RESEARCH-ARTICLE

SeismicSense: Phase Picking of Seismic Events with Embedded Machine Learning

TAYYABA ZAINAB, GEOMAR Helmholtz Centre for Ocean Research Kiel, Kiel, Schleswig-Holstein, Germany

LAURA HARMS, University of Kiel, Kiel, Schleswig-Holstein, Germany

JENS KARSTENS, GEOMAR Helmholtz Centre for Ocean Research Kiel, Kiel, Schleswig-Holstein, Germany

OLAF LANDSIEDEL, University of Kiel, Kiel, Schleswig-Holstein, Germany

Open Access Support provided by:

University of Kiel

GEOMAR Helmholtz Centre for Ocean Research Kiel

Published: 31 March 2025

Citation in BibTeX format

SAC '25: 40th ACM/SIGAPP Symposium on Applied Computing
March 31 - April 4, 2025
Catania, Italy

Conference Sponsors:
SIGAPP

SeismicSense: Phase Picking of Seismic Events with Embedded Machine Learning

Tayyaba Zainab[1,2], Laura Harms[2], Jens Karstens[1], Olaf Landsiedel[2]
[1]*GEOMAR Helmholtz Centre for Ocean Research Kiel*, Germany
[2]*Kiel University*, Germany
tzainab@geomar.de, laura.harms@cs.uni-kiel.de, jkarstens@geomar.de, olaf.landsiedel@cs.uni-kiel.de

Abstract

Analyzing seismic data is essential for understanding natural geological processes and anthropogenic activities, particularly in localizing seismic events. While recent advances in seismic analysis rely heavily on resource-intensive machine learning approaches, these methods are impractical in resource-constrained environments such as underwater, underground, or rural areas. To address this, we introduce SeismicSense, a lightweight neural network (NN)-based solution for sensor-level seismic data analysis. SeismicSense detects seismic events and localizes them by identifying seismic event phases through a cascading architecture. Initially, SeismicSense uses an NN to filter out non-earthquake events, minimizing false positives. Upon identifying an earthquake, it detects the P- and S-phases, which are crucial for determining the origin and magnitude of seismic activity. SeismicSense significantly reduces data transmission by communicating only the arrival times of these phases to the cloud, enabling efficient and selective communication during seismic events. Despite being 20 times smaller than state-of-the-art models and requiring just 186 KB of RAM, SeismicSense achieves exceptional performance, with F1-scores of 99.4% for earthquake detection, 98% for P-wave detection, and 96% for S-wave detection. Additionally, leveraging integer acceleration on modern MCUs enhances efficiency, reducing inference time on Cortex-M MCUs by 18-fold compared to non-accelerated methods, enabling real-time execution.

CCS Concepts

• **Computer systems organization** → **Embedded software**; *Real-time system architecture*; **Sensor networks**; • **Computing methodologies** → **Neural networks**.

Keywords

Seismological data analysis, Earthquake detection, TinyML, Deep Neural Networks, Low-Power, Internet of Things, Edge AI

ACM Reference Format:
Tayyaba Zainab, Laura Harms, Jens Karstens, Olaf Landsiedel. 2025. SeismicSense: Phase Picking of Seismic Events with Embedded Machine Learning. In *The 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25), March 31-April 4, 2025, Catania, Italy*. ACM, New York, NY, USA, Article 4, 9 pages. https://doi.org/10.1145/3672608.3707845

SAC '25, March 31-April 4, 2025, Catania, Italy

ACM ISBN 979-8-4007-0629-5/25/03
https://doi.org/10.1145/3672608.3707845

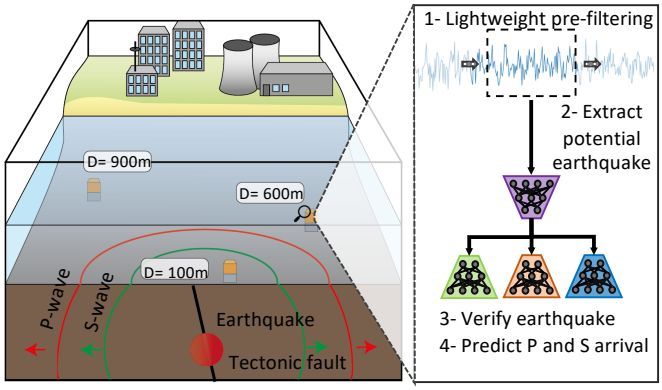


Figure 1: Design overview of SeismicSense, our system for earthquake analysis: We utilize a lightweight pre-filter to identify potential seismic events in a continuous data feed originating from seismic sensors. Upon identification, we forward a data window of 60 sec to our neural network for validation and classification of primary (P) and secondary (S) seismic waves.

1 Introduction

Seismic event analysis plays a pivotal role in comprehending geological processes, monitoring tectonic and volcanic hazards, as well as human activities such as oil and gas exploration or underground CO2 storage [5]. Traditionally, seismic data is analyzed centrally, requiring the raw data to be transferred from the sensor to a server, where analytical and machine learning-based analyses extract relevant information from a continuous data stream. However, transmitting vast amounts of raw data collected every millisecond is impractical when systems are deployed in constrained environments (i.e. underwater) with limited communication bandwidth of only a few hundred bits per second [9]. In such environments, performing data analysis directly at the sensor level allows us to process the data on-site, significantly reducing the amount of data that needs to be transmitted.

Recent advances in neural network (NN)-based seismic signal analysis through state-of-the-art (SOTA) models such as PickNet [22], Phasenet [28], EQTransformer [13], and LCANet [27] outperform traditional methods [12, 19]. However, their large model size and resource requirements prohibit deployment on embedded systems, i.e. MCUs. In contrast, LightEQ [26] has demonstrated strong performance for event detection on MCUs but falls short in addressing the more challenging task of identifying seismic phases, which are necessary for localizing seismic events.

In this paper, we introduce SeismicSense[1], a lightweight NN model to detect earthquakes and identify primary (P) and secondary (S) wave arrivals (hereafter P- and S-picks) in seismic data (Figure 1). Due to its lightweight design, SeismicSense efficiently processes data directly on the sensor without the need of offloading raw data to remote servers. Subsequently, the predicted P- and S-picks provide the necessary information to accurately determine the distance of the seismic source. Triangulating the distances between P- and S-wave arrival times determined by SeismicSense from multiple sensors, an edge or cloud device can subsequently determine the location of an earthquake [6]. As energy conservation is critical in constrained environments, continuously processing raw data (which is mostly noise, as seismic events are rare) using NN is impractical. Therefore, similar to SOTA methods, we employ STA/LTA (Short-Term Average/Long-Term Average) as a pre-filter for seismic event detection and noise filtering [10, 11, 26]. This approach ensures efficient detection of potential seismic events while minimizing unnecessary NN usage and conserving resources.

