IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 23, 2020, accepted November 20, 2020, date of publication December 4, 2020,
date of current version December 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3042337

EchoBERT: A Transformer-Based Approach for Behavior Detection in Echograms

HÅKON MÅLØY
Department of Computer Science, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway
e-mail: hakon.maloy@ntnu.no

This work was supported in part by the Center for Research-Based Innovation in Aquaculture Technology, Exposed Aquaculture Operations under Grant RCN 237790 and in part by the Project Echofeeding under Grant RCN 267815.

ABSTRACT Monitoring fish welfare has become increasingly important for salmon farmers. Current approaches require manual labor and physical inspection or interpretation of video. Echo sounders make real-time monitoring of the entire fish population over time possible. However, current approaches for automatic interpretation of echograms mainly focus on species classification and therefore fail to appropriately encode the spatiotemporal properties contained within the data. Other approaches are primarily aimed at the feeding process and require a human-in-the-loop. Transformer-based approaches have been shown to better handle long sequences than Long Short-Term Memory networks in recent Natural Language Processing research. We therefore introduce EchoBERT - Echo Bidirectional Encoder Representation Transformer, a transformer-based approach for behavior detection in farmed Atlantic salmon (*Salmo salar, Salmonidae*), using the spatiotemporal properties contained in echograms. The model interprets the spatiotemporal dynamics of echograms through attention mechanisms to classify fish behavior. We compare EchoBERT to a traditional sequence modeling approach on the task of detecting behavior indicative of pancreas disease in a six-fold cross-validation study using data from 6 distinct farming cages. We show that EchoBERT shows a strong correlation between model predictions and true labels, indicated by a Matthew's Correlation Coefficient score of 0.694 ± 0.178 using an ensemble approach, compared to 0.626 ± 0.084 for traditional sequence models. We also find that EchoBERT is capable of detecting disease indicators over a month prior to detection using standard procedures. Our results show that EchoBERT has high potential for automatic behavior detection through unintrusive methods suitable for applications in aquaculture. The source code is available at: https://gitlab.com/hakonma/echobert.

INDEX TERMS Atlantic salmon, behavior detection, deep learning, fish welfare, sequence modeling, transformer.

I. INTRODUCTION

Fish welfare is an increasingly important topic in the salmon (*Salmo salar, Salmonidae*) farming industry. With increased governmental focus on humane treatment of salmon, increasing fish welfare can reduce costs and increase production. Traditional methods for monitoring fish welfare include manually examining fish each week, surgically tagging a small portion of fish with health tags, and manual video monitoring. These methods require human expertise, are labor intensive and can introduce stress in the fish. They are also reliant on a representative sampling process, but since they do not easily facilitate sampling from the entire cage population, a good sampling will rarely happen. The use of non-intrusive, automatic approaches facilitating a more representative sampling processes could greatly increase fish welfare through detecting indicators of reduced welfare faster and more accurately, while reducing the need for manual labor and stress in the fish.

Echo sounders use acoustic back-scatter to visualize fish and are a common tool for fish discovery and monitoring for both fishermen and scientists. Recently, this technology has been adopted in salmon farming facilities, since it offers a non-intrusive approach to monitor the entire water column for activity. The acoustic data are acquired through the use of echo sounders, which send out an acoustic signal and measure the resulting echo generated from the sound scattered back to the transducer. This produces distinct echo signals as the

The associate editor coordinating the review of this manuscript and approving it for publication was Amjad Ali.

acoustic signal transitions from water to air when it hits the swim bladder of the fish. These signals are visualized through plotting the depth along the y-axis and time along the x-axis. Color is used to represent the volume of the measured echo. This process produces an echogram as shown in Fig. 1. Echo sounders therefore offer real-time visualizations of the entire fish population in a single visualization. However, current approaches for such extended monitoring of echograms are either fully manual or semi-automatic and are primarily used for feeding operations. They are therefore not well-suited for large-scale fish welfare monitoring. Furthermore, most automatic approaches for echogram analysis fail to account for the temporal aspects contained within the data. As echograms visualize time along the x-axis, they contain an historic account of the position of the fish and their activities. Such information could contain important behavioral cues indicating changes in fish health and general welfare.

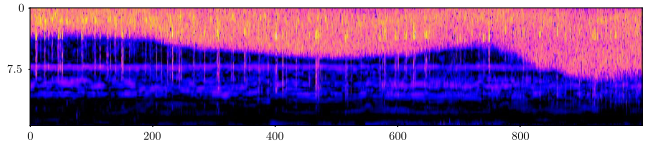


FIGURE 1. A sample echogram from one of our available cages. The y-axis shows the depth in meters, while the x-axis shows the time steps.

Recently, the field of deep learning has seen adoption in aquaculture. The use of deep neural networks has led to applications ranging from fish identification [13] and fish feeding behavior [12] to segmentation of blood defects in cod fillets [16]. Deep Neural Networks are highly parameterized statistical models used in combination with learning algorithms to approximate the functions underlying the data used to train the model. This has led such models to become the state-of-the-art in a many fields, including Computer Vision (CV) and Natural Language Processing (NLP) [9].

In this work we propose EchoBERT, a novel transformer-based approach for general fish behavior classification from echograms as seen in Fig. 2. The model is based on recent research in NLP [3], [25], but is modified for the new domain. EchoBERT introduces a non-intrusive option for near real-time fish health monitoring using the raw data obtained from echo sounders. In contrast to [25] and [3] the echo domain does not contain a fixed-sized vocabulary from which the language is built. Instead each echo sequence is unique as fish position themselves dynamically; even bubbles in the water introduce noise. We therefore train EchoBERT using a novel vector-substitution pre-training technique to further enhance its robustness to the domain specific challenges produced by the dynamic environment and to enable it to better understand the behavior dynamics underlying the echograms. We validate the approach on the task of disease detection and compare EchoBERT to well-known sequence processing models trained using the same pre-training approach. We show that EchoBERT outperforms other models by a significant margin, indicating its effectiveness for automatic behavior detection in echograms.

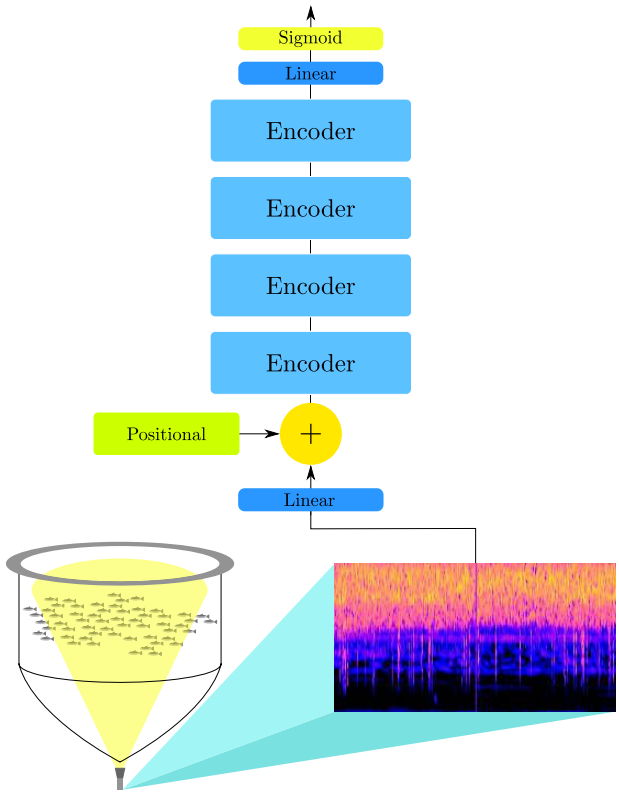


FIGURE 2. An overview of EchoBERT. The echogram is linearly transformed and positionally encoded before it is fed into four encoders. The model output is calculated using a single sigmoid activated unit.

Contributions:

1) We propose the application of sequence models to echogram behavior classification.
2) We propose a general approach for fish welfare classification making use of both the spatial and temporal information contained within echograms.
3) We introduce a novel pre-training technique making models more robust to the dynamical echo domain and more capable of understanding the underlying behavior producing echograms.

