Image registration for distributed acoustic sensing acquired data using convolutional neural networks

Heather K. Hardeman-Vooys*, Matt McDonald†, and Michael P. Lamoureux *

ABSTRACT

We provide a brief overview of convolutional neural networks. Then we introduce a training set extracted from real data. We test the trained convolutional neural network on a portion of the training set to determine accuracy. Afterwards, we employ this trained convolutional neural network to identify events in data which contains walking and digging events. We then discuss our results which suggests that more than just the source of an event affects the way the event looks in the data.

INTRODUCTION

Using distributed acoustic sensing and fibre-optic cables to acquire seismic data enables the generation of terabytes worth of data in a short amount of time. While a plethora of data provides an abundance of information, it is necessary to interpret that data. In particular, seismic imaging and interpretation depends on finding events and anomalies in seismic data. With terabytes of data to consider, the creation of a new method beyond human capabilities becomes essential for analyzing data efficiently. One answer for this interpretation problem is through machine learning and image-recognition.

Over the last century, the advancement of computers led to the question of how to teach computers to perform tasks. Recently, image-recognition joined the ranks of such tasks for computers to learn. One aspect of machine learning that is especially useful in image-recognition are neural networks. Given the large nature of seismic data sets, convolutional neural networks are among the most effective.

In Hardeman et al. (2017), we combine a convolutional neural network with an inverted wavelet tree in order to classify events in DAS. While convolutional neural networks are very efficient, events in DAS data can be larger than a typical training image. They require some help with reducing the scale of events. One answer to this problem is the tree-structured wavelet transforms. These transformations apply a wavelet transform across a tree where each node of the tree represents the data convolved with a wavelet in the wavelet family. The wavelets themselves correspond to a specific frequency band, and consequently scale the original data to smaller sizes. Therefore, pairing convolutional neural networks with tree-structured wavelet transforms provides a scale-invariant image-recognition method. In particular, the inverted wavelet tree proves to be very useful for addressing scale-invariance.

In the following section, we introduce neural networks with a focus on convolutional neural networks (CNN). In the next section, we discuss an application of convolutional

*University of Calgary, Department of Mathematics
†Fotech Solutions

neural networks to an image-recognition problem similar to the one found in Hardeman et al. (2017). In this case, we train a CNN to identify between three events and consider image registration for DAS data. In the example, we discuss the architecture of the CNN created to detect events. We built the CNN with the help of code from the UFLDL Deep Learning Tutorial at Stanford University; c.f. (Ng et al., 2013) and (Yang, 2014). From there, we consider the training set and test it on a portion of known events. For the data set, we then introduce the CNN we trained to identifying walking, digging, and noise events in DAS data. Next, we consider application to two data sets acquired at different pulse-repetition-frequencies using a Fotech Solutions interrogator which contain walking and digging events. Our results led to a discussion of image-registration when applying image-recognition techniques to DAS-acquired data. Finally, we conclude.

For multiple hyperbolic events such as walking and digging, the convolutional neural network drops to an 80% accuracy, potentially due to the similarities between walking and digging data sets. We saw that the pulse-repetition-frequency (PRF) at which the data was collected also affects the success of the convolutional neural network. With a variety of possible obstacles affecting the outcome of the convolutional neural network, procuring a method which normalizes the distributed acoustic sensing acquired data should be investigated in order to improve the success of a convolutional neural network for image-recognition purposes on DAS acquired data if we want it to work for data acquired using different PRF or other methods. This outcome also suggests that the way the DAS system collects the data also affects how well the CNN works at identifying specific images.

CONVOLUTIONAL NEURAL NETWORKS

The natural world provides inspiration for mathematical and computational methods, and neural networks hold a strong resemblance to the human body's nervous system (Wu, 1992). Just as the body's nervous system interprets outside stimuli to determine how the body should react or decipher information, an artificial neural network utilizes 'neurons' to make decisions, or output answers, about information fed into the network.

An activation function $f : \mathbb{R} \to \mathbb{R}$ is utilized by the neural network to make decisions (Adcock and Dexter, 2019). There are standard activation functions used in the literature such as the sigmoid function, the identity, the arctangent function, and the hyperbolic tangent function to name a few. We use the sigmoid function as the activation function for our neural network in this paper as they do in Ng et al. (2013). The activation function is scaled using a weight W and translated using a bias b. The parameters W and b in each node of the neural network are optimized in a hypothesis function $h_{W,b}$ using a cost function J as part of the neural network's training. The hypothesis function $h_{W,b}$ is defined as

$$h_{W,b}(x) = f\left(\sum_{i=1}^{m} W_i x_i + b_i\right) \tag{1}$$

for m examples (x, y). For this paper, the cost function J is the average of the least squares difference between the hypotheses $h_{W,b}$ of each sample x and the label y for that sample with a regularization term (Ng et al., 2013). Mathematically, we write the cost function J as follows:

$$J(W, b; x, y) = \frac{1}{2} \left\| h_{W,b}(x) - y \right\|^2 \tag{2}$$

where given m examples of samples paired with labels (x, y), we expand the cost function J to be:

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} J(W,b;x^{(i)},y^{(i)})\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ji}^{(l)}\right)^2 \tag{3}$$

$$= \left[\frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ji}^{(l)}\right)^2. \tag{4}$$

The term λ is the weight decay parameter which helps prevent overfitting by decreasing the magnitude of the weights.

For the most part, the architecture of a neural network can be generalized to a connected graph with layers and units in each layer. Several different types of layers exist for neural networks, each of which performs different tasks. The type of neural network is dependent on how the units in these layers are connected to units of other layers. For larger data sets, it is important to limit these connections between units, as this decreases computational cost. One way to address the need for limiting connections is a convolutional neural network.

In a convolutional neural network, there is at least one convolutional layer. A convolutional layer has a number of units equal to the number of filters defined when training the network. The choice of filters determines the number of weights in the layer; it is also necessary to fix a dimension for the filters. For improved accuracy, the dimension of the filters should be the size of the event being trained to detect. These filters are convolved through the data and produce a smaller set of convolved features. The convolved features save the computational cost of processing the entire data set in a fully-connected network.

After the convolutional layer, the results from each unit are directed to the next layer. This layer can be another convolutional layer or a pooling layer. Depending on the size of the data, it may be prudent to have another convolutional layer. A pooling layer takes the convolved features and pools them together using different methods to produce what is called a pooled feature. This layer condenses the data further, cutting down on computational costs and allowing for much faster classification. Some methods employed for pooling include max-pooling or mean-pooling. The max-pooling involves taking the maximum of each convolved feature and the mean-pooling takes the average of each convolved feature. Not only does pooling reduce the dimension of the data, but it also allows for less overfitting, which in turn improves results.

Depending on the complexity of the convolutional neural network, the pooling layer can feed into another convolutional layer or into another pooling layer. Our convolutional neural network will feed into a densely-connected output layer. The output layer gives the results or predictions of the neural network. For example, in hand-written digit recognition, there would be 10 output units: one for each integer between 0 and 9. In this paper, the CNN has three output units in the output layer: one for if an image has walking events, one if it has digging events, and one if it has noise events.

Networks are trained by inputting learned parameters. The weights involved in the convolutional layer are part of these learned parameters. In order to learn the parameters

for a neural network, it is necessary to have a training set. It follows that if a network must learn to predict events it needs training objects containing samples of the events which should be predicted. In image recognition, the training set contains images of events the user wants to detect.

Parameters can be taught using gradient methods. These gradient methods must converge to a local optima which is why batch methods are often used for convolutional neural networks. We used the stochastic gradient descent (SGD) method to learn our parameters. The update step of the SGD method for a pair $(x^{(i)}, y^{(i)})$ is

$$\theta = \theta - \alpha \nabla_\theta J(\theta; x^{(i)}, y^{(i)}) \tag{5}$$

where α is the learning rate, x is the sample, and y is the label assigned to x. The learning rate α is chosen to be small prior to computation, and is updated after each iteration. The stochastic gradient descent method overcomes the slow calculation common to batch methods (Ng et al., 2013). It learns parameters by computing the gradient of the cost function $J(\theta)$ over a few batches of training images as opposed to the entire training set to update the parameters after each iteration.

For the SGD method, it is also necessary to pick a momentum. The momentum helps push the SGD objective down any shallow ravines present in the calculation in order to improve performance of the method. As with the learning rate, the momentum is updated after each iteration. The initial choice of the momentum is around 0.5 before being chosen between 0.9 and 1. The SGD outputs learned parameters after a user-determined number of loops through the training set. The user then employs the learned parameters in the neural network to classify events.