To further conserve energy, we divide our neural network into four sub-models: an encoder and three decoders, namely the D (detection), P-pick, and S-pick decoder. If the STA/LTA pre-filter flags an input window as a potential earthquake, we process it through the encoder, producing an intermediate signal representation. The D decoder then classifies the input as either an earthquake or noise (false positive). Only if classified as an earthquake, we run the P-pick and S-pick decoders to identify the P and S wave phases. After identifying the phases, SeismicSense reduces the continuous raw data, originally recorded every millisecond, to only the P and S phase arrival times, which are then transmitted to the cloud.

SeismicSense achieves an F1-score of 99.4% for detection, 98% and 96% for P- and S-wave picking on MCUs for the STEAD dataset [14], requiring no more than 189 KB of RAM. SOTA models such as EQTransformer and LCANet achieve F1 scores of 100% for detection, 99% for P-phase, and 98% for S-phase picking but require at least 20 times more RAM, making them impractical for common embedded MCUs. Additionally, leveraging the integer accelerator on MCUs reduces inference latency 18-fold on Cortex-M MCUs compared to float inference. For example, on a Cortex-M7, SeismicSense's inference time for processing a minute of seismic data drops from 10.118 seconds to 0.565 seconds.

Contributions. This paper contributes the following:

(1) We introduce SeismicSense, a lightweight NN model for seismic event detection and phase picking.
(2) We segment SeismicSense into four sub-models: an encoder, and three decoders dedicated to event detection, P-phase detection, and S-phase detection, respectively. All decoders leverage the output of the encoder, and we only execute the later two decoders if the detection decoder identifies an earthquake. This cascading optimization enables on-demand execution of only the necessary sub-models, reducing both execution time and energy consumption.
(3) We show that our model, on an MCU, has a compact memory footprint of 189 KB RAM and detects seismic events with 99.4% F1-score for detection and 98% and 96% for P- and S-phase detection respectively on the STEAD dataset.
(4) We achieve an 18-fold reduction in inference latency on Cortex-M MCUs with integer acceleration compared to float inference without acceleration.

[1]Implementation available at: https://github.com/ds-kiel/SeismicSense

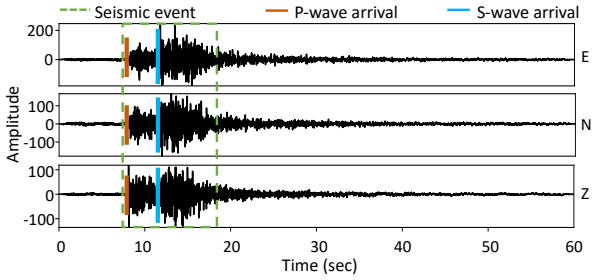


Figure 2: Three-component seismic data: The green dashed box represents the seismic event, while the orange vertical line denotes the P-wave arrival, and the blue line signifies the S-wave arrival.

Outline. We provide background on seismic data and STA/LTA in §2 and summarize the related literature in §3. This is followed by §4 which discusses the challenges for on-device seismic event analysis, SeismicSense model and implementation details. In §5, we evaluate SeismicSense on MCUs, and in §6, we present our conclusions.

2 Background

This section introduces the necessary background on seismic data and the STA/LTA method, commonly used as an anomaly detection technique in seismic event analysis.

2.1 Seismic data

Tectonic and volcanic activity, as well as anthropogenic subsurface activities, cause seismic waves to propagate through the Earth, which can be measured by seismic sensors (e.g. seismometers). Earthquakes and other seismic events produce surface and body waves, which consist of compressional and shear waves. Compressional waves travel faster than shear waves and arrive at a seismic station earlier. Therefore, the compressional wave is also called the primary wave (P-wave), while the later-arriving shear waves are also called secondary waves (S-waves).

The superposition of both waves defines the total seismic signals recorded by seismometers during the seismic event. The temporal gap between P- and S-wave arrivals allows for determining the distance between the seismic signal source (epicenter) and the seismic station. Calculating this distance for three or more stations allows the epicentre to be located [6]. Seismometers used for earthquake monitoring record data on three components oriented along three perpendicular axes: (1) the Z-component, which captures vertical motion, (2) the N-component, which captures horizontal motion in north-south direction, and (3) the E-component, which captures horizontal motion in east-west direction [1], see Figure 2.

2.2 STA/LTA

STA/LTA (Short-Term Average/Long-Term Average) is an anomaly detection technique for time-series seismic data. It computes the

average amplitude of a seismic signal over two consecutive moving time-windows: the STA window, is sensitive to seismic events, while the LTA window provides information about the amplitude level of seismic noise at the site. If the ratio between STA and LTA exceeds a particular threshold, it triggers an alert for a potential earthquake. This threshold determines the sensitivity of STA/LTA. This method is widely used in seismology due to its simplicity and efficiency in identifying sudden changes in seismic signals, making it a robust first step in automated earthquake detection pipelines. We, just like others, use STA/LTA as a pre-filter to detect potential earthquakes in the data stream [10, 11, 26].

3 Related work

Classic techniques for anomaly detection, such as STA/LTA [20, 25], Akaike Information Criterion (AR-AIC) [12], and Kurtosis [19], identify seismic events by detecting abrupt changes in the time-series. However, in recent years, traditional methods have been gradually replaced by data-driven approaches. PhaseNet [28], PickNet [22] and others [4, 16–18, 24] employ deep convolutional neural networks (CNNs) for seismic phase-picking. Other models, such as CRED [15] combine CNNs with recurrent neural networks (RNNs). CRED uses CNN layers for feature extraction and followed by both uni-directional and bidirectional long short-term memory (LSTM) layers to capture sequential patterns in seismic data. EQTransformer [13] employs attention-based [21] deep learning for earthquake detection and phase identification in cloud environments. Taking inspiration from SqueezeNet [7], a parameter-efficient image classifier, LCANet [27] focuses on edge devices and employs context-aware attention networks for earthquake detection and phase identification. Overall, these SOTA models achieve an earthquake detection accuracy of up to 100% and phase identification of about 99%. However, their deployment is limited to high-capacity computational devices like edge or cloud servers and thus requires the transmission of all raw data to the edge or cloud.