II. RELATED WORK

A. ECHO DATA APPLICATIONS

Echo data has widespread marine applications due to the acoustic properties of water and the accuracy and availability of echo-sounders. In [8] they show that discrimination between fish and macroinvertebrates (eg. Amphipoda) is possible using split-beam echo sounders at different frequencies. [2] compare discriminant function analysis to neural networks for classification of fish schools into three groups. The data was labeled through visual expert examination and prior knowledge. Five school descriptors were extracted from the echograms and were used for prediction. They found that both methods perform comparably well and achieve high classification accuracies. [4] use a random forest to distinguish ocean krill and mackerel icefish in echograms. They collected echograms using several frequencies and

applied convolutional kernels and a masking bitmap to clean the data. They extracted school parameters from echograms and labeled their data using trawl composition data from captured fish. The resulting dataset was then used to train a random forest classifier to distinguish ocean krill, mackerel, icefish, and a mixed class. Their approach resulted in a value of $k = 0.92$, with a 95% confidence interval ± 0.04. [19] used convolutional neural networks (CNNs) to detect schools of herring. They use regions of interest to extract salient echogram segments which are then fed into a CNN for classification. They compare several well known CNNs and a baseline Support Vector Machine (SVM) and achieve an F1-score of 0.82 using the DenseNet201 CNN. [6] use a CNN to identify fish species from echograms. They feed segmented images to a custom CNN trained from scratch to achieve F-scores above 0.9 for all species.

Our work separates itself from these approaches in two major ways. First, we use data obtained from salmon farms. Thus, there are several thousands of individual fish present in the echograms at all times. Second, we treat echograms as sequence data and not as images, thus our approach is able to leverage the spatiotemporal aspects of the data to better understand and fish behavior.

B. LANGUAGE MODELING

For machines to be able to do NLP, words from a vocabulary must be represented in a numerical form. The most common approach for converting words to numbers is word embeddings. This approach involves mapping words to vectors of real numbers which are then processed by programs producing models of the language.

Up until recently, language modeling tasks almost always included recurrent neural networks (RNNs). These networks include a loop-back connection that allows them to consider their previous state when generating their current state. They therefore naturally include sequential information in their processing pipeline in an intuitive way. However, vanilla RNNs quickly proved incapable of handling long sequences. Due to vanishing gradients in the backpropagation-through-time (BPTT) algorithm, vanilla RNNs have a hard time when input sequences become long as very little information about the beginning of the sequence is available to the RNN at the end of the sequence. To account for this, Long Short-Term Memory (LSTM) RNNs were introduced [7]. These networks expand on vanilla RNNs by including information gates, allowing LSTMs to learn what information to remember and what information to forget. However, the sequential nature of RNNs also means that their processing is not easily parallelizable. This makes them slow to train and resource inefficient compared to other neural network architectures.

The transformer [25], introduced a new approach to NLP. The transformer handles sequential data without the use of recurrent connections. Instead, it relies purely on an attention mechanism. The attention mechanism computes an attention score for the entire input sequence. It then weights which parts of the sequence to pay attention to and which to discard by applying this score to the sequence. This approach is highly parallelizable and allows the transformer to process the entire input without the need for BPTT. Transformer-based models have become the new go-to model for nearly all NLP tasks and have achieved state-of-the-art results on multiple benchmarks [17], [18], [25], [26]. In [3] the Bidirectional Encoder Representation Transformer (BERT) for language representation is introduced. This approach only uses the encoder portion of the original transformer, together with two novel pre-training tasks to pre-train deep bidirectional representations in an unsupervised setting. The resulting model can then be fine tuned to achieve (previously) state-of-the-art results on eleven natural language processing tasks.

EchoBERT separates itself from these language modeling tasks by not having a fixed vocabulary. Our input sequences contain continuous data rather than a vocabulary words. There are infinitely many ways fish can arrange themselves in the cages, and thus there are equally many resulting echograms. We therefore do not have a fixed corpus to create word embeddings from, leading us to use the raw echo vectors instead. We also introduce a novel pre-training technique to increase EchoBERT's understaining of the domain and the behavioral fish dynamics underlying the resulting echograms.

III. METHOD

A. DATASET

Our dataset consists of echo data collected at the Matre research station (61°N) of the Institute of Marine Research (IMR), Norway [11]. Six square sea cages (12 × 12 m and 15 m depth; approximately 2000 m^3) were used. The fish's vertical distribution and density were observed continuously by a PC-based echo integration system (CageEye MK IV, software version 1.1.1., CageEye AS, Steinkjer, Norway) connected to an upward facing transducer which multiplexes between 50 kHz (42° acoustic beam angle) and 200 kHz (14° beam angle). However, only the 50 kHz data was used. Echo intensity, which is proportional to fish density, was integrated by the echo integration system at 7.5 cm depth intervals using 10x oversampling giving a 75 cm moving average. All six cages were respectively monitored by one single CageEye transducer positioned approximately 18 m depth (below the cage bottom) and directed towards the cage center, as seen in Fig. 3. This resulted in a blind zone in the volume between the water surface and the sphere-shaped end of the echo beam. Since the transducer was facing upwards towards the water surface, the reflections were not impacted by sediments and the frequency used is not sensitive to turbidity. Although the size of the fish is related to the echo data produced by the system, the echo sounder used is not able to distinguish between individual fish. It therefore only provides a crude assessment of the biomass in the cage.

During data collection all six cages became infected with Pancreas Disease (PD) [10], [15], [24]. The first signs were detected in February 2019, when seven out of twenty clinically sampled fish were shown to be infected with the disease through clinical evaluation. A drastic decrease in appetite

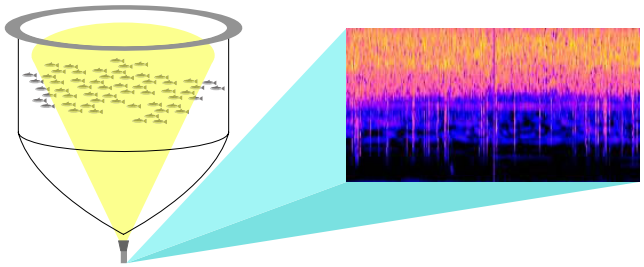


FIGURE 3. An upward facing transducer is placed at the bottom of the cage, producing an echogram. The volume covered by the yellow beam results in one vertical vector in the echogram.

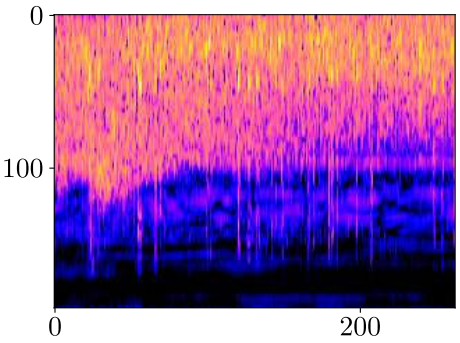


FIGURE 4. A typical time slice from our dataset. Each time slice consists of 256 time steps along the x-axis. The vertical resolution for each time step is 192 × 7.5 cm.

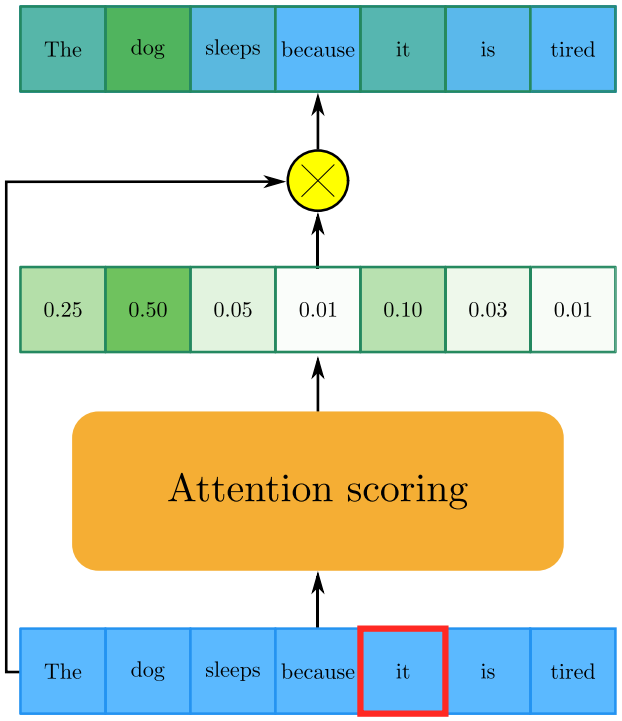


FIGURE 5. Attention score being calculated for the word 'it'. The attention score is highest around 'The' and 'dog' meaning that the attention mechanism strongly relates 'it' with these two words.

across all cages was measured two months later, on the 25th of April. This is a typical indicator of PD infection in the industry. All cages were therefore clinically assessed the following week, with all samples showing signs of PD. Data collection was terminated after another month due to emergency slaughtering of all cages.

The dataset was created by extracting raw echo data to .csv files using a CageEye conversion tool. We removed the sea surface and the lower portion of the echo data by cutting off the upper and lower part of the vertical measurement. This resulted in each measurement having a vertical resolution of $192 \times 7.5\,\text{cm} = 14.4\,\text{m}$. One file per day was created, containing continuous time-series data. The entire dataset contains data from 07.06.2018 to 20.05.2019 for five cages, and from 07.11.2018 to 20.05.2019 for cage 2. The missing data from cage 2 was due to data corruption. For each cage all dates were concatenated and split into 256-step time slices, where each step corresponds to one ping from the echo sounder. The resulting dataset is an $n \times 192 \times 256$ matrix for each cage, where n is the number of time slices for that cage and 192 is the vertical resolution of the sensor below the water surface. Time is along the x-axis and depth along the y-axis, as shown in Fig. 4. This makes each time slice a historical account of 256 echo sounder pings and enables sequence models to treat the time slices as sequences, processing each ping as a time step in the sequence.