For a detailed tutorial on neural networks and other deep learning methods, see Ng et al. (2013).

APPLICATIONS

We now apply the methods discussed earlier in the paper to build a convolutional neural networks. We develop our CNN using supplementary code from Ng et al. (2013) and Schmidt (2005). The CNN is trained to identify between multiple types of events.

Convolutional neural network with three classes

We teach a convolutional neural network to distinguish between three different events: someone walking next to the fibre, someone digging next to the fibre, and noise. Distributed acoustic sensing detects both digging and walking as hyperbolic events. To differentiate between these events, we train on images which contain multiple events instead of just one event. We hypothesize that the distance between the steps and the digging will help differentiate between the two cases.

We take this experiment a step further than the last example in that we will apply the neural network to varying downsampled versions of the data set and compare the results. Specifically, we run the neural network on the data first without any downsampling, then when downsampled by 10 shots, and finally when downsampled by 50 shots.

Architecture of the convolutional neural network

The convolutional neural network has three layers, not including the input layer. The first layer is a convolutional layer which utilizes 20 filters of 101×101 matrix entries. The second layer is a pooling layer that applies a mean-pooling. The outputs propagate into a softmax regression with cross entropy objective. The dimension of the pooling features was 2×2. The final layer is the output layer which contains three units determining one of three cases: whether an image is of walking, digging, and noise events.

We employ a stochastic gradient descent method using momentum in order to optimize the parameters of a given objective. This function involves using the minFunc function created by Mark Schmidt (Schmidt, 2005). The SGD method calculates the learned parameters to train the convolutional neural network to detect hyperbolas in data sets based on the cost function $J(\theta)$ and allows for a faster convergence. We set the momentum at 0.99. To train the neural network, we use the learning rate $\alpha = 0.01$. The SGD subsamples over batches containing 32 training samples. The training ran through the training set three times in order to educate the convolutional neural network.

Training set

We extract images from real data to create a training set. Instead of working with only one data set, we employ 207 data sets: 96 contain walking events and 111 contain digging events. For noise events, we remove images from the data sets which did not overlap with walking or digging events in the data. Given that noise is more prevalent in data, we limit the number of noise events to a similar amount of digging and walking events in order to avoid giving the neural network a bias toward identifying noise.

Figures 1 and 2 provide examples of data sets containing walking and digging respectively. Within both of these data sets, we see examples of noise. The walking is located between 50 and 175 meters and 2000 to 11000 milliseconds in Figure 1. The digging events are located between 50 and 100 meters and 3000 and 10000 milliseconds in Figure 2.

The training set contains 3232 images of size 128×128. The set has 1002 walking events, 1142 digging events, and 1086 noise events. The top row of Figure 3 exhibits an example of a walking event on the left and a digging event on the right. The bottom row of Figure 3 provides two examples of a noise event. Observe from the bottom row of Figure 3 that noise events vary drastically. This may prove to cause the CNN some difficulty in distinguishing between the three different classes.

Test set

We train the CNN on 3232 images of the training set and test the results on a testing set of 2155 images. In Figure 4, we show the probabilities that an image is someone walking, someone digging, or noise in the data. Figure 5 provides some examples of what the CNN decided represented a walking event, a digging event, and a noise event.

In Figure 4, the prevalence of red and yellow dots strictly greater than 0 and strictly less

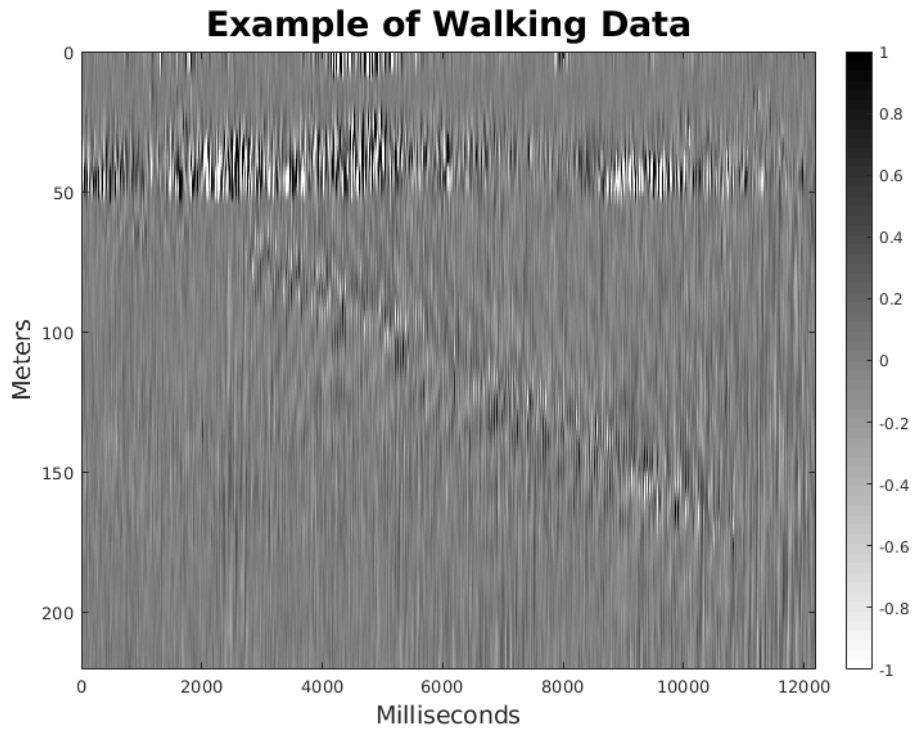

FIG. 1: An image of someone walking parallel to a fibre-optic cable acquired using distributed acoustic sensing. The footsteps are located between 50 and 175 meters and 3000 and 11000 milliseconds. The walking data set is employed to create the walking events in the training set for the convolutional neural network.

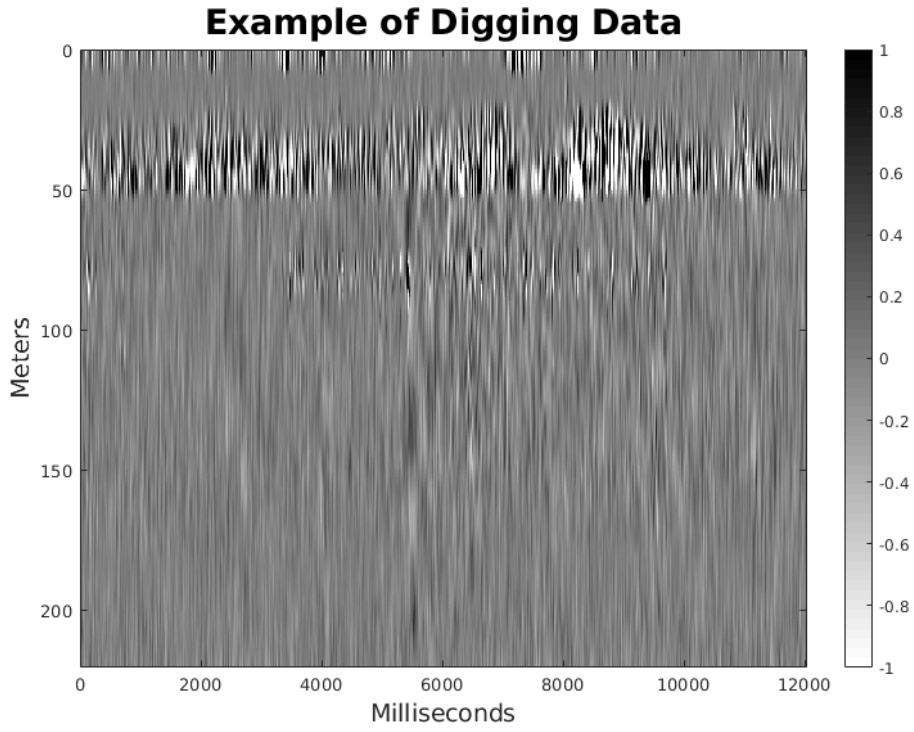

FIG. 2: An image of someone digging next to a fibre-optic cable acquired using distributed acoustic sensing. The digging events are located between 50 and 100 meters and 3000 and 10000 milliseconds. The digging data set is one of several data sets employed to create the digging events in the training set for the convolutional neural network.