Recently, inexpensive and energy-efficient MCUs with Digital Signal Processing (DSP)-based integer acceleration have made on-site data processing feasible using small NN models. In the domain of earthquake detection on MCUs, LightEQ [26] utilizes CNNs and LSTMs and achieves high accuracy in detecting the occurrence of seismic events. However, its accuracy in identifying the precise location of the event within the data is insufficient for phase picking, which requires reliable detection of seismic phase onsets. For example, when detecting P and S arrivals on 100 random samples, LightEQ demonstrates an onset error with a mean of 10.9 seconds and a standard deviation (std) of 13.9 seconds. The offset error exhibits a mean of 19.3 seconds with a std of 13.7 seconds. In contrast, SOTA achieves a mean error of approximately 0 seconds and a std of less than 0.5 seconds.

This paper builds on the concept introduced by LightEQ of bringing efficient seismic data processing to MCUs and addresses its limitations by enabling both accurate earthquake detection and precise phase identification on resource-constrained MCUs, thereby facilitating robust epicenter localization.

4 Design

In this section, we start by discussing the challenges and then delve into each component of our system design pipeline, which comprises a six-step process, as depicted in Figure 3. We also provide implementation details related to the target embedded devices, quantization, and tailored NN layers for MCUs.

4.1 Challenges

On-device monitoring of seismic events using lightweight neural networks faces three key challenges:

Limitations of detection models for phase picking: While seismic event detection models like LightEQ excel at identifying events, they are not suitable for phase picking as this task is more complex and necessitates dedicated neural networks.

Matching state-of-the-art phase picking accuracy with limited resources: Achieving phase-picking accuracy comparable to SOTA models using transformers and attention mechanisms within MCU resource constraints is challenging. However, we prioritize enabling phase picking directly on the MCU to reduce data transmission significantly, even if it involves a slight compromise in P- and S-phase precision (up to 0.5 seconds deviation, as discussed in § 5.2). These minor inaccuracies can be mitigated by deploying additional sensors to improve event triangulation.

Real-time execution constraints: Achieving resource efficiency for prolonged operational lifespan while attaining real-time processing speeds is a significant challenge. This requires, for instance, processing 1 minute of seismic data within 1 minute, to ensure readiness for the next incoming data window without delay.

4.2 Signal Pre-Filtering

Earthquakes are infrequent occurrences and constitute a minor portion of the incoming data. The process of conducting NN inference on continuously generated data poses resource-intensive challenges, particularly when operating on devices with limited battery life. To tackle this concern, we use STA/LTA as a lightweight threshold filter. This reduces the need for NN-based classification to only potential earthquakes identified by the STA/LTA filter, rather than analyzing all incoming data. STA/LTA sensitivity can be configured through a threshold value. We configure STA/LTA with a 6-second STA window and a 22.5-second LTA window. This setup allows us to detect all seismic events (100% detection rate) while maintaining an accuracy of 84% in correctly identifying actual seismic events at a threshold value of 1.8. As this is a well-established technique for sensor deployment, our focus in this paper is on performing NN inference on MCUs and using STA/LTA as a pre-filter.

4.3 Neural Network Model

We conduct a hyperparameter search to achieve the highest accuracy while ensuring a deployable model size for embedded devices, resulting in a NN model shown in Figure 4. This model architecture is based on EQTransformer [13]. The input to this NN consists of sampled time-series data with dimensions of 6000x3, typically representing 60 seconds of recording flagged as potential earthquake by the STA/LTA pre-filter. The NN commences with the encoder block consisting of 1D convolutional layers with varying filters and kernel sizes. Each Conv1D layer is followed by a max pooling layer

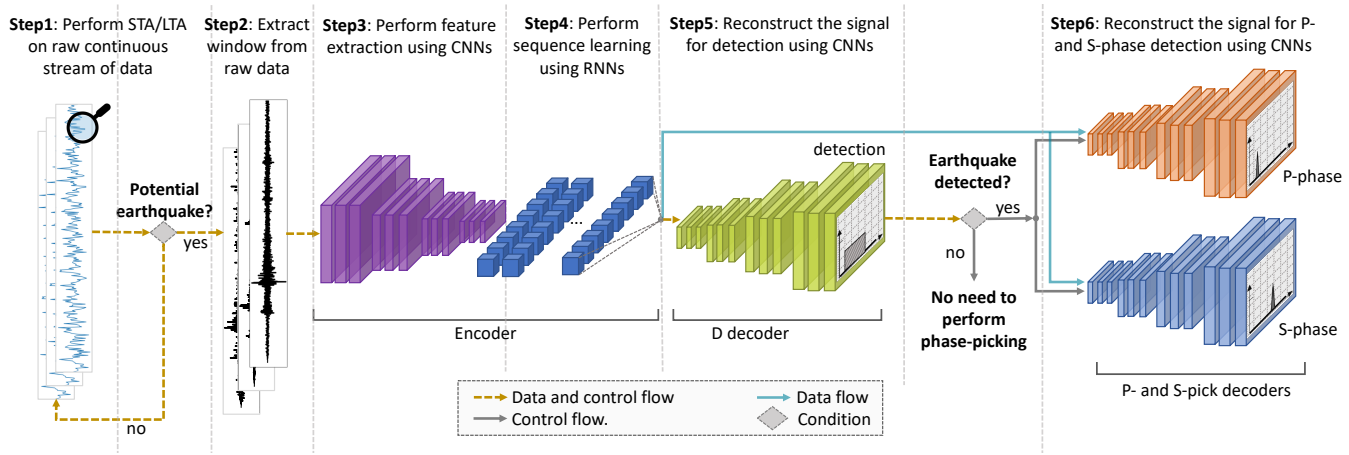


Figure 3: Overview of SeismicSense: STA/LTA performs on-device identification of potential earthquakes on a continuous stream of raw seismic data. Once a frame is identified as a potential earthquake by the pre-filter, we forward this window to our NN, which consists of four sub-models: 1) An encoder block for performing feature extraction and sequence learning, and three decoder blocks. 2) The output from the encoder is sent to the detection (D) decoder to verify the detection of the earthquake signal in the input. 3) Only on detection, we use the output from the encoder to perform P- and S-phase picking, enabling event localization.

with a pool size of 2 to reduce the size of the encoder states. Next, we have seven Residual CNN (ResCNN) blocks. Each ResCNN block has batch-normalization, dropout, Conv1D layers, and a skip connection. For sequence learning, we incorporate three BiLSTM (bidirectional LSTM) blocks, where each block contains a bidirectional LSTM layer, Conv1D, and batch-normalization. Finally, the last layer of the encoder is an LSTM followed by batch-normalization.

The output of the encoder is an embedding of size 47x64, which serves as input for the three decoders: detection (D), P-pick and S-pick. All the decoders have a similar structure: they employ seven Conv1D layers, where each layer is preceded by an upsampling layer. The last layer of each decoder is a sigmoid layer. Each decoder's output is a probability vector with a size of 6000x1.

