Finally, we proposed an automatic procedure to discard outliers, based on a priori velocity information and basic statistical analysis.

We tested the proposed first-break picking on two real land seismic datasets acquired with an explosive source. The first dataset was used for developing architecture and

training the CNN. The second dataset was used for the 'blind' test of the pre-trained CNN. Both survey areas are characterized by a laterally heterogeneous near-surface structure. The total size of considered datasets contains over 7 million seismic traces. The error between original and predicted first breaks is not more than three samples for 95% of traces. The final QC of the first-break picking results was made possible by estimating static corrections and computing seismic stacks. After analysing seismic stacks, we conclude that the proposed algorithm provides comparable or better results than the original first-break picks. The proposed approach is more robust than the industrial picks for offsets over 600 m.

As further development of the proposed algorithm, we plan to improve the CNN model performance by introducing the wavelet-based initialization of the convolution kernel following the traditional convolution model of a seismic trace.

ACKNOWLEDGEMENTS

The authors thank ''Gazpromneft NTC LLC' for the opportunity to test the proposed approach on real data and publish the research results.

DATA AVAILABILITY STATEMENT

Research data are not shared.

ORCID

Georgy N. Loginov
https://orcid.org/0000-0002-4906-5986

References

Akazawa, T. (2004) A technique for automatic detection of onset time of p-and s-phases in strong motion records. In: *Proceedings of the 13th World Conference on Earthquake Engineering*, Vancouver, B.C., Canada. Canadian Association for Earthquake Engineering and International Association for Earthquake Engineering.

Akram, J. and Eaton, D. W. (2016) A review and appraisal of arrival-time picking methods for downhole microseismic data arrival-time picking methods. *Geophysics*, 81(2), KS71–KS91.

Akram, J., Ovcharenko, O. and Peter, D. (2017) A robust neural network-based approach for microseismic event detection. *SEG Technical Program Expanded Abstracts 2017*. SEG, p. 6093.

Allen, R. V. (1978) Automatic earthquake recognition and timing from single traces. *Bulletin of the Seismological Society of America*, 68(5), 1521–1532.

Ayub, M. and Kaka, S. I. (2021) A comparative analysis of machine learning models for first-break arrival picking. *Machine Learning*, 12(1), 493–502.

Boßmann, F. and Ma, J. (2015) Asymmetric chirplet transform for sparse representation of seismic data. *Geophysics*, 80(6), WD89–WD100.

Cova, D., Xie, P. and Trinh, P.-T. (2020) Automated first break picking with constrained pooling networks. In: *SEG International Exposition and Annual Meeting*. SEG, p. 3887.

Cui, H. and Bai, J. (2019) A new hyperparameters optimization method for convolutional neural networks. *Pattern Recognition Letters*, 125, 828–834.

Dai, H. and MacBeth, C. (1997) The application of back-propagation neural network to automatic picking seismic arrivals from single-component recordings. *Journal of Geophysical Research: Solid Earth*, 102(B7), 15105–15113.

Dong, X., Li, Y. and Yang, B. (2019) Desert low-frequency noise suppression by using adaptive DnCNNs based on the determination of high-order statistic. *Geophysical Journal International*, 219(2), 1281–1299.

Duan, X. and Zhang, J. (2019) Multi-trace and multi-attribute analysis for first-break picking with the support vector machine. In: *SEG Technical Program Expanded Abstracts 2019*. Society of Exploration Geophysicists, pp. 2559–2563.

Duan, X. and Zhang, J. (2020) Multitrace first-break picking using an integrated seismic and machine learning method. *Geophysics*, 85(4), WA269–WA277.

Dumoulin, V. and Visin, F. (2016) A guide to convolution arithmetic for deep learning [Preprint]. arXiv:1603.07285.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B. and Herrera, F. (2018) *Learning from Imbalanced Data Sets*. Springer.

Gentili, S. and Michelini, A. (2006) Automatic picking of P and S phases using a neural tree. *Journal of Seismology*, 10(1), 39–63.

Goodfellow, I., Bengio, Y., Courville, A. and Bengio, Y. (2016) *Deep Learning*. MIT Press.