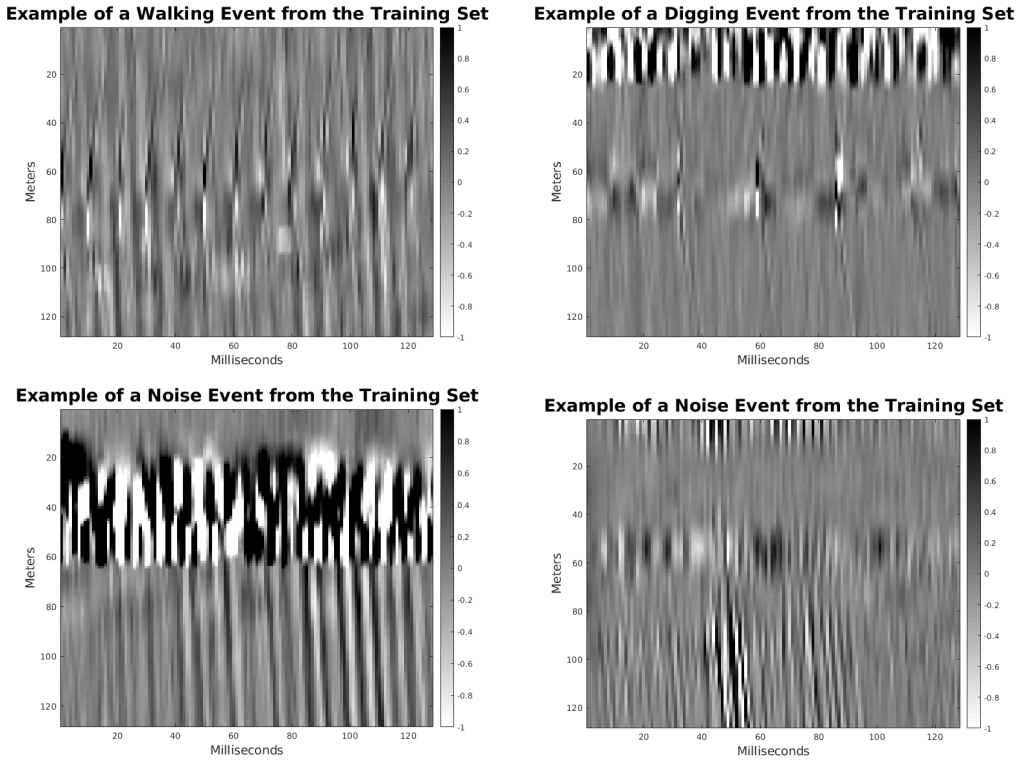

FIG. 3: (Top Left) An image of a walking event from the training set.
(Top Right) An image of a digging event from the training set.
(Bottom) Examples of noise events from the training set.

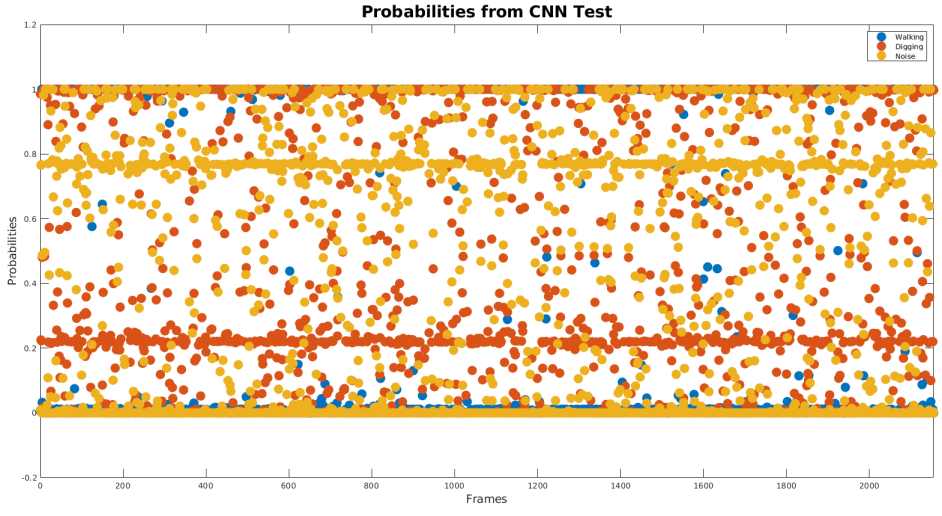

FIG. 4: The probability that each image in the 2155 images of the test set is a walking event (blue), digging event (red), or a noise event (yellow).

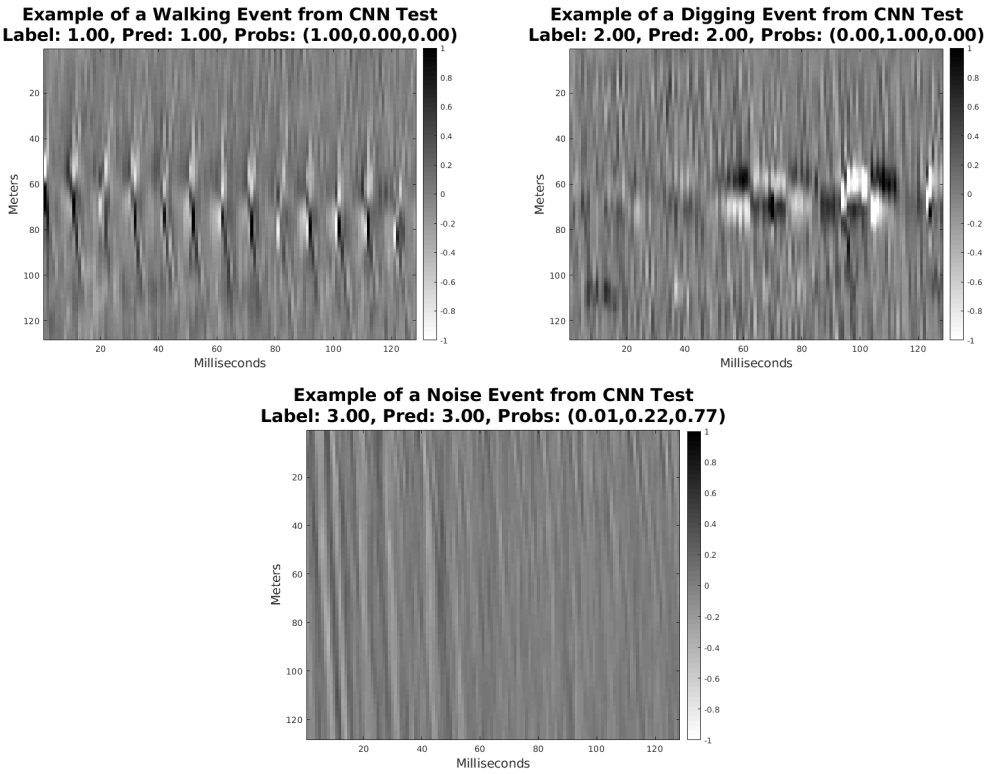

FIG. 5: (Top left) An example of a walking event the CNN predicted from the test set images.
(Top right) An example of a digging event the CNN predicted from the test set images.
(Bottom) An example of a noise event the CNN predicted from the test set images.

than 1 suggests that the neural network struggled to distinguish between digging and noise. The CNN did not have confidence in what was noise versus what is digging. Fewer blue dots representing walking events strictly greater than 0 and strictly less than 1 suggests that the CNN possessed more confidence in what was a walking event.

The CNN has an accuracy of approximately 80%, which is a little more than 5% less than the neural network from the experiment in Hardeman et al. (2017). The decrease in accuracy may be attributed to the fact that noise is the third class given that noise often varies in appearance. The similarities between walking and digging may also contributed to the accuracy's decline. Notice that the CNN possessed more than 5 times the images with which to train than the CNN from the previous example. Intuitively, it seems as if more training images would be helpful for teaching a CNN; however, we see in this experiment that that more training images does not always help increase the accuracy of a training set.

While a significant decline occurred in accuracy between the CNNs in the two experiments, 80% accuracy is still promising for applications to other data. We explore the success of applying the CNN to two data sets which contain walking and digging events in the following section.