B. DATASET SPLIT

Data from within the same cage looks very similar for time slices near each other in time. Generating a test set from the same cages used for training could therefore lead to the models overfitting to the cages rather than learning a generic understanding of what is happening in the echograms. To account for this, we perform k-fold cross-validation with $k = 6$ folds. Each fold uses one of the cages as its test dataset, while training and validation datasets sample random time slices from the remaining cages.

C. EchoBERT

The EchoBERT model uses the transformer approach presented in BERT [3]. We use only the encoder part of the transformer since the task does not involve translation. We use a linear projection layer to linearly transform the input into a sequence of vectors, each of length d_{model} (where d_{model} is normally 256). Following the linear layer is a positional encoding as described in [25]. This enables the model to understand the order of the input sequence and enhances its capabilities to model temporal dependencies. The resulting sequence is fed into a stack consisting of four encoder modules stacked on top of each other, regularized by dropouts [22] between every module. Finally, a classification head consisting of a single sigmoid-activated unit, takes the output of the encoder stack as input and produces a sequence classification output.

1) ENCODER MODULE

Each EchoBERT encoder module consists of a self-attention layer and a feed-forward network. The self-attention layer calculates the attention over its input using an attention

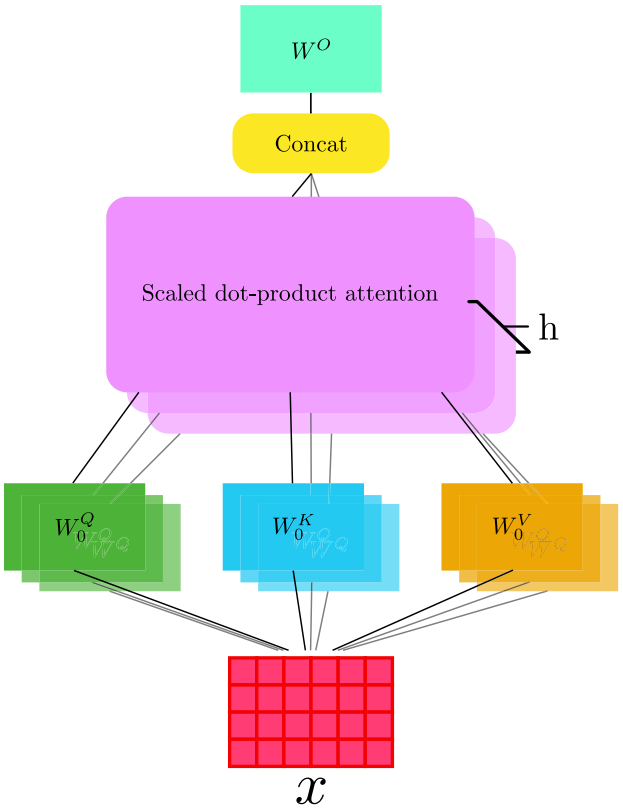


FIGURE 6. Multi-head attention processing the input x in h parallel tracks using h heads before combining the tracks into a single attention calculation.

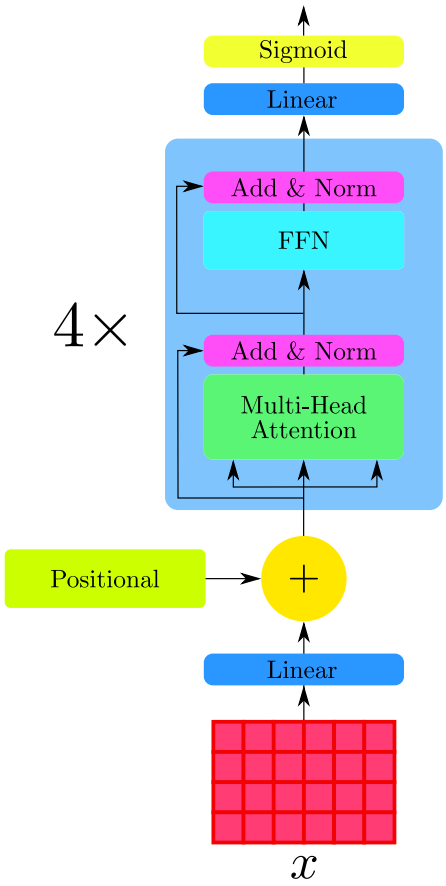


FIGURE 7. EchoBERT consists of the encoder part of a transformer. It uses 4 encoder modules before passing the output through a sigmoid activated decision head.

mechanism. The attention mechanism computes an attention score for the entire input sequence. It then weights which parts of the sequence to pay attention to and which to discard by applying this score to the sequence, as seen in Fig. 5.

To compute an attention score, the input sequence x is linearly transformed into a Query-matrix Q, a Key-matrix K and a Value-matrix V, using a different weight matrix for each of the transformed matrices. The resulting matrices are then used to compute a normalized score by taking the dot product of the query and key matrices, dividing it by a scaling factor $\sqrt{d_k}$, where d_k is the dimensionality of the Key, and applying a softmax function as seen in eq. 1. The resulting attention score is then multiplied with the Value-matrix to compute the attention.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

The self-attention layer is configured in a multi-head attention configuration to improve its performance. This configuration uses a different set of weights for each head, resulting in different Q, K and V matrices for every head. This allows each attention head to learn different attention scores, making the multi-head approach capable of attending to several features in parallel. The multi-head attention scores are then concatenated and linearly transformed to compute the final attention score, as expressed in eq 2 and diagrammed in Fig. 6. To keep computational costs similar to that of a single-head approach, the dimensionality of the query and value matrices is reduced to $d_Q = d_V = \frac{d_{model}}{\text{h}}$, where h is the number of heads used. In EchoBERT we use h $= 16$ heads per encoder module and set $d_{model} = 256$ using the first linear projection layer, resulting in $d_Q = d_V = 16$.

$$\begin{aligned} MultiHead(Q, K, V) &= [head_1, ..., head_n]W^O, \text{ where} \\ head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V), \\ W_i^K &\in \mathbb{R}^{d_K \times d_{model}}, \\ W_i^Q &\in \mathbb{R}^{d_K \times d_{model}}, \\ W_i^V &\in \mathbb{R}^{d_V \times d_{model}}, \text{ and} \\ W^O &\in \mathbb{R}^{\text{h}d_V \times d_{model}}. \end{aligned} \tag{2}$$

Between each sub-layer in the encoder module there is also a residual connection and a layer normalization. The residual connection adds a shortcut through the network, providing an alternative route for the gradient to flow. This allows deeper networks to be built since the error signal can always reach the lower levels in the network through the residual connections. The full EchoBERT model is shown in Fig. 7.

2) TRADITIONAL SEQUENCE MODEL

We compare EchoBERT to a traditional sequence model to show how the long-term dependency capabilities of

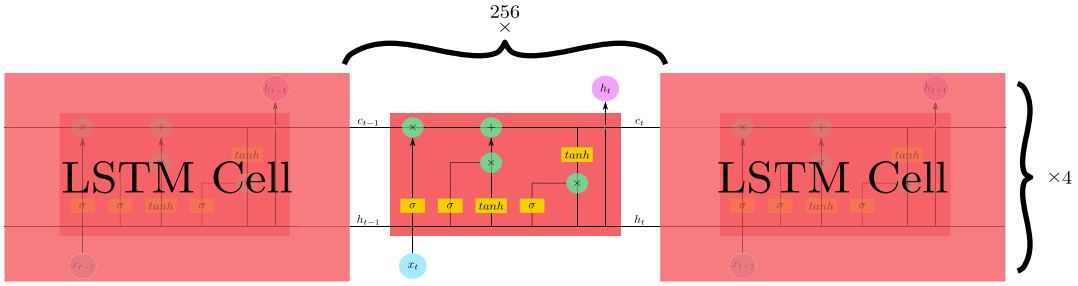


FIGURE 8. The traditional sequence model consists of four LSTM layers, each with 256 cells. The σ represent sigmoid activations, *tanh* represent tangens hyperbolicus activations, c_{t-1} represent previous cell state, h_{t-1} represents the previous hidden state and x_t represents current time step input.

EchoBERT increase model performance. Both models have a comparable number of parameters per layer, making models with the same number of layers directly comparable.