Hagen, D. C. (1982) The application of principal components analysis to seismic data sets. *Geoexploration*, 20(1-2), 93–111.

Hashemi, H., Javaherian, A. and Babuska, R. (2008) A semi-supervised method to detect seismic random noise with fuzzy GK clustering. *Journal of Geophysics and Engineering*, 5(4), 457.

He, H. and Ma, Y. (2013) *Imbalanced Learning: Foundations, Algorithms, and Applications*. John Wiley & Sons.

He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Press, pp. 770–778.

Hollander, Y., Merouane, A. and Yilmaz, O. (2018) Using a deep convolutional neural network to enhance the accuracy of first-break picking. In: *2018 SEG International Exposition and Annual Meeting*. SEG, p. 5520.

Huang, W. (2019) Seismic signal recognition by unsupervised machine learning. *Geophysical Journal International*, 219(2), 1163–1180.

Ioffe, S. and Szegedy, C. (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift [Preprint]. arXiv:1502.03167.

Joswig, M. (1990) Pattern recognition for earthquake detection. *Bulletin of the Seismological Society of America*, 80(1), 170–186.

Kingma, D. P. and Ba, J. (2014) Adam: a method for stochastic optimization [Preprint]. arXiv:1412.6980.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012) Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.) *Advances in Neural Information Processing Systems 25* . Curran Associates, pp. 1097–1105.

Li, S., Yang, C., Sun, H. and Zhang, H. (2019) Seismic fault detection using an encoder–decoder convolutional neural network with a small training set. *Journal of Geophysics and Engineering*, 16, 175–189.

Liashchynskyi, P. and Liashchynskyi, P. (2019) Grid search, random search, genetic algorithm: a big comparison for NAS [Preprint]. arXiv:1912.06059.

Madureira, G. and Ruano, A. E. (2009) A neural network seismic detector. *IFAC Proceedings Volumes*, 42(19), 304–309.

Maity, D., Aminzadeh, F. and Karrenbach, M. (2014) Novel hybrid artificial neural network based autopicking workflow for passive seismic data. *Geophysical Prospecting*, 62(4), 834–847.

McCormack, M. D., Zaucha, D. E. and Dushek, D. W. (1993) First-break refraction event picking and seismic data trace editing using neural networks. *Geophysics*, 58(1), 67–78.

Mousa, W. A., Al-Shuhail, A. A. and Al-Lehyani, A. (2011) A new technique for first-arrival picking of refracted seismic data based on digital image segmentation. *Geophysics*, 76(5), V79–V89.

Mousavi, S. M., Horton, S. P., Langston, C. A. and Samei, B. (2016) Seismic features and automatic discrimination of deep and shallow induced-microearthquakes using neural network and logistic regression. *Geophysical Journal International*, 207(1), 29–46.

Murat, M. E. and Rudman, A. J. (1992) Automated first arrival picking: a neural network approach. *Geophysical Prospecting*, 40(6), 587–604.

Nalçakan, Y. and Ensari, T. (2018) Decision of neural networks hyperparameters with a population-based algorithm. In: *International Conference on Machine Learning, Optimization, and Data Science*. Springer. pp. 276–281.

Pan, S. J. and Yang, Q. (2010) A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359.

Peraldi, R. and Clement, A. (1972) Digital processing of refraction data study of first arrivals. *Geophysical Prospecting*, 20(3), 529–548.

Perol, T., Gharbi, M. and Denolle, M. (2018) Convolutional neural network for earthquake detection and location. *Science Advances*, 4(2), e1700578.

Qu, S., Guan, Z., Verschuur, E. and Chen, Y. (2019) Automatic high-resolution microseismic event detection via supervised machine learning. *Geophysical Journal International*, 222, 1881–1895.

Ronneberger, O., Fischer, P. and Brox, T. (2015) U-net: convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. pp. 234–241.

Sabbione, J. I. and Velis, D. (2010) Automatic first-breaks picking: New strategies and algorithms. *Geophysics*, 75(4), V67–V76.

Sleeman, R. and Van Eck, T. (1999) Robust automatic P-phase picking: an on-line implementation in the analysis of broadband seismogram recordings. *Physics of the Earth and Planetary Interiors*, 113(1-4), 265–275.