Test on outside samples

To create outside examples, we glued a data set containing walking and a data set containing digging together. We did this for two data sets acquired at different pulse repetition frequencies - 4kHz and 6kHz. In a DAS system, the pulse repetition frequency is the rate at which a laser pulse can be sent down the fibre and all the backscattered light exits before we launch the next pulse. Given how large DAS data can be, we also apply the CNN to three different decimations of the two data sets — no decimation, decimation by 10 shots, and decimation by 50 shots. We created the training set from data decimated by 50 shots in order to keep the image dimension of the training images 128×128. This fact suggests that the CNN will do the best on the data decimated by 50 shots for both data sets.

Figure 6 displays the walking and digging data set with no decimation acquired at a pulse repetition frequency of 4kHz. The walking events lie between 50 and 175 meters and 0.25×10^4 and 1.2×10^4 milliseconds. The digging events are located between 50 and 100 meters and 1.5×10^4 and 2×10^4 milliseconds.

Given how the inverted wavelet tree creates levels after the data in previous levels, it is difficult to calculate the accuracy of the CNN on real data at these levels; however, we can judge the accuracy of the first level. We only consider the probabilities of walking, digging, and noise events in the first level of the wavelet tree. Figure 7 shows the probabilities a frame of the data is walking, digging, or noise. Because the data is not decimated, it has more frames. The walking and digging events were localized within the data so many frames will contain noise events which explains why the orange dots are more prevalent in Figure 7.

Let us consider decimating the data by 10 shots; see Figure 8. Visually, it is difficult to distinguish any differences in the two data sets from Figures 6 and 8. Given that we

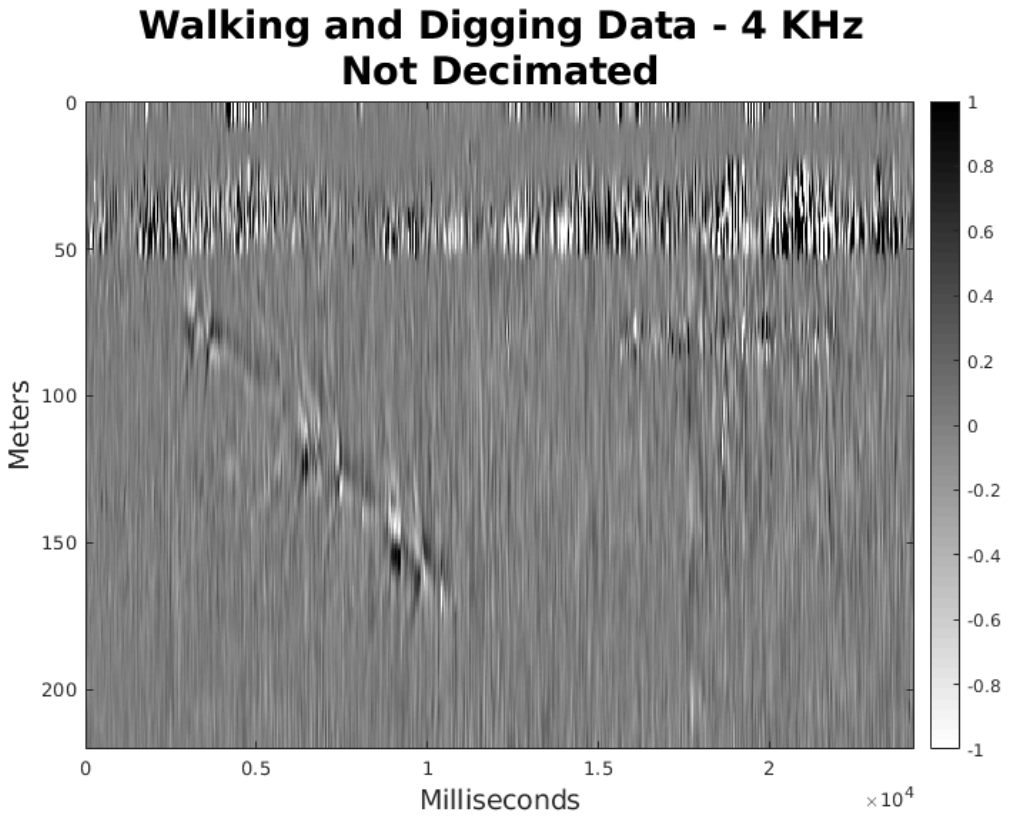

FIG. 6: An image of someone walking parallel to the fibre-optic cable and someone digging next to the fibre. The data was acquired at a pulse repetition rate of 4kHz. No decimation has been applied to the data.

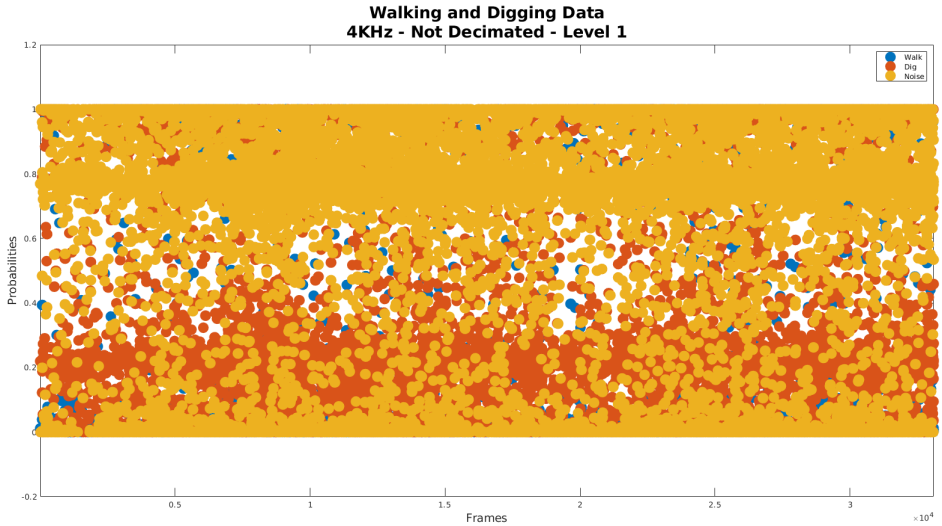

FIG. 7: Probabilities of walking, digging and noise events from applying the trained CNN to the 4kHz PRF walking and digging data set which is not decimated. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

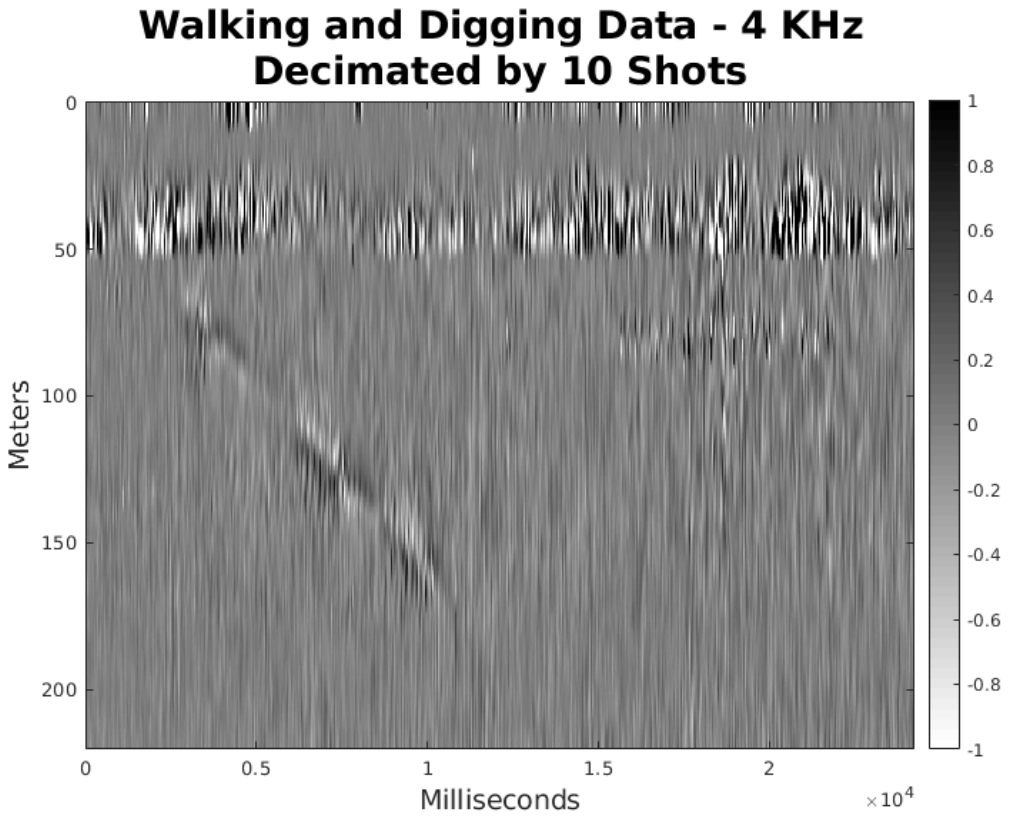

FIG. 8: The 4kHz PRF walking and digging data set decimated by 10 shots. The walking events are located between 50 and 175 meters and 0.25×10^4 and 1.1×10^4 milliseconds. The digging events reside between 50 and 100 meters and 1.5×10^4 and 2.25×10^4 milliseconds.

removed every 10th shot from the data set to create the image seen in Figure 8, there will be fewer frames for the CNN to identify as walking, digging, or noise. We see this reflected in the significant decrease in the density of the probabilities in Figure 9.