For our traditional sequence model, we use an LSTM recurrent network. It consists of four layers, each with 256 cells. The final classification head is, again, a linear layer taking the output of the LSTM as input and outputting a single sigmoid-activated classification. We regularize the model using dropout with a probability of 0.1 between every layer. The model is visualized in Fig. 8.

D. PRE-TRAINING TASK

To increase the model's understanding of the echo data dynamics, we adapt the *next-sequence prediction* task from [3] and introduce a novel substituted-vector prediction task as a unsupervised pre-training step. The use of unsupervised pre-training steps is a common approach in NLP to help models learn the underlying structure of the data before fine-tuning the model on the actual task [3], [17], [18]. In the pre-training step we create a synthetic dataset from our original dataset, where dataset splits are kept as stated in section III-B, but the classification task is changed.

1) TASK #1: NEXT TIME SLICE PREDICTION

For the model to understand long-term dependencies in echograms it is important that it understands how the fish school's position in one time slice affects future school positions. For example, if the school begins feeding behavior in one time slice, it is reasonable to expect it to continue this behavior in the next time slice as well. However, if the school is already feeding in one time slice, it could also be reasonable to expect it to stop feeding in the near future. To this end, we train a binary next-time-slice prediction task. For each pre-training example, two time slices (A and B) are concatenated along the time-axis. With a probability of 50%, time slice B is the actual successor to A, otherwise, B is a random time slice selected from the training dataset, as shown in Fig. 9. To give the model information about where B starts in the concatenated vector, a value of 1 is added to the input vectors of B through a time-slice embedding.

2) TASK #2: SUBSTITUTED VECTOR PREDICTION

To train a bidirectional representation in our model, we take inspiration from the Masked LM pre-training step introduced in [3]. This approach masks individual time steps within time slices through masking randomly chosen time-step-vectors in the original input. However, instead of masking our vectors using a '[MASK]' token, we directly

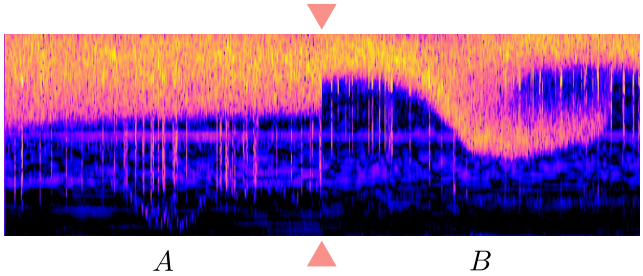
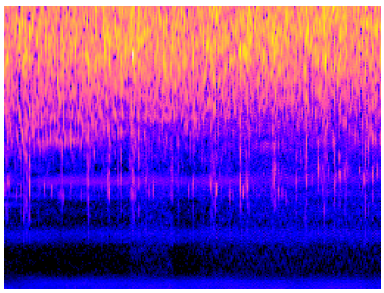
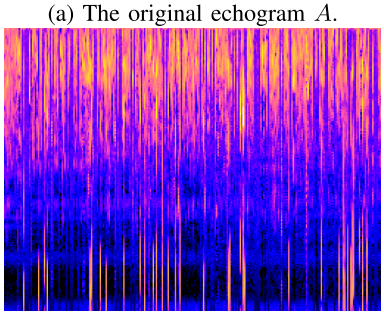


FIGURE 9. The input to the model. Time slices *A* and *B* are concatenated along the x-axis, where *B* is the actual successor to *A* with a probability of 50%. The arrows indicates where *A* ends and *B* begins. Here *B* is not the time slice naturally following *A*.



(a) The original echogram A.



(b) The masked echogram A'.

FIGURE 10. Masking time-step-vectors in the input. Vectors in the original echogram are masked with random vectors from the training data, resulting in a masked echogram. The task is to predict which vectors originate from the original echogram and which are masked.

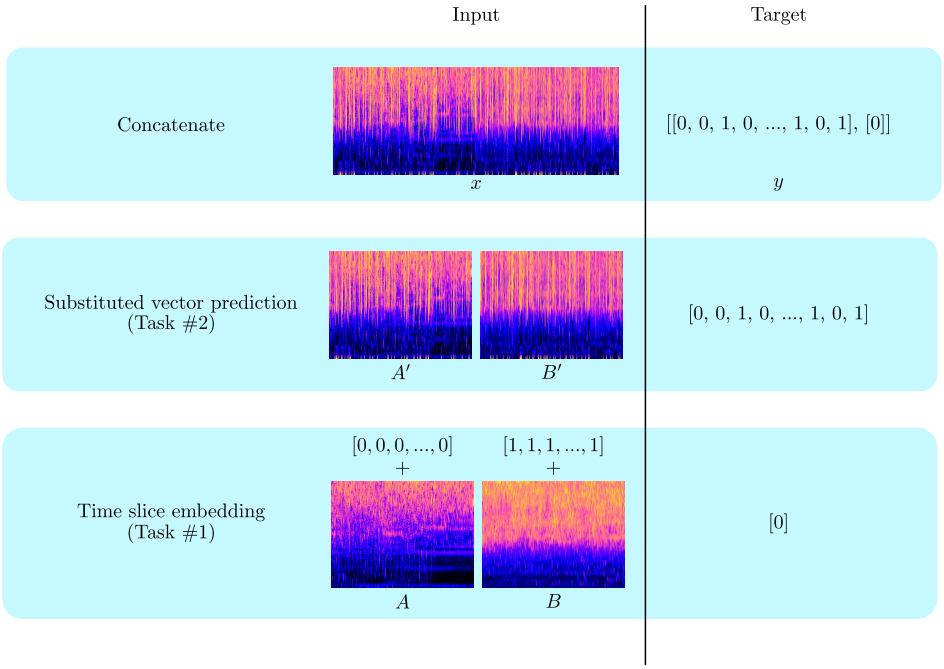


FIGURE 11. Input generation for the pre-training task. The *A* and *B* time slices are first time slice embedded before they are masked using the substituted vector masking and finally concatenated to form the input *x*. The corresponding prediction target is shown to the right of each step.

substitute 50% of the vectors in time slices A and B with random vectors sampled from our training dataset, as shown in Fig. 10. This results in the masked time slices A' and B'. In contrast to [3], we do not task the model with recreating the original vector that was substituted. Instead, we introduce a novel binary task where, for each time-step-vector in the time slice, the model must predict whether it is the true vector or a substitution. The reason for this is that the original vector is inevitably very similar to the prepending vector, making it possible for the model to simply copy that vector for a maximum score. Since we always substitute vectors with real vectors from within the dataset, there is no mismatch between the pre-training and fine-tuning. This allows us to obtain a bidirectional model that does not suffer from the drawbacks mentioned in [3]. Since our model predicts $real = 0$ or $substituted = 1$ for all vectors in the input, we do not expect an increase in pre-training steps to be required for the model to converge.

3) PRE-TRAINING PROCEDURE

To generate each pre-training input, we apply the time slice embedding to both A and B time slices. The resulting time slices are then masked using the substituted vector masking, producing A' and B', before A' and B' are concatenated to form the input x. The generation of input is shown in Fig. 11. The combined length of the input time slices is 512 time steps. For the pre-training task, two output heads are added to the model, one for each pre-training task. Both heads are outputting a sigmoid-activated vector and trained using binary cross-entropy loss. We train with a batch size of 45 for 30 epochs using a 1-cycle learning rate policy [21] with a max learning rate of 0.001. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and L2 weight decay of 0.01. The training loss is the sum of the mean substituted vector error and mean next-time slice error. The training is performed using the NTNU IDUN computing cluster [20]. The models are trained in parallel on 5 nodes, each consisting of two Intel Xeon cores and 2 NVIDIA Volta V100 GPU with 16 GB memory. Each pre-training takes approximately 1 hour to complete.

4) FINE-TUNING PROCEDURE

For the infection classification task, fine tuning requires replacing the two pre-training classification heads in the model with a single infection classification head. This head outputs a single classification for each input, giving the probability of an infected school using a sigmoid activation function. For the disease detection task B, time slices are always the time slice following A, and there is no vector substitution. Thus, the input for the disease detection task is A and B time slices concatenated along the x-axis. The label is a binary label is_{sick} or $is_{healthy}$ represented by 1 and 0 respectively. We train the model using binary cross-entropy loss, using the same approach and hyperparameters as described in section III-D3, but use a learning rate of 0.01. The training time is approximately 45 minutes for each of the six dataset folds using the same hardware as in section III-D3.

IV. EXPERIMENTS

A. DATASET DECISIONS

Although a sudden decrease in feeding response is one of the early signs of PD [15], it is reasonable to assume that the

TABLE 1. A Comparison of LSTM and EchoBERT Validation Losses for Each Omitted Cage (lower is better). The Loss was Calculated Using Binary Cross-Entropy Loss.