4.4 Post processing

The vector output of the detection (D) decoder represents the probability of whether there is an earthquake or not at each point in time. This probability vector is compared to an upper- and lower-threshold value to count the total number of seismic events in a time window.

Phase detection requires a dedicated neural network due to its complexity, but even then, it can identify multiple incorrect P- and S-phases for a single event. To address this, we use the probability vector from the detection block to filter out picks that fall outside the detection curve, significantly improving the overall accuracy of SeismicSense's results.

4.5 Target devices

To evaluate SeismicSense, we use the SoCs nRF52840 (nRF52), nRF5340 (nRF53), and STM32H747i (STM32). nRF52 is a single-core ARM Cortex-M4 with FPU clocked at 64 MHz, 256 KB RAM, and 1 MByte flash. The nRF53 features a dual ARM Cortex-M33 with FPU, which can be clocked at either 128 MHz or 64 MHz; in our setup, we use the 128 MHz configuration. It also has 512 KB of RAM and 1 MB of flash memory. The STM32H747i is a dual-core MCU that offers the performance of the ARM Cortex-M7 core running at up to 480 MHz and the Cortex-M4 running at 240 MHz. It is equipped with 2 MBytes of flash memory and 1 MByte of RAM. We use the Cortex-M7 of the STM32H747i for our evaluation. All three selected MCUs use DSP instructions with SIMD (Single Instruction, Multiple Data) capabilities.

4.6 Quantization

Integer quantization of a NN [2] entails converting the model's weights and activations from high-precision floating-point formats to integers. To achieve compatibility with ARM Cortex's DSP, we quantize our 32-bit floating-point models into 8-bit fixed-point integer models with float fallback, i.e. if a quantized implementation is unavailable for a specific operation, we use higher bit representation as an alternative method for that specific layer only. Applying post-quantization on a NN model typically results in a reduction of model size by a factor of 4. However, it may also lead to a decrease in model accuracy [8]. Our evaluation shows that the accuracy drop for P-phase detection is 1% and for S-phase detection is 2%. We refer to the floating-point version of SeismicSense as SeismicSense-float and interchangeably use SeismicSense and SeismicSense-int for the quantized integer version.

4.7 Implementation

We implement SeismicSense using TensorFlow and apply quantization to leverage the SIMD capabilities available on our target devices, utilizing Zephyr RTOS and TensorFlow Lite for Microcontrollers (TFLite Micro) [3].

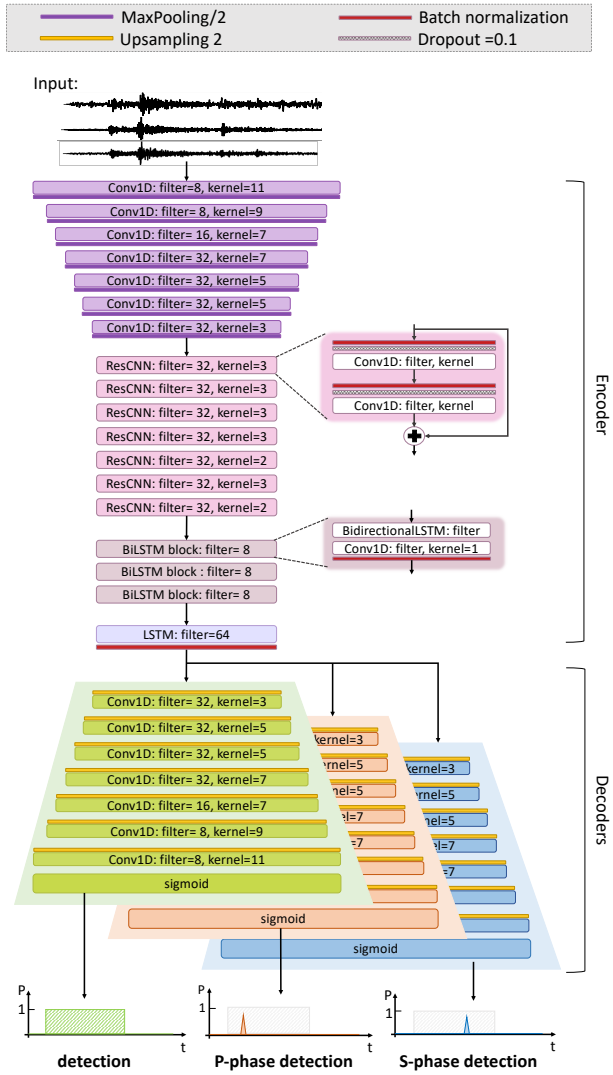


Figure 4: NN architecture: The encoder block of SeismicSense commences with CNNs and max-pooling, followed by CNN residual blocks, a sequence of bidirectional and CNN layers, and concludes with the addition of an LSTM layer. SeismicSense has three identical decoder blocks for detection, P-phase and S-phase detection. Each decoder is composed of upsampling layers, CNN layers, and a final sigmoid layer.

TFLite Micro supported operators constitute a subset of the TensorFlow core library operators, meaning that not all operations available in TensorFlow models are supported in TFLite Micro. While the majority of SeismicSense operators are supported, a few are not. Out of all the layers in SeismicSense, bi-dimensional LSTMs and UpSampling1D layers are currently not supported in TFLite Micro.

To address this limitation, we design and implement these layers. A common approach for implementing bidirectional LSTMs is to employ two LSTMs stacked on top of each other. The input to the second LSTM is provided in reverse order, typically achieved through LSTM configurations. However, LSTM configuration to reverse the input is also not supported in TFLite Micro. Therefore, we extend our implementation to reverse the order of the input manually and then feed it to the second LSTM. We also added a custom layer for upsampling, which increases the temporal resolution of a 1D input tensor by duplicating the input values along the time steps by a defined factor. The algorithm operates on batches of input tensors and efficiently reshapes and scales them for upsampling. We have consciously avoided using operations such as "tile" or "repeat" in our NN architecture, as they are also not supported in TFLite Micro.

5 Evaluation

In this section, we evaluate our model by comparing it to SOTA models in terms of precision, recall, and F1 scores. Additionally, we analyze the resource requirements of the complete SeismicSense model as well as its individual sub-models on our target devices. Furthermore, we quantify the energy consumption and inference latency of our models on MCUs, both with and without DSP support.