Sun, M., Zhang, J. and Wang, Y. (2018) Recognizing shingling seismic data by unsupervised machine learning. In *SEG Technical Program Expanded Abstracts 2018*. Society of Exploration Geophysicists, pp. 2561–2565.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1–9.

Tan, Y. and He, C. (2016) Improved methods for detection and arrival picking of microseismic events with low signal-to-noise ratios. *Geophysics*, 81(2), KS93–KS111.

Tsai, K. C., Hu, W., Wu, X., Chen, J. and Han, Z. (2019) Automatic first arrival picking via deep learning with human interactive learning. *IEEE Transactions on Geoscience and Remote Sensing*, 58(2), 1380–1391.

Tselentis, G.-A., Martakis, N., Paraskevopoulos, P., Lois, A. and Sokos, E. (2012) Strategy for automated analysis of passive microseismic data based on S-transform, Otsu's thresholding, and higher order statistics. *Geophysics*, 77(6), KS43–KS54.

Turhan Taner, M., Lu, L. and Baysal, E. (1988) Unified method for 2-D and 3-D refraction statics with first break picking by supervised learning. In *SEG Technical Program Expanded Abstracts 1988*. Society of Exploration Geophysicists, pp. 772–774.

Ursin, B. and Zheng, Y. (1985) Identification of seismic reflections using singular value decomposition. *Geophysical Prospecting*, 33(6), 773–799.

Van der Baan, M. and Jutten, C. (2000) Neural networks in geophysical applications. *Geophysics*, 65(4), 1032–1047.

Wang, J. and Teng, T.-L. (1995) Artificial neural network-based seismic detector. *Bulletin of the Seismological Society of America*, 85(1), 308–319.

Wong, J., Han, L., Bancroft, J. and Stewart, R. (2009) Automatic time-picking of first arrivals on noisy microseismic data. *CSEG*, 1(1.2), 1–4.

Wu, H., Zhang, B., Li, F. and Liu, N. (2019) Semiautomatic first-arrival picking of microseismic events by using the pixel-wise convolutional image segmentation method. *Geophysics*, 84(3), V143–V155.

Xie, T., Zhao, Y., Jiao, X., Sang, W. and Yuan, S. (2019) First-break automatic picking with fully convolutional networks and transfer learning. In: *SEG Technical Program Expanded Abstracts 2019*. Society of Exploration Geophysicists, pp. 4972–4976.

Xu, Y., Yin, C., Pan, Y., Ni, Y., Zou, X. and Yang, T. (2021) First-break automatic picking technology based on semantic segmentation. *Geophysical Prospecting*, 69(6), 1181–1207.

Yaskevich, S., Loginov, G., Duchkov, A. and Serdukov, A. (2016) Pitfalls of microseismic data inversion in the case of strong anisotropy. *Applied Geophysics*, 13(2), 326–332.

Yilmaz, Öz. (2001) *Seismic Data Analysis: Processing, Inversion, and Interpretation of Seismic Data*. Society of Exploration Geophysicists.

Yu, S., Ma, J. and Wang, W. (2019) Deep learning for denoising. *Geophysics*, 84(6), 1–107.

Zhang, K., Zuo, W., Chen, Y., Meng, D. and Zhang, L. (2017) Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7), 3142–3155.

Zhao, Y. and Takano, K. (1999) An artificial neural network approach for broadband seismic phase picking. *Bulletin of the Seismological Society of America*, 89(3), 670–680.

Zheng, J., Lu, J., Peng, S. and Jiang, T. (2017) An automatic microseismic or acoustic emission arrival identification scheme with deep recurrent neural networks. *Geophysical Journal International*, 212(2), 1389–1397.

Zhu, W. and Beroza, G. C. (2018) Phasenet: a deep-neural-network-based seismic arrival time picking method [Preprint]. arXiv:1803.03211.

Zwartjes, P., Fernhout, M. and Yoo, J. (2020) Evaluation of neural network architectures for first break picking. In *82nd EAGE Annual Conference & Exhibition*, volume 2020. European Association of Geoscientists & Engineers, pp. 1–5.