Model	Cage 1	Cage 2	Cage 3	Cage 4	Cage 5	Cage 6
LSTM	0.2724	0.2097	0.2537	0.2869	0.2188	0.2136
EchoBERT	**0.0978**	**0.1577**	**0.1179**	**0.1086**	**0.1023**	**0.1437**

majority of fish were actually infected before the decrease in feeding response was measured. Thus actual infection time is likely before 25.04.2019. In our experiments we therefore elected to set the infection date to 15.03.2019. However, an ablation study with two other dates was also performed in section IV-F3. As a result of this, the disease dataset contains relatively few examples of infected fish compared to healthy fish. Imbalanced datasets are a common problem in machine learning tasks, and there are several ways of dealing with them [23], including undersampling the majority class during training, which we have done. We also report our results using the Matthews Correlation Coefficient (MCC) score [14], which has been shown to be robust to imbalanced datasets [1]. The MCC score ranges from -1 to 1. A score of -1 indicates total disagreement between model output and target, while a score of 1 indicates a perfect match. A score of 0 is equivalent with random predictions. The MCC score is calculated as shown in eq. 3, where TP, FP, TN and FN are the True Positive, False Positive, True Negative and False Negative rates, respectively. When any of the sums in the denominator is zero, the denominator is instead set to one, resulting in an MCC score of 0.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (3)$$

B. MODEL PRE-TRAINING RESULTS

For the pre-training step, we train one model per fold in our dataset. This model is trained on all the training data except data from the excluded cage in a six-fold cross-validation setup. We report validation loss for both EchoBERT and the LSTM-based approach. The results can be seen in Table 1 and clearly show that EchoBERT outperforms the LSTM on every cage.

C. MODEL FINE-TUNING RESULTS

To get a fair evaluation of model architecture performance on the infection classification task, we train five identical models starting from the aforementioned pre-trained models. The classification head is randomly initialized for each of the five models, enabling us to show the general architecture performance rather than the performance of a particular model. We report the mean MCC score and standard deviation over all cross-validation folds in Table 2 and show the mean and standard deviation for each cage in Fig. 12. EchoBERT outperform the LSTM by a substantial margin on the average MCC score for the entire dataset. This indicates that the ability to handle longer time-sequences is key to model performance on our task.

TABLE 2. The Mean and Standard Deviation of the MCC Score Over all Cages. Five Runs Were Performed for Each Model Per Cage.

Model	Average MCC score
LSTM	0.626 ± 0.084
EchoBERT	$\mathbf{0.657} \pm 0.168$

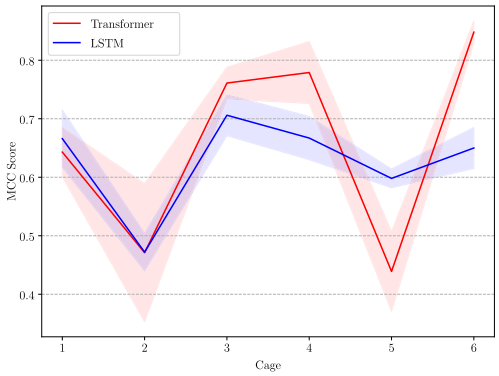


FIGURE 12. A comparison of EchoBERT and the LSTM. The mean and standard deviation of the MCC score are shown per cage.

D. ENSEMBLE APPROACH

Recent work has shown that ensembles of neural networks can outperform single models by a significant margin. Furthermore, networks achieving the same loss do not produce the same function approximation [5]. We therefore form an ensemble using the five fine-tuned EchoBERT models per cage. The ensemble is formed using the mean output over the five models to produce the ensemble output. We compare the ensemble to the individual models in a per-cage manner in Fig. 13 and Table 3.

TABLE 3. The Mean and Standard Deviation of the MCC Score Over All Cages. Five Runs Were Performed for Each Model Per Cage in the Individual Models Row, While the Ensemble Row Used an Ensemble of 5 Models, Taking the Mean of Their Outputs for Each Cage.

Model	Average MCC score
Individual Models	0.657 ± 0.168
Ensemble	$\mathbf{0.694} \pm 0.178$

It is clear that the ensemble performs as well as or better than the individual models on all cages. It also outperforms the best available model on cages 4 and 6, indicating that the ensemble approach is even more robust to differences between cages. These findings are further supported by the Receiver Operating Characteristics (ROC) curve in Fig. 14, the Precision-Recall curves in Fig. 15, and the normalized confusion matrices in Fig. 16. Here, we show a clear tendency

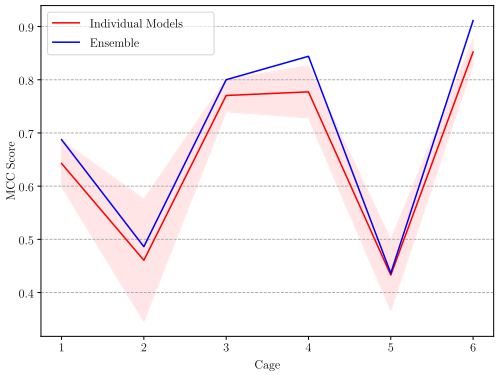


FIGURE 13. A comparison between the ensemble model and the individual models. The mean and average MCC score is presented for the individual models, while the direct score is presented for the ensemble.

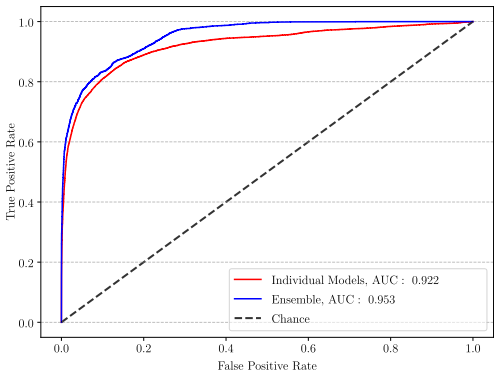


FIGURE 14. The Receiver Operating Characteristics (ROC) curve and the Area Under Curve (AUC) for the ensemble and the individual models.

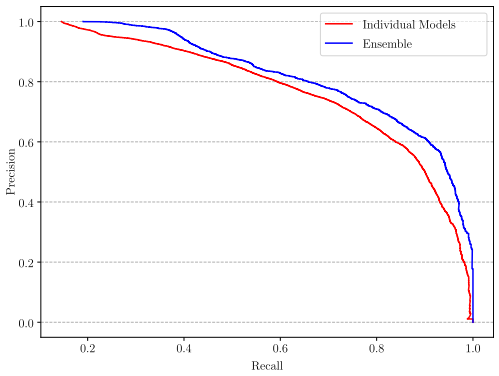


FIGURE 15. The Precision-Recall curves for the ensemble and the individual models.

for the increased performance of the ensemble approach through a significantly higher Area Under Curve (AUC) for the ensemble. The Precision-Recall curve is also higher at every point. In the normalized confusion matrices we have the true positive rate and the true negative rate on the diagonal, and again, the ensemble outperforms the individual models.

E. WHAT IS THE MODEL LOOKING AT?

Since all processing and categorization of behavior patterns happen inside the EchoBERT model it can be interesting to visualize the final attention activations in the encoder

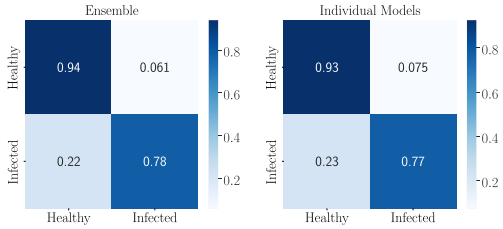


FIGURE 16. The normalized confusion matrices for the ensemble and the individual models.

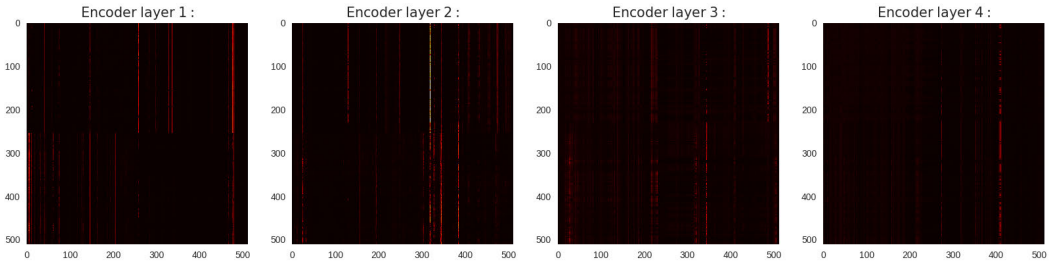
modules to see what the model is paying attention to at any given prediction. In Fig. 17 we show the attention activations for all four modules of two different EchoBERT models during testing on the 2nd and 6th cage. From the visualization it seems that later time-steps are heavier weighted during healthy classification, while time steps seem more evenly weighted during infection classification for the model tested on cage 6. For the model tested on cage 2 however, a pattern is not very clear. This could indicate that better performing models have learned better attention weights as the cage-6 models are generally much better performing than the cage-2 models. From the visualization it is also clear that some time-steps are much higher weighted by the models. This patterns is visible in both models and shows up as vertical stripes at the given time-step in the attention visualization. From manual inspection of the echograms at the time-steps highlighted by the models, nothing out of the ordinary seems to be happening. However, this could still indicate that the models view individual time-steps as particularly important and can pick out seemingly ordinary behavior and single it out as behavior indicating healthy or infected fish.