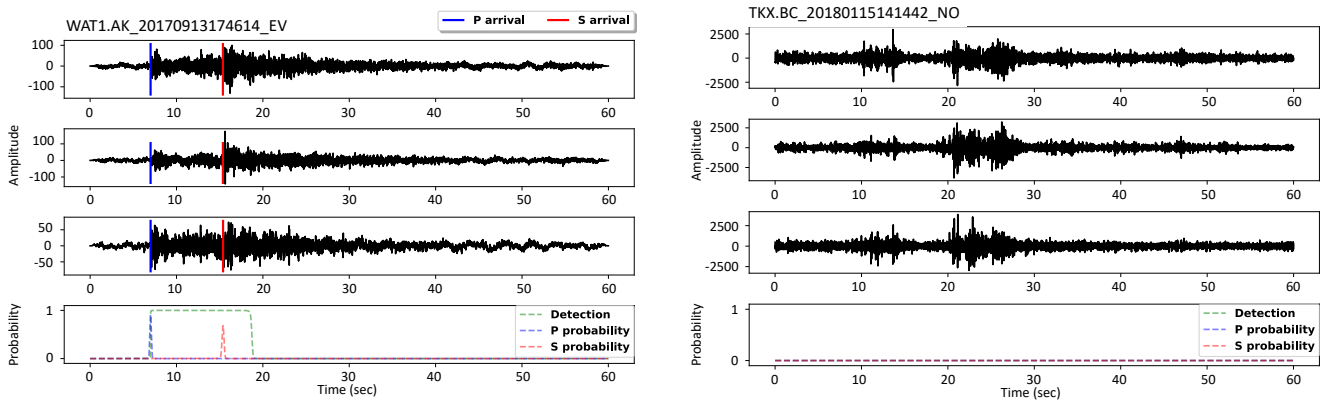
5.1 Datasets

We train SeismicSense on the STEAD [14] dataset. It is a vast, global dataset that contains 1,030,231 earthquakes and 235,426 noise waveforms. These waveforms are recorded by 2,613 seismometers located at local distances within 350 km of the epicenters. However, most of the recordings are localized within 100 km from the epicenter. The dataset encompasses earthquakes ranging from magnitude -0.5 to magnitude 7.9, with a predominant representation of smaller earthquakes (magnitudes < 2.5). Although our pipeline works on a moving window, we adhere to the common practice in SOTA approaches by using one-minute windows for each waveform for a fair comparison with our baselines. Data is sampled at 100 Hz, constituting to 6000 samples per channel for a window of one minute. Similar to EQTransformer, we augment the training dataset (STEAD) to enhance the diversity and robustness of training and validation data.

We additionally test our model on the Iquique dataset [23], which unlike STEAD is an offshore dataset. It contains 13,400 examples of seismic events from the aftershock sequence following the magnitude 8.1 Iquique earthquake occurring in Northern Chile in 2014. We have a total of 407 unique aftershock events, with each event recorded at multiple stations. We select events that have proper P- and S-labels, resulting in 11,288 events, which we use for our evaluation. All stations are sampled at 100 Hz, and collect 3-component data. Each waveform has a variable length, typically around 2 minutes. We downsample the data to 50 Hz and truncate the end of each waveform, resulting in waveforms with dimensions of 6000x3, making the input manageable on embedded devices. Our evaluation demonstrates that despite this reduced input dimension, we still achieve accuracy on par with SOTA model.

5.2 Training and Metrics

We randomly split the STEAD dataset into training, validation, and test sets, with proportions of 60%, 20%, and 20%, respectively. We employ binary cross-entropy as our loss function, accounting for



(a) Earthquake signal **(b) Noise signal.**

Figure 5: Example results for SeismicSense on the STEAD test set: both sub-figures consists of four plots. The initial three plots illustrate the input data corresponding to the East, North, and Z components. The fourth plot presents the aggregate outputs generated by SeismicSense: (a) Earthquake magnitude of 0.8, a seismic source distance of 57.45 km, an SNR of 17.5, and P- and S-wave arrival at 7.00 and 15.37 sec respectively. SeismicSense has detected the P- and S-arrival at 7.00 and 15.39 sec, respectively. (b) Noise signal. Our model shows promising capabilities in distinguishing between noise waveforms and earthquakes, even when they exhibit similar wave patterns.

the possibility of multiple seismic events occurring within a single input. The loss weights are set at 0.2 for the D decoder, 0.3 for the P-pick decoder, and 0.5 for the S-pick decoder. The weight for the S-pick decoder is higher due to the increased difficulty of identifying S-picks, particularly because of tailing coda effects. We use the Adam optimization algorithm to train the complete model initially, after which we split it into sub-models for cascading execution.

For the Iquique dataset, we partition the data into 40% training, 10% validation and 50% testing sets. Notably, multiple recordings of the same event are exclusively present in either the training, validation, or test set, ensuring their independence from each other.

Detection: The output of our D decoder is a probability vector. We use upper- and lower-bound thresholds (tr) in order to actively count the number of earthquake events for binary classification. SeismicSense uses threshold values of 0.5 for the upper-bound and 0.25 for the lower-bound. Finally, we use precision, recall, and the F1 score to evaluate the performance of our detection model. Precision (Pr) is the correctness of the model when it detects an earthquake, recall (Re) is the fraction of true positives correctly detected, and the F1 score is the harmonic mean of precision and recall.

P- and S-pick: Similar to the evaluation of D decoder, we use the same metrics to assess the performance of the P- and S-pick decoders. Inline with EQTransformer [13], we define a pick as true positive (TP) that has an absolute temporal deviation from the ground truth of less than 0.5 seconds. Additionally, we incorporate mean error (μ) and standard deviation of error (μ) metrics to provide a comprehensive evaluation of picker performance.

5.3 Exemplary Results

Before comparing SeismicSense to SOTA baselines, we present selected exemplary results. This serves to illustrate the complexity of the task, particularly when there is similarity between noise and earthquake data. Figure 5 shows the results of SeismicSense on two distinct events from the STEAD dataset: one representing an earthquake and the other representing noise. Each sub-figure consists of four plots. The first three plots of each sub-figure represent the three input channels as mentioned in Section 2. Earthquake waveforms are marked with blue and red markers, denoting P- and S-arrival ground truth, respectively. The fourth plot displays the combined output of SeismicSense for D, P- and S-pick decoders. Comparing Figure 5a with Figure 5b, seismograms of both examples exhibit minor amplitude disturbances from background noise followed by stronger peaks resembling earthquake-like impulses, our model effectively distinguishes between the two instances and correctly classifies them as earthquake and noise, respectively.

5.4 Comparison with SOTA

In this section, we compare our NN approach with a standalone STA/LTA with default configuration, and data-driven models such as CRED [15], LightEQ [26], PhaseNet [28], PickNet [22], EQTransformer [13], and LCANet [27]. We evaluate these standalone models in their original configurations, as quantizing them to run efficiently on MCU is not feasible.