While the number of frames drops between the zero decimation and the decimated by 10 shots cases, the probabilities for digging and noise show prominence in the area strictly greater than 0 and strictly less than 1; however, we witness less confidence with regards to some walking events (blue dots). In both cases, a large cluster of noise probabilities landed around 0.8 and a large cluster of digging probabilities resided at approximately 0.2, which suggests that the CNN had some difficulty distinguishing between noise and digging.

Now, we examine the case when we decimate the 4kHz PRF walking and digging data by 50 shots; see Figure 10. Here the walking events are located between 50 and 175 meters and 0.25×10^4 and 1.1×10^4 milliseconds. The digging events reside between 50 and 100 meters and 1.5×10^4 and 2.25×10^4 milliseconds.

As with the 10 shot case, there are fewer frames. Figure 11 depicts the probabilities for the walking, digging, and noise. The fewer frames allows us to decipher more details from the probabilities. Recall from the image of the data that the walking occurs at the beginning

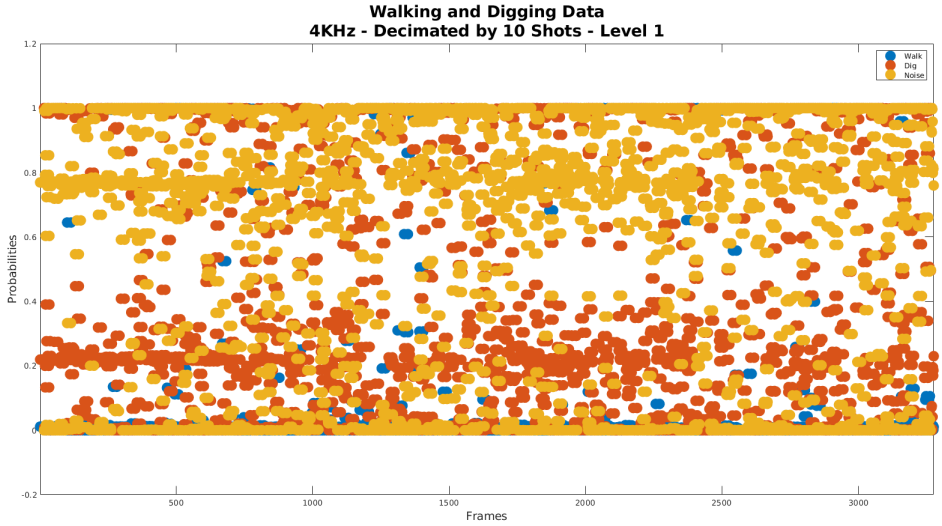

FIG. 9: Probabilities of walking, digging, and noise events from applying the trained CNN to the 4kHz PRF walking and digging data set, decimated by 10 shots. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

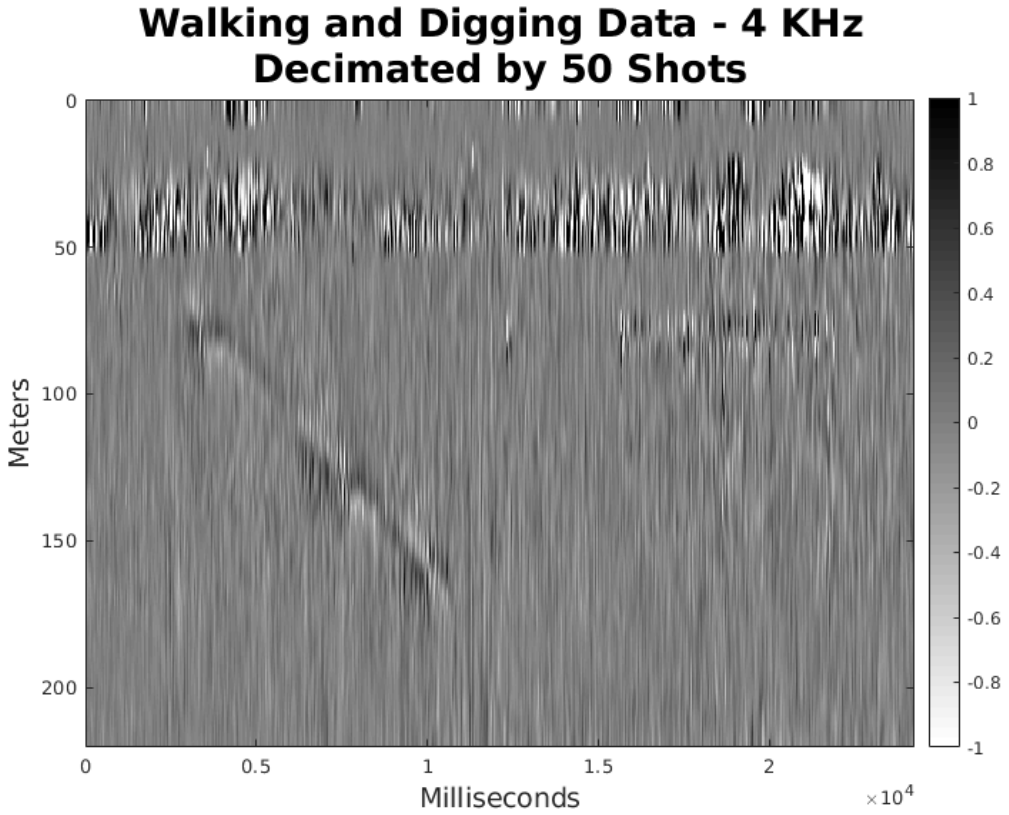

FIG. 10: The 4kHz PRF walking and digging data set decimated by 50 shots. The walking events are located between 50 and 175 meters and 0.25×10^4 and 1.1×10^4 milliseconds. The digging events reside between 50 and 100 meters and 1.5×10^4 and 2.25×10^4 milliseconds.

FIG. 11: Probabilities of walking, digging, and noise events from applying the trained CNN to the 4kHz PRF walking and digging data set, decimated by 50 shots. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

of the data set, while the digging occurs towards the end of the data set. In Figure 11, more frames identify as walking in the first 300 frames vs the last 300 frames. Similarly, more frames have a higher probability of being a digging event in the last 300 frames than in the first 300.

Let us consider the data acquired using a pulse repetition frequency of 6kHz. Figure 12 presents the image of someone walking next to the fibre and then someone digging next to the fibre acquired at a pulse repetition frequency of 6kHz with zero decimation of the data. As with the 4kHz case, we consider the CNN results from the data which was not decimated, the data decimated by 10 shots, and then the data decimated by 50 shots. For all cases, we find the walking events between 25 and 150 meters and 0.25×10^4 and 1×10^4 milliseconds. The digging events are located between 25 and 75 meters and 1.5×10^4 and 2×10^4 milliseconds.

In Figure 13, we see the probabilities that an event is walking, digging, or noise for the data which was not decimated. The results are similar to the 4kHz PRF case; however, the density of probabilities in the last few frames is less spread out than in the 4kHz case.

Figure 14 provides an image of the walking and digging data acquired at a PRF of 6kHz decimated by 10 shots.

As with the 4kHz, we see fewer frames as witnessed in the decrease in the dot density seen in Figure 15. Interestingly, some frames near the beginning of the data appear to be identified as digging events given the cluster of red dots near probability 1 in this region. Since digging events occur at the end of the data set, this suggests that the CNN struggled with identifying frames for this data set.