F. ABLATION STUDIES

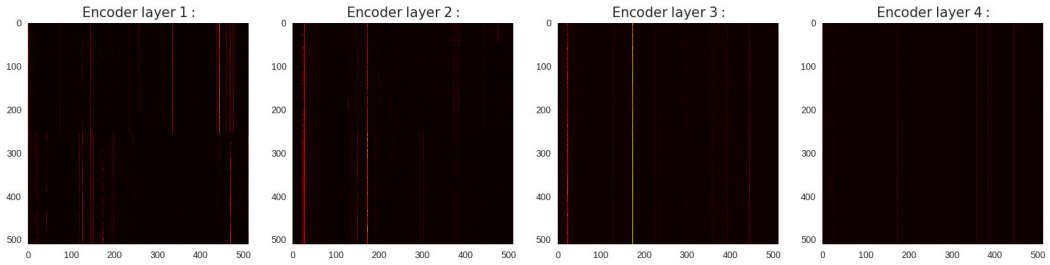
We have shown that EchoBERT is able to accurately detect behavior indicating PD infection in echograms from the real world. However, we have not shown the effects of the different aspect of the models presented. We therefore present a number of ablation experiments to evaluate some of these aspects in our models.

1) MODEL DEPTH

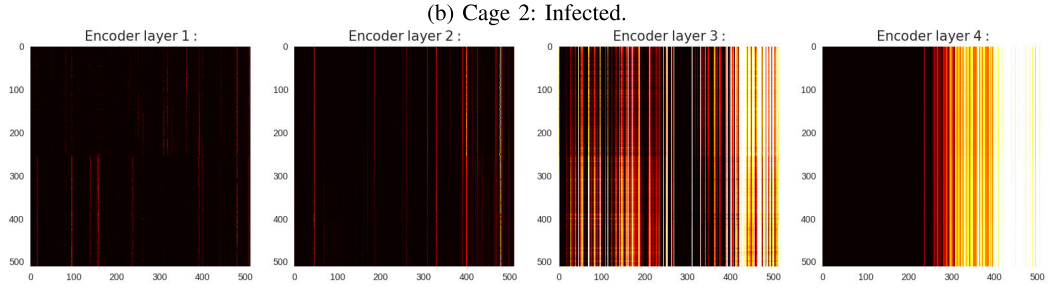
In section IV-C we claim that the best performing model is a transformer-based architecture consisting of $N = 4$ encoder modules. This was compared to an LSTM with the same number of parameters and the same number of layers. To evaluate this claim we trained four EchoBERT models, each with a differing number of layers, using the same hyperparameters and training procedure as described earlier. The results are shown in Table 4 and Fig. 18. We again report the mean MCC score along with the standard deviation over 5 models per cage, starting from the same pre-trained model. The table shows that $N = 4$ indeed seems to be the best performing number of encoder modules for EchoBERT. This is perhaps surprising given the trend of deeper models achieving better results in the deep learning literature [9]. For example,



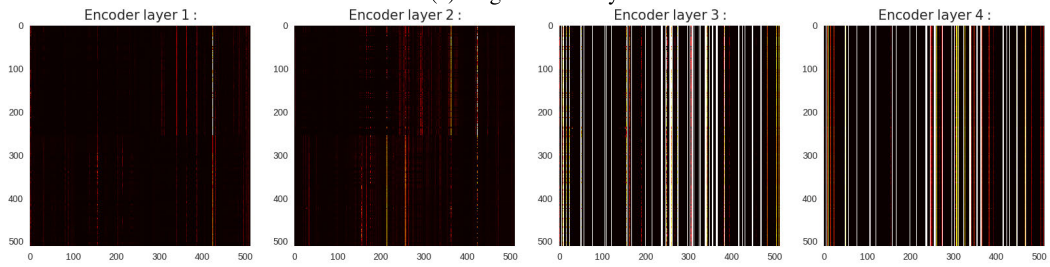
(a) Cage 2: Healthy.



(b) Cage 2: Infected.



(c) Cage 6: Healthy.



(d) Cage 6: Infected.

FIGURE 17. Attention visualization for each of the 4 encoder layers of two EchoBERT models being tested on the 2nd and 6th cage during both healthy and infected time slices. The attention is visualized in a correlation matrix where we plot the input time slice against itself on both the x-axis and the y-axis. Thus when bright pixel values are visible in the matrix, it means that one time step pays particular attention to another time step. For instance in (a) Encoder Layer 2, we see that the time steps from y = 0 to 225, pays particular attention to the x = 310 time step, showing up as a bright vertical line.

TABLE 4. A Comparison of the Different Sizes of EchoBERTs. The Models Consist of N self-Attention Layers, Where Each Row is a Different Size. The Results are Reported Using the Mean and Standard Deviation of the MCC Scores for 5 Separately Trained Models Per Cage.

Model depth	Average MCC score
N = 4	$\mathbf{0.657} \pm 0.168$
N = 6	0.647 ± 0.082
N = 8	0.582 ± 0.109
N = 10	0.559 ± 0.069

the largest EchoBERT is significantly worse than the smallest one. This could be due to the fact that deeper models have more parameters and therefore are more capable of overfitting to the data.

2) EFFECTS OF PRE-TRAINING

We claim that our pre-training task is important to give the model a deep understanding of the dynamics of the echo data. To quantify this claim, we train another set of five EchoBERTs per fold, using the same hyperparameters, but without the pre-training task, thus training them directly on the disease detection task. The results are shown in Table 5 and in Fig. 19. From the results it is clear that the pre-training task is crucial for the performance of the model as the

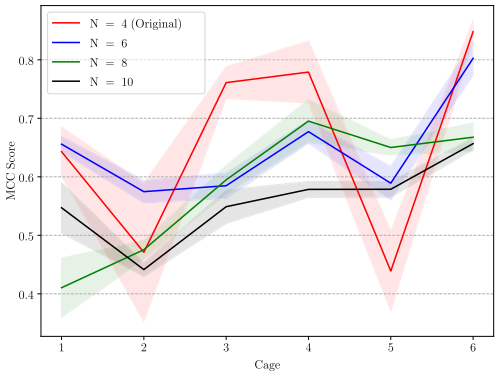


FIGURE 18. A comparison of increasingly deeper EchoBERTs shown using the mean and standard deviation of the MCC score per cage.

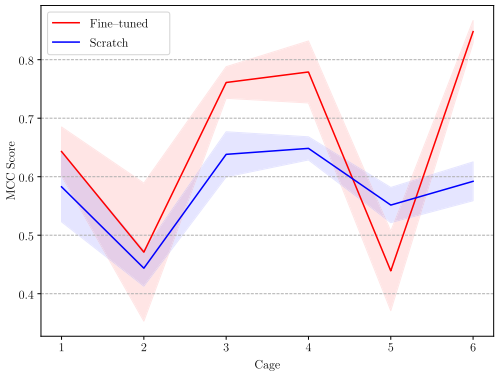


FIGURE 19. Comparing EchoBERTs trained from scratch on the disease data and EchoBERTs that have been pre-trained before they were fine-tuned on the disease data using the mean and standard deviation of the MCC score per cage.

TABLE 5. The Table Shows the Effect of the Pre-Training Task by Comparing Models That Has Been Pre-Trained to Models That Were Trained From Scratch. The Results are Reported Using the Mean and Standard Deviation of the MCC Scores for 5 Separately-Trained Identical Models Per Cage.

Model	Average MCC score
pre-trained	$\mathbf{0.657} \pm 0.168$
scratch	0.576 ± 0.077

non-pre-trained model is far worse across all cages (except cage 5). They are even worse than the LSTM architectures, indicating the importance of a successful pre-training.