Table 1 compares our float and integer quantized model with SOTA models in terms of their characteristics and resource demands. Our SeismicSense-float is not deployable on target MCUs. Our primary competitors for SeismicSense-int on MCUs are STA/LTA and LightEQ, both of which focus solely on detection and do not tackle the more complex task of phase identification. However, when comparing SeismicSense with non-MCU deployable models, our model outperforms CRED, PhaseNet, and PickNet. While SOTA models such as EQTransformer and LCANet slightly outperform SeismicSense-int in phase picking—achieving P-phase F1 scores that are 0.01 percentage points higher and S-phase F1 scores that are 0.02 percentage points higher than SeismicSense (see Figure 6a)—this difference is not significant and can be mitigated by deploying

Table 1: Summary for the SOTA models compared to SeismicSense, regarding model characteristics, resource demands and possible target devices. STA/LTA, CRED, and LightEQ can only perform detection, while Phasenet, PickNet, EQTransformer, LCANet, and SeismicSense are designed for phase-picking. However, only STA/LTA, LightEQ and SeismicSense are deployable on MCU.

Model	Model size	task		Detection			P-phase					S-phase					DNN	MCU
		det.	ph.	Pr	Re	F1	μ	std	Pr	Re	F1	μ	std	Pr	Re	F1		
STA/LTA	-	✓		0.88	0.99	0.93	-	-	-	-	-	-	-	-	-	-	-	✓
CRED	3.7MB	✓		1.0	0.96	0.98	-	-	-	-	-	-	-	-	-	-	RNN	
LightEQ	39KB	✓		0.97	0.99	0.98	-	-	-	-	-	-	-	-	-	-	RNN	✓
Phasenet	3.2MB		✓	-	-	-	0.0	0.08	0.96	0.96	0.96	0.0	0.11	0.96	0.93	0.94	CNN	
Picknet	72MB		✓	-	-	-	0.0	0.09	0.81	0.49	0.61	0.0	0.17	0.75	0.75	0.75	CNN	
EQTransformer	5.2MB	✓	✓	1.0	1.0	1.0	0.0	0.03	0.99	0.99	0.99	0.0	0.11	0.99	0.96	0.98	Att.	
LCANet	3.7MB	✓	✓	1.0	1.0	1.0	0.0	0.01	1.0	0.99	0.99	0.0	0.04	1.0	0.96	0.98	Att.	
SeismicSense $_{float}$	756KB	✓	✓	0.99	0.99	0.99	0.0	0.12	0.98	0.99	0.99	0.0	0.30	0.97	0.99	0.98	RNN	
SeismicSense $_{int}$	189KB	✓	✓	0.99	0.99	0.99	0.0	0.16	0.96	0.99	0.98	0.2	0.38	0.93	0.98	0.96	RNN	✓

additional sensors to improve localization. This drop in performance is attributed to the integer quantization of the floating-point model. Furthermore, EQTransformer and LCANet are 28 and 20 times larger than SeismicSense, respectively (see Figure 6b).

We also present the mean error (μ) and standard deviation (std) to demonstrate the effectiveness of our model's phase-picking capabilities alongside PhaseNet, PickNet, LCANet, and EQTransformer. For SeismicSense, the standard deviation (std) of the error is 0.16 seconds for P-picks and 0.38 seconds for S-phase picks, which is a 4 to 5 times higher std compared to SOTA models. The increase in std especially for S-picks compared to SOTA models may not be ideal. We acknowledge this trade-off when deploying our model on MCUs. While our recall, and F1 scores align well with the SOTA models (EQTransformer and LCANet), the higher standard deviation of the error suggest a marginally decreased precision in our picks. However, this is a trade-off that we accept in our pursuit of deploying the model on embedded devices. This also impacts the accuracy of source-receiver distance calculations, and consequently, the localization of the epicenter. To mitigate this disadvantage, we can include more nodes in the localization process, which helps to improve the overall reliability of the location estimates and compensates for the increased standard deviation in our picks.

In our evaluation of SeismicSense, we compare it with EQTransformer using an additional dataset. Both models were initially trained on the STEAD dataset and subsequently fine-tuned using the Iquique dataset [23], which consists of offshore seismic events, in contrast to the onshore events represented in the STEAD dataset. Our results show that on the Iquique dataset, SeismicSense achieves comparable precision, recall, and F1 scores to EQTransformer (see Table 2). However, SeismicSense exhibits a slight increase in mean error by 0.06 seconds and in standard deviation by 0.04 seconds for S-pick compared to EQTransformer.

5.5 Resource Requirements

We evaluate the resource requirements of SeismicSense's integer-quantized models on our three target devices (nRF52840, nRF5340, and STM32H747i) by analyzing the total resource consumption of the combined model and its individual sub-models, i.e. the encoder and decoders. This assessment includes detailed analyses of RAM and flash usage, as well as execution times.

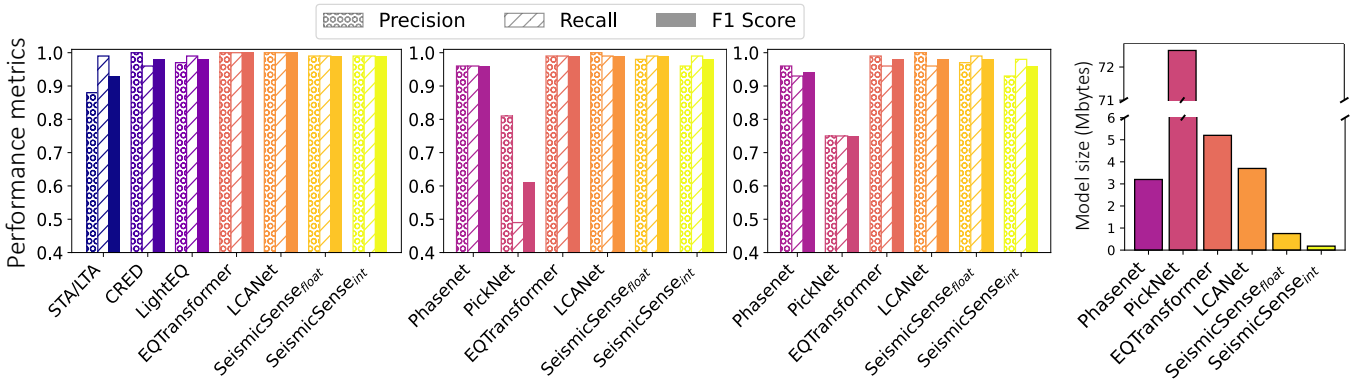
The RAM and flash requirements for SeismicSense are approximately 189 KB and 415 KB, respectively, corresponding to 72% RAM utilization and 40% flash usage on nRF52. The encoder requires 117 KB of RAM, while each decoder requires 99 KB. Additionally, the flash requirements for the encoder and decoder are 283 KB and 154 KB, respectively. The RAM and flash requirements for the individual sub-models exceed those of the entire model. This is attributed to the additional buffers and graph information, resulting in a higher resource demand at the sub-model level.