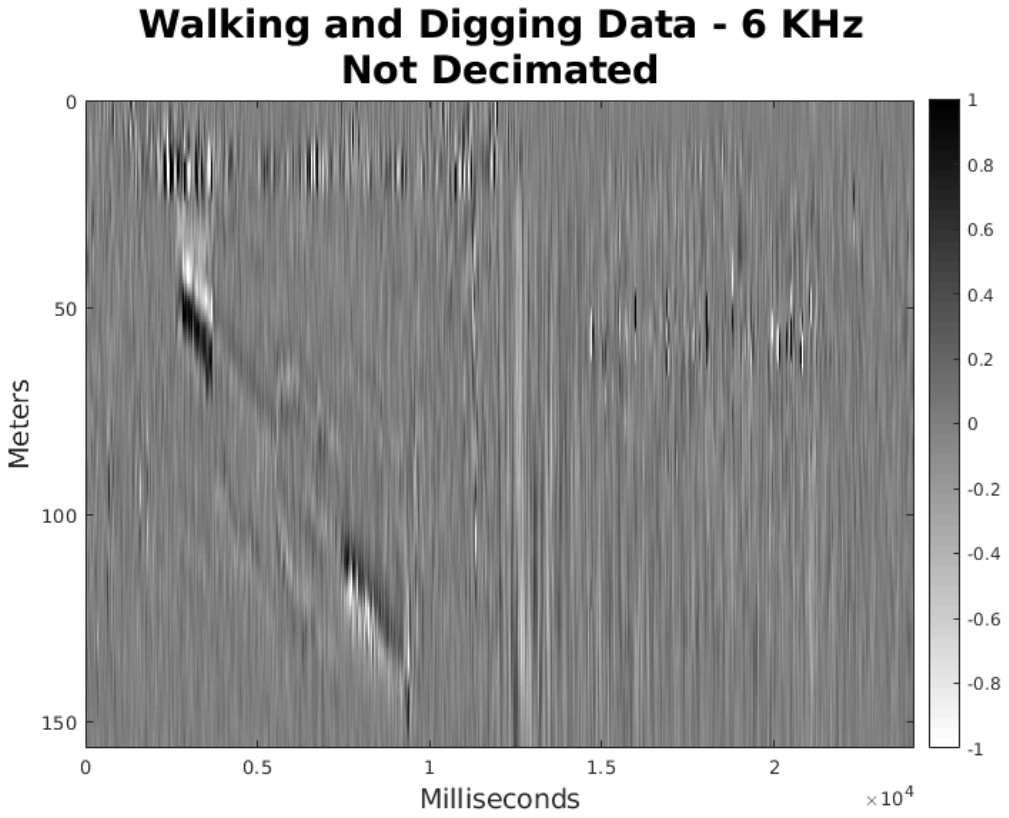

FIG. 12: The 6kHz PRF data set which has not been decimated of someone walking next to the fibre (between 25 and 150 meters and 0.25×10^4 and 1×10^4 milliseconds) and someone digging next to the fibre (between 25 and 75 meters and 1.5×10^4 and 2×10^4 milliseconds).

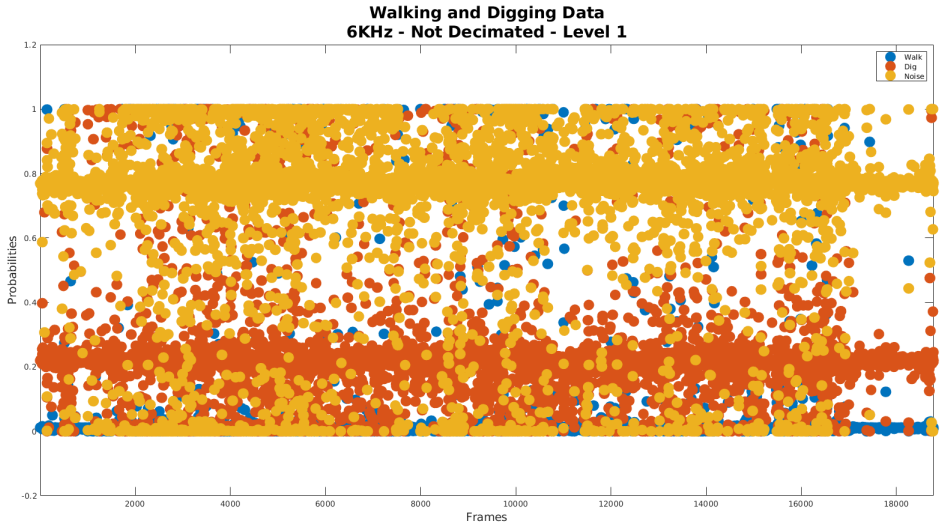

FIG. 13: Probabilities of walking, digging, and noise events from applying the trained CNN to the 6kHz PRF walking and digging data set with no decimation. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

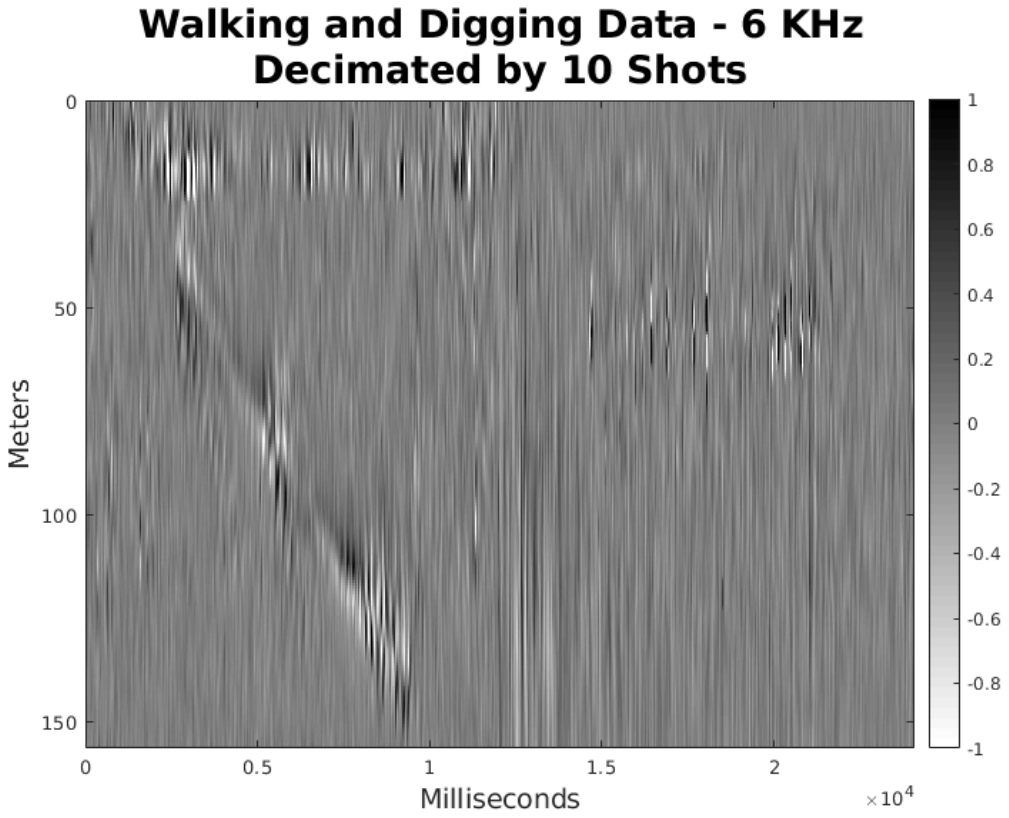

FIG. 14: The 6kHz PRF data set, decimated by 10 shots, of someone walking next to the fibre and someone digging next to the fibre. The walking events reside between 25 and 150 meters and 0.25×10^4 and 1×10^4 milliseconds. The digging events are located between 25 and 75 meters and 1.5×10^4 and 2×10^4 milliseconds.

