Pipeline joint identification using neural networks

Michael Byington[1], Anouk van Pol[1], John van Pol[1]

[1]Ingu Solutions Inc

Organized by

Proceedings of the 2023 **Pipeline Technology Conference** (ISSN 2510-6716).

www.pipeline-conference.com/conferences

1 ABSTRACT

Conventional ILI tools use odometer wheels to determine the location of identified defects. In addition, typically above ground markers (AGMs) are used to confirm and potentially correct for odometer wheel slippage. Free-floating unconventional ILI tools use information from a variety of sensors to accurately locate defects. Accurately identifying joints is a prerequisite for localization and automatic identification is key for a cost-effective inspection.

This work focuses on automating the joint identification process with a neural network. We will present deep learning strategies for discrete feature identification and segmentation in time series data, how those strategies are increasing data processing efficiency, current accuracy and limitations, and normalization strategies for data from multiple sensors.

2 INTRODUCTION

In 2016, a pipeline failure in North America resulted in the uncontrolled release of 2,000 metric tons of hydrocarbons (both liquid and gas) [1]. Pipeline failures of this magnitude cost operators billions of dollars in fines and clean up costs. To reduce the risk of failures, the industry relies on inspection tools. Magnetic flux leakage (MFL) and ultrasound devices have been the primary tools used for pipeline integrity management since the 1980s. However, due to tight bends, non-circular valves, diameter changes or unknown geometry [1,2], approximately 70% of US gas lines built before modern inline inspection (ILI) was a viable technology are considered unpiggable [3]. Industry surveys from 2012 indicate that 40% of gas pipelines in North America are unpiggable [4]. In response to this issue, more recent technologies have been developed since the early 2000s to address unpiggable lines [5,6].

Regardless of the technology, accurately determining location within the pipe has always been a challenge. Traditional ILI tools use odometer wheels but slippage remains an issue, particularly in lines with heavy deposits or rough surfaces [7]. To minimize inaccuracies in distance, external above ground markers (AGMs) can be used. AGMs come in magnetic, acoustic, or geophone array varieties [8,9,10]; however, they are expensive and often cannot be used in crowded urban areas due to their limited accuracy depending on the depth of the

pipeline. Multi-sensor free-floating devices provide an alternative method for solving the localization problem with its own set of challenges and strengths. Welds identified from magnetometer data are a primary component of the conversion from measurement time to measurement distance [11]. To make these tools more cost effective with higher reliability and short analysis times, it is necessary to automate the joint identification process. This paper will discuss how deep learning strategies can be used for discrete feature identification and segmentation in time series data, how these strategies are increasing data processing efficiency, current accuracy and limitations, and normalization strategies for data from multiple sensors. Specifically, we will focus on the use of a convolutional neural network called UNet for this purpose.

3 ONE DIMENSIONAL SEGMENTATION WITH UNET-STYLE NETWORK

Joints morphology, as detected by residual magnetometry, is highly varied. Some pipeline joints can be found with a simple peak search, while others appear as subtle increases in oscillation frequency. Others are so subtle that they can only be determined by considering the average joint spacing in the surrounding area. As joint identification lacks precise articulatable attributes which might lend it to imperative programming, a neural network (NN) is the most promising solution. To develop an automated joint detection system, this is conceptualized as a 1D segmentation problem. Labels are applied to each data point (joint or no-joint), and a NN is designed to classify each point. The network takes four magnetic vectors as inputs (m_x, m_y, m_z, and m_t where $m_t = \left(m_x^2 + m_y^2 + m_z^2\right)^{0.5}$) and returns two y_1, y_2 values. The joint likelihood vs position function is given by a softmax $p = -\log\left(\frac{e^{y_2}}{e^{y_1}+e^{y_2}}\right)$. See Figure 1.