3) DIFFERENT INFECTION DATES

As we mentioned in section IV-A, actual infection date is not clear. We therefore examine the effects of changing the onset date by training two sets of 5 models per fold using different infection dates. If the actual infection date is earlier than the date we originally used, we should expect an increase in MCC performance. Since the original model is trained on later infection dates, it incorrectly learns that early disease behavior is healthy behavior. This could confuse the model and make it overly reluctant when presented with infected behavior. However, if the actual infection date is later than the one used in the original model, the opposite may occur. This would cause the original model to be overly eager to classify healthy behavior as infected behavior. To examine such effects, each set of models is trained using a different infection date than the original model. The dates were set to 15.02.19 and 25.04.19, respectively. The results are presented in Table 6 and visualized in Fig. 20. The results show that the originally decided infection date seems to result in the best model performance. Both of the other dates result in comparable performance reductions. This indicates that the originally used infection date is close to the optimal date for detecting disease using behavior patterns from echograms. This could be because this date captures both early infected behavior as well as late infected behavior in a balanced manner. If this is the case, it also indicates that behavior changes as the disease progresses.

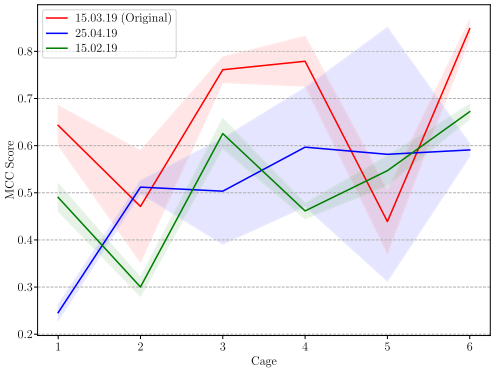


FIGURE 20. Comparing EchoBERTs trained on different target infection dates showing the mean and standard deviation of the MCC score per cage.

TABLE 6. EchoBERTs Trained Using Different Disease Onset Dates. The Results are Reported Using the Mean and Standard Deviation of the MCC Scores for 5 Separately Trained Identical Models Per Cage.

Infection date	Average MCC Score
15.02.19	0.516 ± 0.123
15.03.19 (original)	$\mathbf{0.657} \pm 0.168$
25.04.19	0.505 ± 0.123

V. DISCUSSION

Section III-B mentions that we split our test data on cages, meaning that we end up with a total of six cross-validation folds. Each fold has its own training dataset, where one cage is excluded and used as that fold's test dataset. While this approach gives a better indication of EchoBERT's generalization capabilities, it also means that a large piece of data is unavailable for the model during training. Since salmon behavior is not necessarily uniform across all cages, this could lead to the model being confused. For example, one of the cages might be located in a spot where the underwater current is much stronger than in the other cages. This would change how the fish orient themselves within that cage, causing the echogram to look different than those of the other cages. This could be what is happening with cages

2 and 4. However information about the current conditions is not available for the site, leaving us only to speculate. In spite of this problem, we still achieve strong results, even for these cages. This indicates the robustness of EchoBERT to changing conditions. Our dataset is also relatively small and only contains samples from a single site. Despite the fact that the data consists of several cages, we believe that data from different sites could look very different and therefore cause the model to be fooled. To account for this we would like future work to also include data from other sites.

Another important fact of the echo data is that biomass increases in the cage as the salmon grow over the production period. Since the biomass is at its highest right before the outbreak of PD, one could argue that the model is only considering biomass and decides that all activity happening after the peak is reached is considered infected behavior. However, as we stated in section III-A, the echo sounder used is not able to distinguish individual fish and can therefore only give a very crude estimate of the cage biomass. Furthermore, since the model processes time slices independently, it can only use the temporal information available within a time slice. It therefore does not have access to the historical changes in the biomass that is not present in the current time slice being processed. It is therefore also blocked from leveraging information about biomass in previous time slices when it performs its prediction. This fact substantiates our claim that the model actually understands and uses the behavior of the salmon in its predictions.

When we plot the output of all EchoBERT models a pattern emerges. Throughout most of the 30 trained models a trend is that they start predicting infected behavior slightly earlier than the target time. Rather than being a mistake on the part of the models, this could indicate that the infection date is actually slightly earlier than what we set as our target. Since the models seem to be quite capable at correctly detecting infected behavior it would not be unreasonable to adjust the infection date based on this observation. Consequently the reported scores of our models could actually be even higher, were we to adjust the infection date slightly. The plots showing the model outputs and the targets are shown in the Appendix in Fig. 21.

In our comparison with LSTM networks, we only compare EchoBERT to left-to-right LSTMs. This means that the EchoBERT is able to see time steps both before and after the current position when making its prediction, while the LSTM is only able to see the preceding time steps. This could result in an unfair comparison, especially given that bidirectionality seems important in the pre-training task. This is further substantiated by the pre-training task results. It would therefore be interesting to compare EchoBERT to a bidirectional LSTM to make a better comparison of the architectures. We therefore leave this for future work.

Since EchoBERT is based on BERT [3], it could be interesting to directly compare the two. There are even pre-trained weights for BERT available which could have been used to bootstrap EchoBERT and avoid the need for our pre-training step. However, since BERT is trained on NLP tasks it is poorly suited for the echo domain. This is due to the fixed corpus available in NLP. In the echo domain we do not have a fixed set of possible time step vectors as they are generated from a dynamical system containing nearly unlimited distinct states. This results in a continuous set of possible time step vectors rather than a set of categories, making BERT poorly suited to handle them. It is however possible to drastically downscale the vertical resolution of the echogram to a vertical dimension of n and then bin values to a set of fixed size k. This would effectively create a echogram vocabulary of size k^n, enabling the use of BERT. However, since the vocabulary of BERT is only around 30000 tokens, this would create a vocabulary exceeding BERT's even at very small values of n and k. There are, however, other ways of compressing the information contained within time-step-vectors and this could be an interesting future approach since it would also create a common vocabulary across different locations, potentially enabling a model trained on data from one location to perform quite well in other locations. However, BERT is still trained on data that have vastly different word-embeddings in sequence whereas the echo data would still contain very similar time step vectors when they are temporally close. For this reason it is still reasonable to assume that the pre-training tasks used in EchoBERT are necessary to achieve a sufficient understanding of the underlying dynamics producing the data, thus resulting in a good model even if a common echo vocabulary is used.

Since EchoBERT is not the first example of machine learning being applied to the domain of echo data, it could be interesting to compare it to the other approaches introduced in [2], [4], [6], [19]. However, none of those approaches are concerned with behavior detection, and none include the temporal aspects of echo data making them less likely to be easily adopted. However, in [12] CNNs are combined with an LSTM, making the model able to utilize the temporal aspects of the data. This model could be feasibly adapted to work with echo data, but the fact that it uses multidimensional CNNs makes it better suited for video. The echo data only consists of a sequence 1-dimensional vectors. The use of 2D-convolutional layers would therefore treat the echograms as images, convolving over multiple time steps at once. This makes the model unable to utilize the pre-training tasks used in EchoBERT, making it less likely to accurately represent the data, which would degrade performance. However, since the vertical dimension of the echograms indicates fish position in the water column it is still natural to expect 1-D convolutional layers to be present in EchoBERT. Such layers can capture proximity information and therefore create a better representation for the self-attention layers to process. This is an interesting addition to EchoBERT which we would like to investigate in future work.

In section IV-D we show that an ensemble of EchoBERT models perform even better than individual EchoBERT models. However, it would also be interesting to compare a pure EchoBERT ensemble to an ensemble also containing