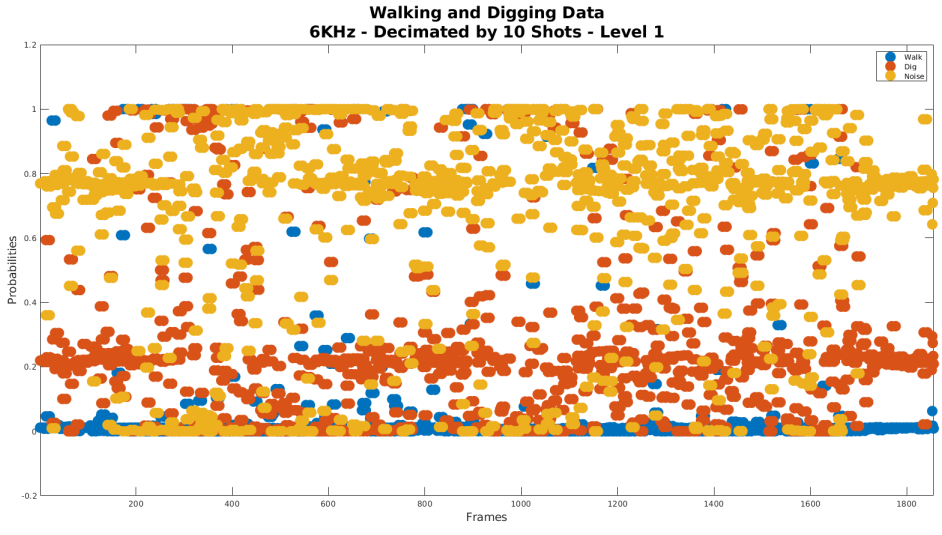

FIG. 15: Probabilities of walking, digging and noise events from applying the trained CNN to the 6kHz PRF data set, decimated by 10 shots. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

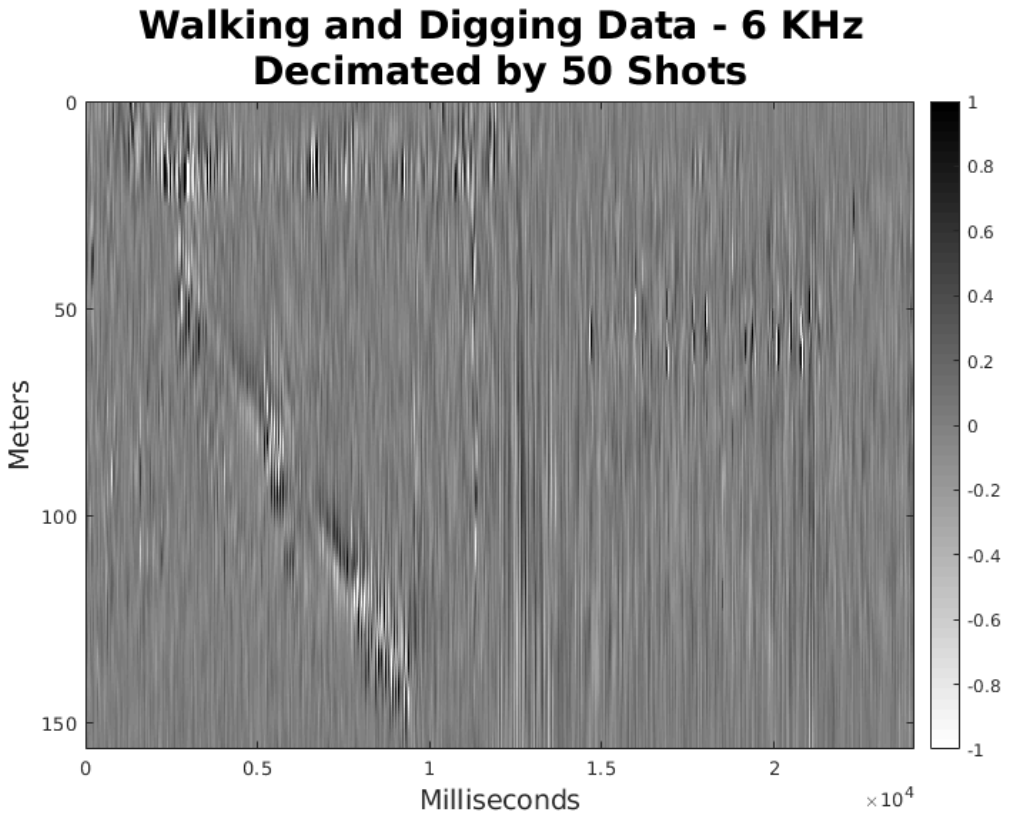

FIG. 16: The 6kHz PRF data set, decimated by 50 shots, of someone walking next to the fibre and someone digging next to the fibre. The walking events are between 25 and 150 meters and 0.25×10^4 and 1×10^4 milliseconds. The digging events are located between 25 and 75 meters and 1.5×10^4 and 2×10^4 milliseconds.

Finally, we consider the 6kHz PRF walking and digging data decimated by 50 shots as seen in Figure 16.

This case has significantly fewer frames. In Figure 17, we see a cluster of walking events in the first 300 frames which correlates to where the walking is in the data set. The CNN identifies a few frames near the beginning as digging events, signified by the red dots. Recall that the digging occurs near the end of the data set, so these frames were misidentified; however, a small cluster of digging events can be seen in the last few frames. Given the prominent cluster of walking events around probability 1 during the first 300 frames, the CNN likely performed adequately with regards to identifying walking; however, it does not appear to have done as well with regards to digging events.

Table 1 shows the accuracy of the CNN at identifying walking and digging events in each of the data sets. Specifically, we only looked at the results when a frame of data overlapped with known locations of walking or digging events.

For both PRFs, the CNN had the highest accuracy when we decimated the data by 50 shots. This result follows given that we built the training set using 50 shot decimated data. Interestingly, the CNN performed significantly better on the 4kHz data than the 6kHz data.

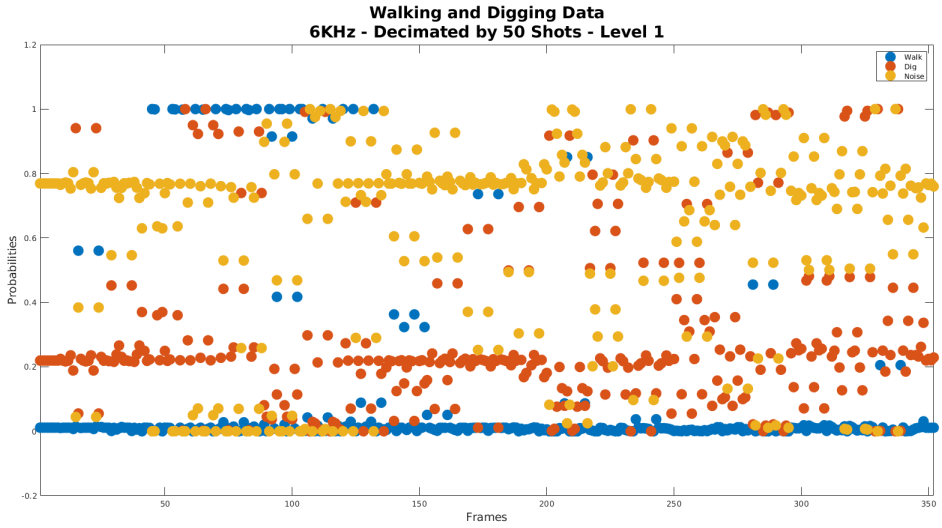

FIG. 17: Probabilities of walking, digging, and noise events from applying the trained CNN to the 6kHz PRF walking and digging data set, decimated by 50 shots. The probabilities in each frame of a walking event (blue), digging event (red), and noise event (yellow) being present on Level 1.

Table 1: The accuracy of the convolutional neural network for each type of 4kHz PRF and 6kHz PRF acquired walking and digging data.

	No Decimation	10 Shots Decimation	50 Shots Decimation
4kHz	11%	34%	77%
6kHz	9%	0%	39%

Potentially, we employed more 4kHz data sets to build the training set than 6kHz data; given that the CNN's accuracy never achieves higher than 40% for the 6kHz case, this is likely the case. The accuracy for the 4kHz case dips drastically when the data is decimated by 10 shots or less, which supports our theory that the CNN would work most successfully on the data decimated by 50 shots. Furthermore, the 4kHz case, when decimated by 50 shots, had a 77.14% accuracy which is around 3% worse than how well the CNN performed on the test set.

Figure 18 provides examples of frames the CNN identified correctly (left column) and incorrectly (right column) of walking, digging, and noise events in the 4kHz PRF acquired data. Interestingly, the CNN identified some walking as digging as seen in the middle row of the figure. Moreover, in the bottom row, we see that the CNN distinguished a noise event which clearly has walking in it.