Table 3 presents the inference latency of SeismicSense on our target devices along with inference latency for each individual sub-model. We only report numbers for one decoder block, as they have the same structure, they also have identical latency. To further enhance the model's latency, we utilize the built-in integer DSP accelerators on these MCUs. SeismicSense's total processing latency is 6.57 sec on nRF52, 5.22 sec on nRF53, and 0.56 sec on STM32, with DSP enabled. We observe approximately an 18-fold improvement in the overall SeismicSense inference latency compared to non-accelerated inference (see Figure 8). This substantial reduction in latency ensures that SeismicSense effectively meets real-time execution constraints.

Figure 7 illustrates a cumulative analysis of energy consumption for each sub-model on an nRF52840 MCU for an input instance with enabled support for DSP acceleration. Our measurements indicate that the nRF52840 MCU, consumes 10.3 mJ, 11.5 mJ and 21.9 mJ for STA/LTA, encoder and each decoder blocks, respectively, when operating at 3.3 V.

5.6 System Lifespan Projection

In this section, we estimate the operational lifespan of SeismicSense on the nRF52 platform. For this, we assume that the STA/LTA algorithm triggers approximately five potential earthquake events per hour. Out of these, four are actual earthquakes. The STA/LTA algorithm runs every 10 milliseconds. The encoder and D decoder runs five times, whereas the P- and S-pick decoders run for four times within an hour. This results in the MCU being active for



(a) Detection and phase picking accuracy. (b) Model size.

Figure 6: (a) Comparison of detection, P- and S-phase picking accuracy for SeismicSense and SOTA models. STA/LTA, CRED, and LightEQ solely perform the detection task, whereas Phasenet and PickNet exclusively handle phase picking. Similar to EQTransformer and LCANet, SeismicSense performs both detection and phase picking. We achieve on par accuracy for event detection and P-phase identification, but experience a minimal decrease in accuracy for S-phase identification. (b) Comparison of SeismicSense and SOTA models in terms of model sizes. The y-axis represents model sizes in megabytes (MB). SeismicSense demonstrates a model size that is 20 times smaller than LCANet and 28 times smaller than EQTransformer

Table 2: Comparison of the detection and phase picking accuracy between SeismicSense and EQTransformer on the Iquique dataset, both after fine-tuning. Performance metrics include mean error (μ), standard deviation of the error (std), precision (Pr), recall, and F1 score. The units for μ and std are seconds.

Model	Detection			P-phase detection					S-phase detection				
	Pr	Recall	F1	μ	std	Pr	Recall	F1	μ	std	Pr	Recall	F1
EQTransformer	1	0.99	0.99	0.00	0.05	1	0.99	0.99	0.14	0.86	1	0.90	0.95
SeismicSense $_{float}$	1	0.99	0.99	0.00	0.04	1	0.99	0.99	0.20	0.90	1	0.91	0.95

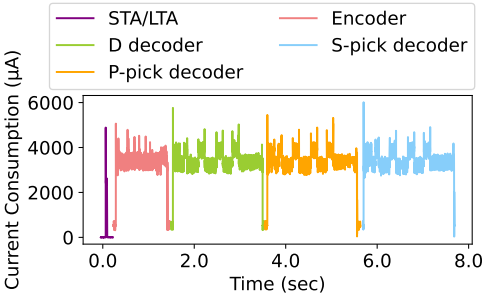


Figure 7: Energy measurements for our pipeline on nRF52840 with DSP enabled: The STA/LTA block executes in 1.5 msec, drawing 2.09 mA on average and consuming 10.3 mJ. The encoder performs a 1.1-sec inference with 3.17 mA average current, consuming 11.5 mJ. Each D-pick, P-pick, and S-pick decoder takes 2.0 seconds per inference, averaging 3.3 mA and consuming 21.9 mJ at a supply voltage of 3.3 V.

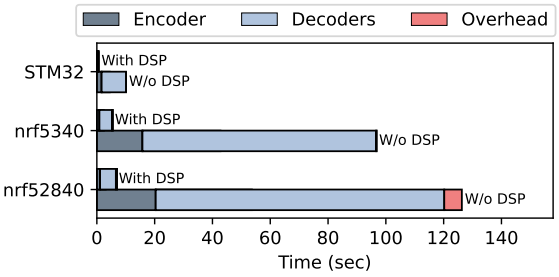


Figure 8: Effect of DSP accelerator on inference latency for SeismicSense: we observe a speedup of approx. 18x.

15.8% of the time, consuming 1.1295 mWh, while the remaining 84.2% is spent in sleep mode, drawing just 3.16 µA and consuming 0.0056 mWh. Combined, the total energy consumption amounts to 1.1342 mWh per hour.

Assuming the use of two alkaline batteries with a total usable energy capacity of 32 Wh (66% of their nominal capacity), the system is projected to operate continuously for approximately 3 years. State-of-the-art seismic data loggers, such as submarine seismic recorders, consume as little as 120 mW[2]. Running SeismicSense on such a recorder would increase power consumption by only 1%. This conservative estimate excludes energy needed for data transmission to the edge or cloud for epicenter localization.

[2]6D7 Datasheet by KUM: https://www.kum-kiel.de/products/6d7.html

Table 3: Comparison of inference latency for int-quantized models, with and without DSP support, for SeismicSense, encoder, and decoder. Inference can be improved approximately 18-fold with a DSP accelerator for the entire model, whereas a 17-fold improvement can be seen for encoder and decoder blocks.

	nRF52840			nRF5340			STM32h747i		
	without DSP	**with DSP**	Acc.	**without DSP**	**with DSP**	Acc.	**without DSP**	**with DSP**	Acc.
SeismicSense (sec)	126.351	6.575	x 19	96.648	5.222	x 18	10.118	0.565	x 18
Encoder block (sec)	20.378	1.121	x 18	15.789	0.894	x 17	1.658	0.105	x 15
Decoder block (sec)	33.281	1.956	x 17	27.004	1.540	x 17	2.826	0.162	x 17

6 Conclusion

This paper presents SeismicSense, a NN-based system that enables on-device earthquake detection and epicenter distance analysis (P- and S-phase detection) in resource-constrained settings. As a result, the system transmits only the phase arrival times—two distinct values—instead of continuously transmitting recorded data, enabling selective communication only when it detects an event. SeismicSense is partitioned into encoder/decoder sub-models to enable on-demand execution, which optimizes inference latency. The results of our integer-quantized NN model show a 99.4% F1-score for seismic event detection, with phase-picking models achieving F1-scores of 98% for P-phase detection and 96% for S-phase detection. The SOTA models, such as EQTransformer and LCANet, exhibit a slightly better performance of F1 scores of 100% for detection, 99% for P-phase, and 98% for S-phase identification. However, they demand at least 20 times more RAM than SeismicSense and are not as widely deployable as SeismicSense. Additionally, the integration of an integer accelerator further enhances inference latency, making SeismicSense ideal for low-power MCU deployments.