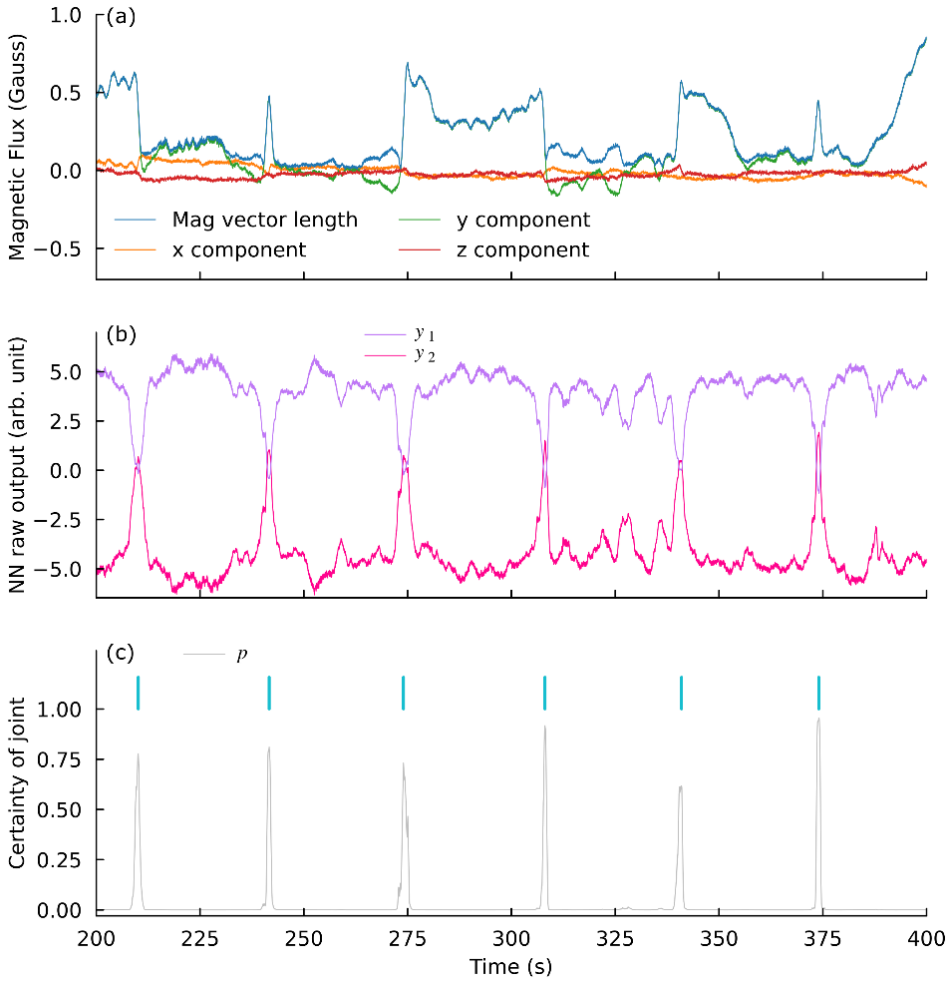

Figure 1 (a) Inputs to the neural network are not normalized due to the magnitude of the values and the absence of a natural normalization reference. (b) The raw output of the neural network is displayed in arbitrary units. (c) The softmax of the neural network output vectors from (b) provides a measure of certainty for each joint detected, with a peak search producing a list of joints (shown in cyan).

Since the seminal paper on UNet was published in 2015 [13], fully convolutional neural networks have become the de facto standard for segmentation tasks. Thousands of variations and applications of UNet have been documented since then, with 54,960 citations of the original paper as of 2023-01-11. The original UNet was built to label pixels in microscope images as cells or background. Our application of UNet necessitates several modifications; firstly, due to our 1D data instead of 2D data, the receptive fields must be increased to account for the increased distance between data points. Secondly, the precision required for joint identification is less than that for medical image segmentation, thus we do not need output arrays to be as

large as input arrays. Lastly, since our data is pipeline magnetics rather than micrographs of cells, appropriate data augmentations for training are not identical.

3.1.1 Architecture and receptive fields for 1D data

Expanding the receptive field to encompass multiple joints allows the neural network to replicate the contextual understanding that manual joint identification employs. It is possible that what may be interpreted as a joint in one pipeline's magnetic signature may simply be a normal fluctuation within a pipe segment on another line. When manually identifying joints, we look for patterns of signatures with similar distances between them. For the neural network to accurately imitate this process, the receptive field of its deepest neurons must encompass multiple joints. This insight is essential for adapting UNet to 1D segmentation. The 572x572 square pixel images of the original UNet described in [13] contain 327,184 data points per image, none of which are more than 571 steps away from any other. With a magnetometer data frequency of 200 Hz and 250 second snapshots, our input data is 51,840 data points long; however, the maximum distance between these points is 51,839 rather than 571. By transitioning from 2D to 1D, we can reduce our total data input by 85%, while simultaneously increasing the maximum distance between points by a factor of 90. As information traverses down the encoder side of the UNet, a broader receptive field is necessary to capture context from a wider area and make accurate classifications for each point.