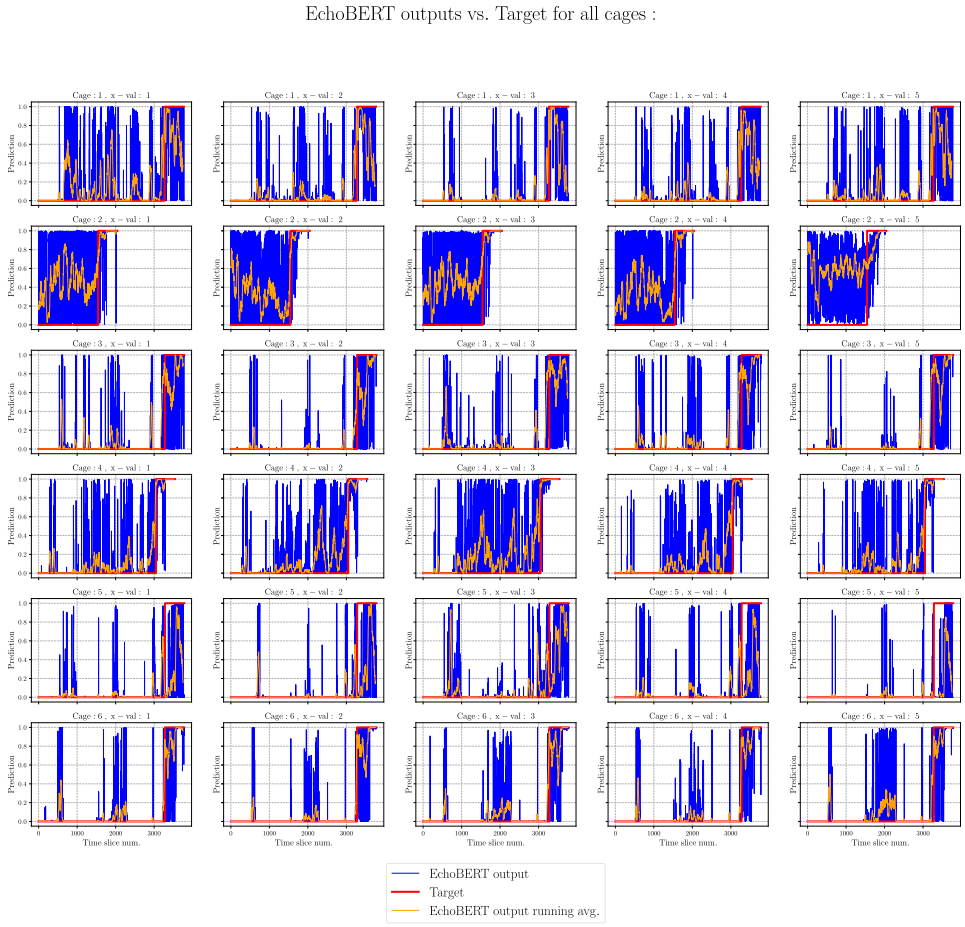


FIGURE 21. EchoBERT outputs, targets and the EchoBERT moving average output for all cages and all cross-validation folds.

LSTMs as the LSTM sometimes outperforms EchoBERT. However, as this work mainly focuses on the introduction of EchoBERT, it is out of the scope of this report. We therefore leave it to future work.

Our results are presented only on the task of disease detection, but there are several other behaviors that are interesting to monitor in salmon, including feeding, fatigue and stress, as well as changes during growth. Since our dataset did actually contain a disease outbreak, this was the most natural task to explore, but future work should include other behaviors, since they are important to salmon farmers and may give better insight into salmon welfare.

In section III-A we mention that the characteristic PD-caused drop in appetite was measured on the 25th of April. However, in our experiments we set our infection date to the 15th of March due to some individuals showing signs of PD already in February through clinical inspection. Our results show that EchoBERT is able to detect infected behavior using this date, thus EchoBERT can detect signs of PD infection from echograms over a month prior to when they show up in traditional PD detection protocols using the measured reduction in appetite on the 25th of April. Since PD is a disease that has a high mortality rate in salmon, these results show that EchoBERT can help reduce fish mortality through early detection that results in earlier treatment.

VI. CONCLUSION

In this work we present EchoBERT, a transformer-based approach for understanding the spatiotemporal dynamics of echograms generated at salmon farming facilities. We show that EchoBERT is able to detect the onset of Pancreas Disease (PD) in salmon, purely from the behavior patterns visible in echogram data. To the best of our knowledge, our approach is the first to treat echograms as sequential data and to use sequence modeling to interpret it. We also introduce a novel vector-substitution pre-training task to further improve the model's understanding of echo data dynamics. We compare EchoBERT to a traditional LSTM model and show that it outperforms the LSTM by a significant margin using Mathew's Correlation Coefficient as a measure. Using an ensemble approach, EchoBERT achieves an MCC score of 0.694 ± 0.178. Furthermore, we show that EchoBERT was able to detect signs of PD over a month prior to the detection using appetite reduction as an indicator. Finally, we perform several ablation experiments to show the effects

of changing various aspects of both the EchoBERT model and the dataset. We find strong indications that EchoBERT is robust to variations in the data and overfitting. However, future work involves verifying this on new data from different cages and farming locations.

APPENDIX A
EchoBERT OUTPUTS

See Fig.21.

REFERENCES

[1] S. Boughorbel, F. Jarray, and M. El-Anbari, "Optimal classifier for imbalanced data using matthews correlation coefficient metric," *PLoS ONE*, vol. 12, no. 6, Jun. 2017, Art. no. e0177678.

[2] A. Charef, S. Ohshimo, I. Aoki, and N. Al Absi, "Classification of fish schools based on evaluation of acoustic descriptor characteristics," *Fisheries Sci.*, vol. 76, no. 1, pp. 1–11, Dec. 2009.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1. Minneapolis, MN, USA: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[4] N. G. Fallon, S. Fielding, and P. G. Fernandes, "Classification of southern ocean krill and icefish echoes using random forests," *ICES J. Mar. Sci.*, vol. 73, no. 8, pp. 1998–2008, Apr. 2016.

[5] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," 2020, *arXiv:1912.02757*. [Online]. Available: https://arxiv.org/abs/1912.02757

[6] Y. Hirama, S. Yokoyama, T. Yamashita, H. Kawamura, K. Suzuki, and M. Wada, "Discriminating fish species by an Echo sounder in a set-net using a CNN," in *Proc. 21st Asia Pacific Symp. Intell. Evol. Syst. (IES)*, Nov. 2017, pp. 112–115.

[7] S. Hochreiter and J. J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 80–1735, Dec. 1997.

[8] J. Jurvelius, F. R. Knudsen, H. Balk, T. J. Marjomäki, H. Peltonen, J. Taskinen, A. Tuomaala, and M. Viljanen, "Echo-sounding can discriminate between fish and macroinvertebrates in fresh water," *Freshwater Biol.*, vol. 53, no. 5, pp. 912–923, May 2008.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 44–436, May 2015.

[10] N. J. MacLachlan and E. J. Dubovi, Eds., "Togaviridae," in *Fenner's Veterinary Virology*, 5th ed. Boston, MA, USA: Academic, 2017, ch. 28, pp. 511–524, doi: 10.1016/B978-0-12-800946-8.00028-3.

[11] H. Måløy, "EchoBERT echo dataset," IEEE Dataport, 2020, doi: 10.21227/76ma-tw16.

[12] H. Måløy, A. Aamodt, and E. Misimi, "A spatio-temporal recurrent network for salmon feeding action recognition from underwater videos in aquaculture," *Comput. Electron. Agricult.*, vol. 167, Dec. 2019, Art. no. 105087.

[13] B. M. Mathisen, K. Bach, E. Meidell, H. Måløy, and E. S. Sjøblom, "FishNet: A unified embedding for Salmon recognition," in *Proc. 24th Eur. Conf. Artif. Intell. (ECAI), 10th Conf. Prestigious Appl. Artif. Intell. (PAIS*, in Frontiers in Artificial Intelligence and Applications, vol. 325, G. De Giacomo, A. Catalá, B. Dilkina, M. Milano, S. Barro, A. Bugarín, and J. Lang, Eds. Santiago de Compostela, Spain: IOS Press, Aug./Sep. 2020, pp. 3001–3008, doi: 10.3233/FAIA200475.

[14] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Struct.*, vol. 405, no. 2, pp. 442–451, Oct. 1975.

[15] A. H. McVicar, "Pancreas disease of farmed atlantic salmon, salmo salar, in scotland: Epidemiology and early pathology," *Aquaculture*, vol. 67, nos. 1–2, pp. 71–78, Dec. 1987.

[16] E. Misimi, E. R. Øye, Ø. Sture, and J. R. Mathiassen, "Robust classification approach for segmentation of blood defects in cod fillets based on deep convolutional neural networks and support vector machines and calculation of gripper vectors for robotic processing," *Comput. Electron. Agricult.*, vol. 139, pp. 138–152, Jun. 2017.

[17] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. (2018). *Improving Language Understanding By Generative Pre-Training*. [Online]. Available: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

[18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[19] A. Rezvanifar, T. P. Marques, M. Cote, A. B. Albu, A. Slonimer, T. Tolhurst, K. Ersahin, T. Mudge, and S. Gauthier, "A deep learning-based framework for the detection of schools of herring in echograms," *CoRR*, vol. abs/1910.08215, 2019. [Online]. Available: http://arxiv.org/abs/1910.08215

[20] M. Själander, M. Jahre, G. Tufte, and N. Reissmann, "EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure," 2019, *arXiv:1912.05848*. [Online]. Available: https://arxiv.org/abs/1912.05848

[21] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," *Proc. SPIE*, vol. 11006, pp. 369–386, May 2019.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[23] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 4, pp. 687–719, Jun. 2009.

[24] T. Taksdal, A. B. Olsen, I. Bjerkås, M. J. Hjortaas, B. H. Dannevig, D. A. Graham, and M. F. McLoughlin, "Pancreas disease in farmed Atlantic salmon, Salmo Salar L., and rainbow trout, oncorhynchus mykiss (Walbaum), in Norway," *J. Fish Diseases*, vol. 30, no. 9, pp. 545–558, Sep. 2007.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, U. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2017, pp. 5998–6008.

[26] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 5753–5763.

HÅKON MÅLØY received the B.Sc. degree in informatics and the M.Sc. degree in computer science from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2014 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Department of Computer Science. His research interests include data-driven methods applied to sequential data and multimedia.