Acknowledgment

We thank our anonymous reviewers for their insight and suggestions for improvement. This project has received funding from Helmholtz School for Marine Data Science (MarDATA) [Grant No. HIDSS-0005].

References

[1] Stefanie J Baxter. 2000. *Earthquake basics.* Technical Report. Newark, DE: Delaware Geological Survey, University of Delaware.

[2] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. 2019. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 3009–3018.

[3] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Tiezhen Wang, et al. 2021. Tensorflow lite micro: Embedded machine learning for tinyml systems. *Proceedings of Machine Learning and Systems* 3 (2021), 800–811.

[4] Ramin MH Dokht, Honn Kao, Ryan Visser, and Brindley Smith. 2019. Seismic event and phase detection using time–frequency representation and convolutional neural networks. *Seismological Research Letters* 90, 2A (2019), 481–490.

[5] Gregory Giuliani, Hy Dao, Andrea De Bono, Bruno Chatenoux, Karin Allenbach, Pierric De Laborie, Denisa Rodila, Nikos Alexandris, and Pascal Peduzzi. 2017. LiMES: A framework for monitoring environmental changes from Earth Observations. *Remote Sensing of Environment* 202 (2017), 222–233.

[6] Jens Havskov, Peter Bormann, and Johannes Schweitzer. 2012. Seismic source location. In *New Manual of seismological observatory practice 2 (NMSOP-2)*. Deutsches GeoForschungsZentrum GFZ, 1–36.

[7] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. *arXiv preprint arXiv:1602.07360* (2016).

[8] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[9] Shengming Jiang. 2019. Marine internet for internetworking in oceans: A tutorial. *Future Internet* 11, 7 (2019), 146.

[10] Guojin Liu, Rui Tan, Ruogu Zhou, Guoliang Xing, Wen-Zhan Song, and Jonathan M Lees. 2013. Volcanic earthquake timing using wireless sensor networks. In *Proceedings of the 12th international conference on Information processing in sensor networks*. 91–102.

[11] Han Liu and Jian-zhong ZHANG. 2014. STA/LTA algorithm analysis and improvement of Microseismic signal automatic detection. *Progress in Geophysics* 29, 4 (2014), 1708–1714.

[12] Naoki Maeda. 1985. A method for reading and checking phase times in autoprocessing system of seismic wave data. *Zisin* 38 (1985), 365–379.

[13] S Mostafa Mousavi, William L Ellsworth, Weiqiang Zhu, Lindsay Y Chuang, and Gregory C Beroza. 2020. Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. *Nature communications* 11, 1 (2020), 1–12.

[14] S Mostafa Mousavi, Yixiao Sheng, Weiqiang Zhu, and Gregory C Beroza. 2019. STanford EArthquake Dataset (STEAD): A global data set of seismic signals for AI. *IEEE Access* 7 (2019), 179464–179476.

[15] S Mostafa Mousavi, Weiqiang Zhu, Yixiao Sheng, and Gregory C Beroza. 2019. CRED: A deep residual network of convolutional and recurrent units for earthquake signal detection. *Scientific reports* 9, 1 (2019), 1–14.

[16] Thibaut Perol, Michaël Gharbi, and Marine Denolle. 2018. Convolutional neural network for earthquake detection and location. *Science Advances* 4, 2 (2018).

[17] Zachary E Ross, Men-Andrin Meier, Egill Hauksson, and Thomas H Heaton. 2018. Generalized seismic phase detection with deep learning. *Bulletin of the Seismological Society of America* 108, 5A (2018), 2894–2901.

[18] Omar M Saad and Yangkang Chen. 2020. Earthquake detection and P-wave arrival time picking using capsule neural network. *IEEE Transactions on Geoscience and Remote Sensing* 59, 7 (2020), 6234–6243.

[19] Christos D Saragiotis, Leontios J Hadjileontiadis, and Stavros M Panas. 2002. PAI-S/K: A robust automatic seismic P phase arrival identification scheme. *IEEE Transactions on Geoscience and Remote Sensing* 40, 6 (2002), 1395–1404.

[20] Amadej Trnkoczy. 2009. Understanding and parameter setting of STA/LTA trigger algorithm. In *New Manual of Seismological Observatory Practice (NMSOP)*. Deutsches GeoForschungsZentrum GFZ, 1–20.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[22] Jian Wang, Zhuowei Xiao, Chang Liu, Dapeng Zhao, and Zhenxing Yao. 2019. Deep learning for picking seismic arrival times. *Journal of Geophysical Research: Solid Earth* 124, 7 (2019), 6612–6624.

[23] Jack Woollam, Andreas Rietbrock, Angel Bueno, and Silvio De Angelis. 2019. Convolutional neural network for seismic phase classification, performance demonstration over a local seismic network. *Seismological Research Letters* 90, 2A (2019), 491–502.

[24] Yue Wu, Youzuo Lin, Zheng Zhou, David Chas Bolton, Ji Liu, and Paul Johnson. 2018. DeepDetect: A cascaded region-based densely connected network for seismic event detection. *IEEE Transactions on Geoscience and Remote Sensing* 57, 1 (2018), 62–75.

[25] Clara E Yoon, Ossian O'Reilly, Karianne J Bergen, and Gregory C Beroza. 2015. Earthquake detection through computationally efficient similarity search. *Science advances* 1, 11 (2015), e1501057.

[26] Tayyaba Zainab, Jens Karstens, and Olaf Landsiedel. 2023. LightEQ: On-Device Earthquake Detection with Embedded Machine Learning. In *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*. 130–143.

[27] Yu Zhao, Pan Deng, Junting Liu, Mulan Wang, and Jiafu Wan. 2022. LCANet: Lightweight Context-Aware Attention Networks for Earthquake Detection and Phase-Picking on IoT Edge Devices. *IEEE Systems Journal* 16, 3 (2022).

[28] Weiqiang Zhu and Gregory C Beroza. 2019. PhaseNet: A deep-neural-network-based seismic arrival-time picking method. *Geophysical Journal International* 216, 1 (2019), 261–273.