IMAGE REGISTRATION IN DAS-ACQUIRED DATA

Before we conclude, we observe that the drastic difference in accuracy between the 4kHz case and the 6kHz case suggests that convolutional neural networks would struggle in application to identifying seismic events which contain multiple hyperbolic events in data acquired using distributed acoustic sensing and fibre-optics. Even within the more successful 4kHz case, we saw a dip in success as the number of shots we decimated by

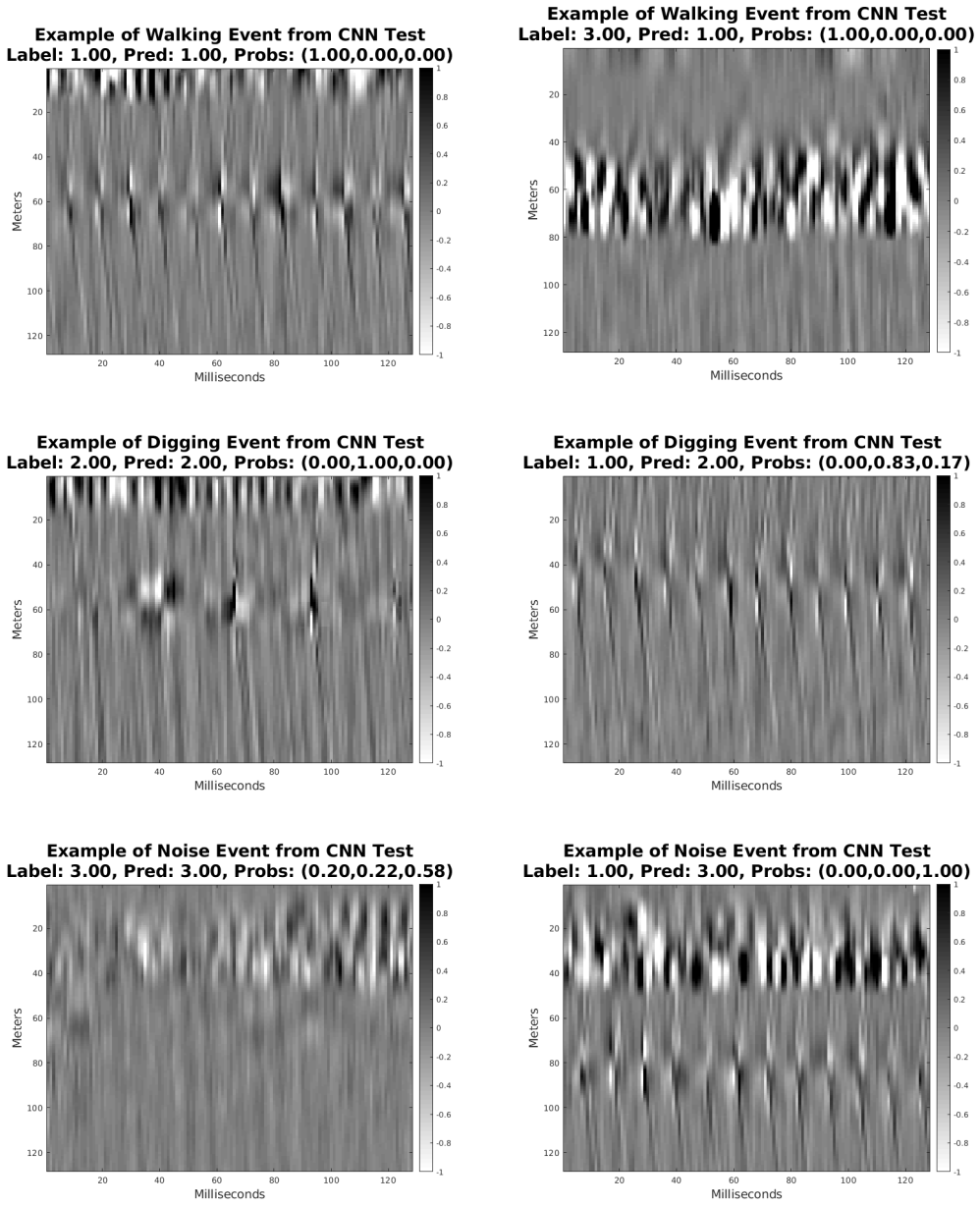

FIG. 18: Images of walking (top row), digging (middle row), and noise (bottom row) events from the 4kHz PRF acquired data when decimated by 50 shots. The left column shows accurately defined events and the right column displays inaccurately defined events.

decreased. The success of the CNN depends heavily on how similar the data is to the training images from which the CNN was taught.

The results of this experiment raise the question of feature-based image registration for data acquired using a distributed acoustic senors. Image registration is an important part of image processing or image recognition. It entails aligning two or more images into one coordinate system when the images are taken at different times, from different sensors, or from different viewpoints (Islam and Kabir, 2013). For example, features for a photograph of a person would include scaling and skewing the image as well as shifting, flipping, and rotating the person inside the image. For DAS-acquired data, the general shape of an event is hyperbolic. The features for image registration of such data would involve different methods for changing the shape or location of the hyperbolic response. Some clear attributes include scaling and shifting the response in the data. From Hardeman-Vooys and Lamoureux (2018), we know that the velocity of the source changes the shape of the hyperbolic response as well as the size of the gauge length applied to the data. The experiment in this section suggests that the pulse-repetition-frequency also affects the shape of the response. Not only does information about the signal determine features for image registration, but also how the distributed acoustic sensor collects the data, such as the gauge length and PRF employed. As such, all these aspects must be considered when developing image recognition methods for data collected using a DAS system.

FUTURE WORK

The experiments in this paper and Hardeman et al. (2017) applied convolutional neural networks to DAS data shown in the space-time domain. We have shown throughout this section that DAS acquired data can differ enough to cause a CNN to struggle with image-recognition. Changing the domain of the DAS data to depend on frequency may allow the convolutional neural network to be more successful. Given the opportunity, this is definitely a route to explore with regards to using convolutional neural networks for image recognition in DAS data.

CONCLUSIONS

We began with a short study of neural networks with an emphasis on convolutional neural networks for the purposes of image-recognition. We considered an example where we trained a CNN to distinguish between walking, digging, and noise. Once trained, we saw that the CNN had approximately 80% accuracy when identifying events. We applied the CNN to two data sets containing walking and digging events: one collected using a pulse repetition frequency of 4kHz and one using 6kHz. Given the relative accuracy between the two data sets, we noted that some equalizer is necessary for DAS data before image-recognition using convolutional neural networks can be generally successfully, instead of successful on isolated cases. Furthermore, there is much to consider with regards to image registration when dealing with data acquired using a distributed acoustic sensor. We must also consider the effects of how the interrogator collects the data on the way events look when conducting image-recognition experiments on the data.

ACKNOWLEDGMENTS

We thank the sponsors of CREWES for continued support. This work was funded by CREWES industrial sponsors, and NSERC (Natural Science and Engineering Research Council of Canada) through the grant CRDPJ 461179-13, the grant CRDPJ 522863-17, and the Discovery grant RGPIN-2015-06038 of the third author.

REFERENCES

Adcock, B., and Dexter, N., 2019, Lecture I - Introduction to machine learning with neural networks, PIMS CRG Summer School: Deep Learning for Computational Mathematics.

Bruns, A., 2004, Fourier-, Hilbert- and wavelet-based signal analysis: are they really different approaches?: Journal of Neuroscience Methods, **137**, 321–332.

Chang, T., and Kuo, C.-C. J., 1993, Texture analysis and classification with tree-structured wavelet transforms: IEEE Transactions on Image Processing, **2**, 429–441.

Hardeman, H. K., McDonald, M., and Lamoureux, M. P., 2017, Scale-invariant image-recognition using convolutional neural networks and wavelet analysis: CREWES Research Report, **29**, 1–17.

Hardeman-Vooys, H. K., and Lamoureux, M. P., 2018, Analytic models of distributed acoustic sensing data for straight and helical fibre: CREWES Research Report, **30**, 1–17.

Islam, M., and Kabir, M., 2013, A new feature-based image registration algorithm: Computer Technology and Application, **4**, 79–84.

Ng, A., Ngiam, J., Foo, C. Y., Mai, Y., Suen, C., Coates, A., Maas, A., Hannun, A., Huval, B., Wang, T., and Tandon, S., 2013, UFLDL: Welcome to the deep learning tutorial, `http://ufldl.stanford.edu/tutorial/`.

Schmidt, M., 2005, minfunc, `https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html`.

Wu, B., 1992, An introduction to neural networks and their applications in manufacturing: Journal of Intelligent Manufacturing, **3**, 391–403.

Yang, B., 2014, Bin Yang, `https://github.com/byangderek`.