The receptive field of a neuron in a NN of convolutional and pooling layers [14] is given by

$$r_0 = \sum_{l=1}^{L} \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right) + 1$$

where k_l is the kernel at layer l and s_i is the stride of layer i. The original UNet featured a receptive field of 140x140 pixels for the first 14 layers of the encoder. By augmenting the stride of the initial convolution in each pair to 2 and increasing the kernel and stride of the max pooling layers to 3, we can expand this receptive field from 140 to 10,367. This expansion enables the encoder to accurately contextualize magnetometer data within its respective line.

On the decoder side of the network, we used transpose convolutions to reverse the compressions of the max pool and convolutional layers with stride 2. By selecting an input array length evenly divisible by the strides of all the encoder layers, we were able to design a network which

required no cropping before appending encoder layers into the decoder side of the network. This yielded a slight enhancement in training performance and imparted an aesthetic quality to the network design. See Figure 2.

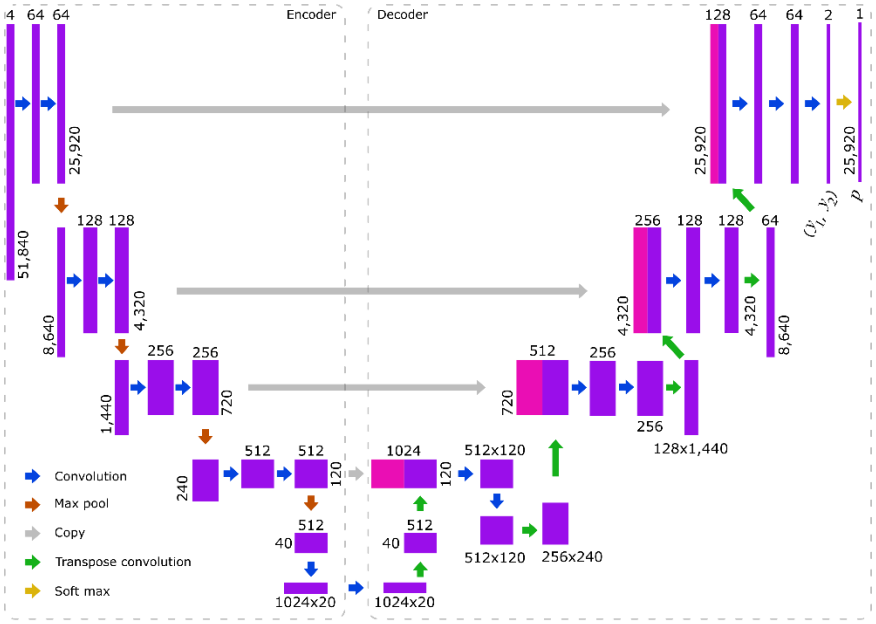

Figure 2 Architecture of fully convolutional UNet for 1D segmentation. Rectangular dimensions are not proportional to length because the length of the tensors changes by a factor of 2,600 as it moves through the network. All convolutional layers have a kernel of 3. On the encoder side of the network, the stride is 2 or 1 for the first or second in each pair, respectively. On the decoder side, the stride is 1 for all convolutional layers. All transpose convolutional layers have a kernel of 3; the stride is 2 or 3 for the first or second in each pair respectively. Max pool layers have a kernel and stride of 3. ReLU activations functions follow each convolutional or transpose convolutional layer.

3.2 DATASET AND AUGMENTATION

Despite much attention, the question of how much data a NN requires for training remains an active research area [15,16]. Generally, estimates range between 10 and 1000 times the number of parameters in the model. In our case, we have a NN with 12.6 million parameters and only ~100,000 examples of joints. These joint examples are not even independent training samples since multiple joints are included in each training sample. Fortunately, we have access to many data augmentations which enabled us to create 3 million training samples. Additionally, even

with these augmentations, these NNs were able to converge and generalize with two orders of magnitude less data than would typically be expected for models of this size. Consequently, we anticipate significant improvements as more data is incorporated into the training library.

To generate an augmented data sample

- Select random survey
- Select random subset of survey 37.5k to 70k data points long
- Resample by linear interpolation to 50k data points (stretch or compress in x axis)
- Multiply each component by random scaler value from 0.7 to 1.33 (stretch or compress in y axis)
- With 50% probability, reverse the sample (mirror in x axis)
- With 50% probability, multiply x, y, and z components by -1 (mirror in y axis)
- Randomly reorder x, y, z components. Because measurements are taken free-floating, the orientation of the sensor is random so random reordering of components is a valid augmentation strategy.

3.3 LOSS FUNCTIONS AND GROUND TRUTH

The training data y values are represented as a discrete set of points, while the output is a segmentation output. To generate a ground truth tensor suitable for a segmentation algorithm, we define the ground truth as "joint" within a window of 50 data points and "not joint" elsewhere. This initial approximation achieved near-human accuracy without attempting to adjust for joint width variability depending on sensor velocity.

This network was trained using a standard cross-entropy loss function without applying any different weights to the data points. The original UNet was focused on the edges of cells, and by weighting the loss function higher at these boundaries, it was able to accurately localize predictions. In this application, however, it is more important to learn the classes of the data points in the center of the joint region than to precisely identify the edges of the target segmentation. Therefore, we uniformly weighted the loss function across all data points.

4 MULTISENSOR DATA APPROACH

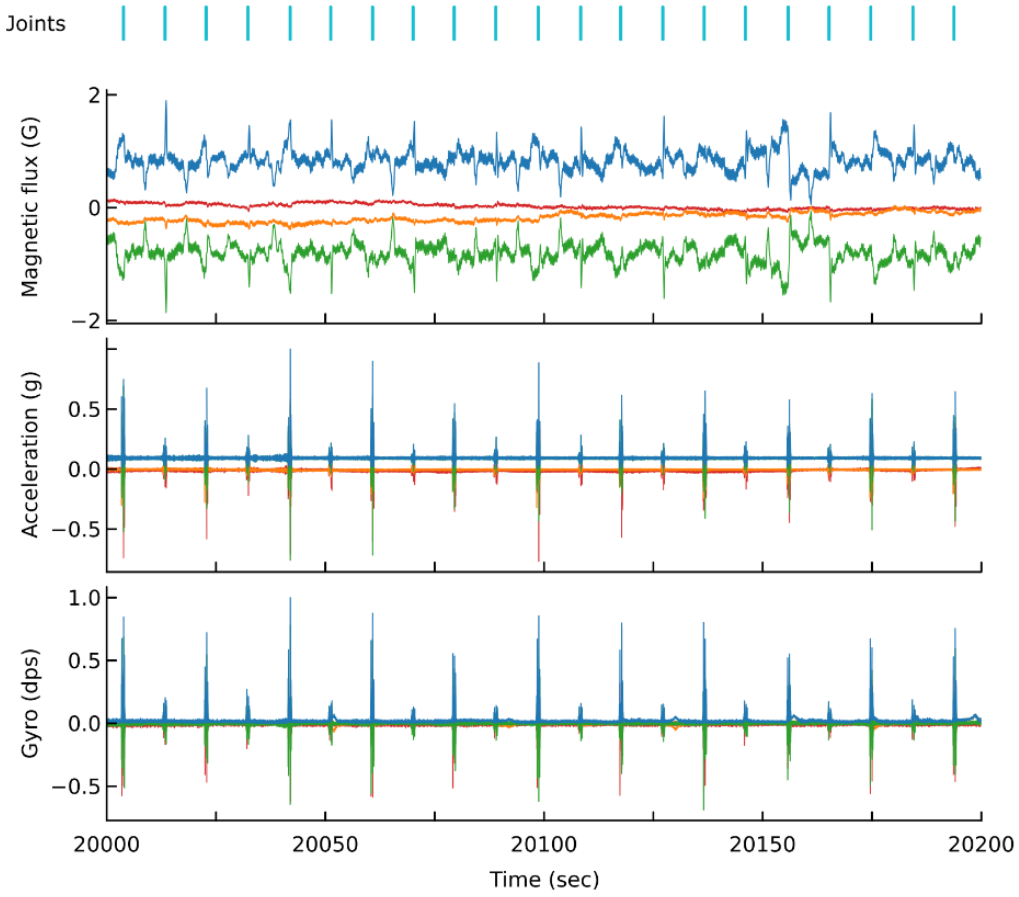

Figure 3 When the sensor is run on the back of a cleaning pig, joints appear in the magnetometer, gyroscope, and accelerometer data. The network accepts 12 input channels (x, y, z, and total for each of the three sensors; x: red, y: green, z: orange, and total: blue). The four acceleration or gyroscope channels are normalized by the max of the total acceleration or gyroscope signal, respectively.

Although joint signatures are visible in the magnetometer data in >95% of pipelines, when affixed to the back of a cleaning pig the combination of accelerometer, gyroscope, and magnetometer data allows for cross validation among different sensors and higher accuracy. For instance, if the pig has a relatively small diameter compared to the pipe or if the welds do not traverse completely through the pipe wall, joints may be ambiguous in the acceleration/gyroscopic data. To ensure that this data is correctly incorporated into our neural network, we normalize the gyroscopic and acceleration data by the maximum vector length in each 200 second window. See Figure 3.

For the pigged-only neural networks, we can randomly switch the x and z axes, but the y axis is fixed in the field data (parallel to the direction of travel within the pipe). Other data augmentations are valid but must be applied equally across all sensors (mirrored the same, stretched in x the same, etc).

5 RESULTS AND FUTURE WORK

We have employed a range of neural networks in this style, utilizing 50-150k training cycles, learning rates from 0.01 to 0.3, and minibatches of 20-50 samples. In validation data sets, these networks have yielded false positive rates of 1-2% and false negative rates of 3-5%, with human level performance as measured by the discrepancy between manual labelers being 0.1 to 2%. Despite training networks with 10 million parameters on less than 100,000 examples (prior to augmentation), our networks are approaching human level performance. We anticipate that as our data library grows, the neural networks will surpass human level performance.

6 REFERENCES

[1] M. Xie, and Z. Tian, "A review on pipeline integrity management utilizing in-line inspection data," Engineering Failure Analysis, vol. 92, pp. 222—239, 2018.

[2] F. Varela, M. Yongjun Tan, and M. Forsyth, "An overview of major methods for inspecting and monitoring external corrosion of on-shore transportation pipelines," Corrosion Engineering, Science and Technology, vol. 50, pp. 226—235, 2015.

[3] Interstate Natural Gas Association of America, "Report to the National Transportation Safety Board on Historical and Future Development of Advanced In-Line Inspection Platforms for Use in Gas Transmission Pipelines," 2012.

[4] J. Tiratsoo, "Ultimate Guide to Unpiggable Pipelines," 2013.

[5] R. Fletcher, and M. Chandrasekaran, "SmartBall: a new approach in pipeline leak detection," International Pipeline Conference, vol. 48586, pp. 117—133, 2008.

[6] J. Smith, A. Van Pol, D. Ham, and J. Van Pol, "Leak detection and prevention using free-floating in-line sensors," Pipeline Pigging and Integrity Management, 2019.

[7] R. Bickerstaff, M. Vaughn, G. Stoker, M. Hassard, and M. Garrett, "Review of sensor technologies for in-line inspection of natural gas pipelines," Sandia National Laboratories, 2002.

[8] X. Wu, A. Xu, Y. Xiao, B. Zhou, G. Wang, and R. Zeng, "Research on Above Ground Marker System of pipeline Internal Inspection Instrument Based on geophone array," 2010 6th International Conference On Wireless Communications Networking and Mobile Computing, pp. 1—4, 2010.

[9] Y. Li, D. Wang, and L. Sun, "A novel algorithm for acoustic above ground marking based on function fitting," Measurement, vol. 46, pp. 2341—2347, 2013.

[10] L. Sun, Y. Li, Y. Wu, J. Guo, and Z. Li, "Establishment of theoretical model of magnetic dipole for ground marking system," 2017 29th Chinese Control And Decision Conference (CCDC), pp. 6134—6138, 2017.

[11] M. Kindree, S. Campbell, A. Van Pol, and J. Van Pol, "Defect localization using free-floating unconventional ILI tools without AGMs," Pipeline Pigging and Integrity Management, 2022.

[12] Z. Shand, A. Van Pol, and J. Van Pol, "Pipers; an inline screening tool for unpiggable pipelines," Unpiggable Pipeline Solutions Forum, 2019.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," Medical Image Computing and Computer-Assisted Intervention (MICCAI), vol. 9351, pp. 234—241, 2015.

[14] A. Araujo, W. Norris, and J. Sim, "Computing receptive fields of convolutional neural networks," Distill, vol. 4, pp. e21, 2019.

[15] A. Alwosheel, S. Van Cranenburgh, and C. G. Chorus, "Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis," Journal of Choice Modelling, vol. 28, pp. 167—182, 2018.

[16] T. Oyedare, and J. J. Park, "Estimating the required training dataset size for transmitter classification using deep learning," 2019 IEEE International Symposium On Dynamic Spectrum Access Networks, pp. 1—10, 2019.