Estudio de Data Augmentation con Modelos Generativos en Audios de Ballenas

Martín Bianchi
Departamento de Ingeniería
Universidad de San Andrés
Victoria, Argentina
martinbianchi@udesa.edu.ar

Federico Gutman
Departamento de Ingeniería
Universidad de San Andrés
Victoria, Argentina
fgutman@udesa.edu.ar

***Resumen*—Se aborda la detección automática de llamados de ballenas en entornos ruidosos utilizando el dataset del Marinexplore and Cornell University Whale Detection Challenge, caracterizado por un fuerte desbalance de clases. Para mitigar este problema, se entrenan modelos generativos como el autoencoder variacional con regularización (Beta-VAE), el autoencoder adversarial (AAE) y la red generativa adversarial (GAN), que sintetizan representaciones espectro-temporales de up-calls. Estas muestras se incorporan al entrenamiento de clasificadores bajo distintos esquemas: sin datos sintéticos, con ponderación de clases y con datos generados. Aunque los generadores producen ejemplos plausibles que se solapan con los datos reales en el espacio latente, no se observan mejoras significativas en las métricas de clasificación. Esto sugiere que, al entrenarse sobre el mismo conjunto, las muestras sintéticas no aportan información nueva ni enriquecen el espacio de decisión. Los resultados destacan la relevancia del preprocesamiento y del clasificador elegido, y marcan los límites del data augmentation generativo en este contexto.**

***Index Terms*—Data augmentation, modelos generativos, VAE, AAE, GAN, mel-espectrogramas, clasificación binaria, espectrogramas.**

I. Introducción

En este trabajo se abordó un problema de clasificación binaria altamente desbalanceado: identificar la presencia de llamados de ballena (*up-calls*) en grabaciones de audio marcadas como positivas (con llamado) o negativas (solo ruido). Estas señales, grabadas por hidrófonos en ambientes marinos, se transforman en mel-espectrogramas, una representación en frecuencia que facilita el análisis estructural de los sonidos y mejora el desempeño de los modelos de aprendizaje automático.

El objetivo es entrenar un modelo que, dada una nueva muestra, prediga si contiene un up-call o no. Una dificultad central es que las muestras positivas representan una fracción muy pequeña del total, lo que suele sesgar a los clasificadores hacia la clase mayoritaria y dificulta la detección de los llamados reales.

Muchos enfoques previos se basan en técnicas de preprocesamiento avanzadas y arquitecturas robustas para mitigar este problema. Nosotros exploramos una estrategia complementaria: utilizar modelos generativos para sintetizar nuevos ejemplos de la clase minoritaria y aumentar así su presencia en el conjunto de entrenamiento. La hipótesis es que, si estas muestras son lo suficientemente representativas y diversas, podrían ayudar al clasificador a generalizar mejor, reduciendo el overfitting y facilitando la separación entre clases.

Para ello, se emplearon modelos como el autoencoder variacional (VAE), el autoencoder adversarial (AAE) y la red generativa adversarial (GAN) para generar espectrogramas sintéticos de up-calls. Estas muestras fueron utilizadas junto con las reales para entrenar clasificadores supervisados como perceptrón multicapa (MLP) con y sin convoluciones, comparando su rendimiento bajo distintos escenarios de entrenamiento. En particular, se evalúa si el agregado de datos sintéticos mejora las métricas de clasificación respecto de entrenamientos basados únicamente en datos reales o técnicas de ponderación por clase.

Además, se exploró el uso de arquitecturas convolucionales para explotar la estructura local de los espectrogramas, contrastando su desempeño con modelos más simples como redes densamente conectadas sin uso de convoluciones. El objetivo final es evaluar si una estrategia basada en generación de datos mediante modelos generativos puede mejorar efectivamente el aprendizaje en contextos de desbalance extremo.

II. Conjunto de datos y características

Este trabajo se basó en el dataset de la competencia *The Marinexplore and Cornell University Whale Detection Challenge*, disponible en Kaggle [4]. El conjunto incluye 30.000 audios etiquetados para entrenamiento y 50.000 archivos de test sin etiquetas públicas. Cada archivo contiene un clip de audio registrado por sensores submarinos.

El objetivo es detectar llamadas de ballena (up-calls), clasificando cada audio como positivo (1) o negativo (0). Uno de los mayores desafíos es el fuerte desbalance de clases: el 77 % de las muestras corresponde a ruido, mientras que solo el 23 % restante corresponde a audios de ballenas.

Desde el punto de vista auditivo, los ejemplos positivos presentan un sonido característico: un tono agudo que

asciende suavemente en frecuencia, correspondiente al *up-call* típico de las ballenas. Este patrón es fácil de reconocer al escucharlo, ya que contrasta con los audios negativos, que en general contienen solo ruido ambiental o estático, sin ninguna estructura tonal clara. Esta diferencia entre las clases también se refleja en los espectrogramas, donde los up-calls aparecen como trazos inclinados con mayor energía en bandas específicas, mientras que el ruido se muestra como una textura más dispersa y sin forma definida.

Para contar con un clasificador de referencia lo más sólido posible, se intentó reutilizar el modelo de uno de los participantes más destacados del leaderboard oficial [3]. Este modelo, originalmente desarrollado en **Python 2** y con varias dependencias rotas o desactualizadas, requería una adaptación completa para poder ejecutarse en nuestro entorno actual. Tras varios ajustes, se logró ponerlo en funcionamiento y utilizarlo como un **discriminador externo fuerte**, como herramienta auxiliar en la evaluación de muestras generadas por modelos como el GAN o el AAE.

Cada audio fue transformado a su representación en frecuencia usando *espectrogramas de mel* o *mel-espectrogramas*, que reflejan mejor la percepción humana del sonido. Fue detectado que los up-calls se concentran entre 100 y 500 Hz, por lo que los espectrogramas fueron truncados a 600 Hz, descartando información irrelevante.

También fue aplicada normalización global (z-score). Otras técnicas fueron probadas (como min-max), pero reducían demasiado la variabilidad del espectrograma. Esta decisión fue clave para mejorar la estabilidad de modelos generativos.

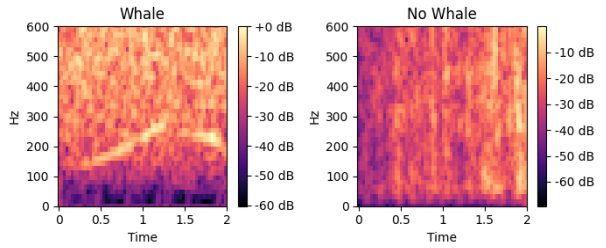

Figura 1: Promedios de mel-espectrogramas: clase 1 (ballena), clase 0 (ruido).

En la Fig. 1 se observan diferencias claras entre clases: los ejemplos positivos presentan un aumento localizado de energía, mientras que los negativos exhiben ruido más difuso.

También se extrajo un conjunto de características acústicas agregadas con el objetivo de evaluar si las clases 1 (ballena) y 0 (ruido) resultaban lo suficientemente diferenciables como para entrenar modelos directamente sobre estas features. Sin embargo, no fue posible observar diferencias significativas entre clases, por lo que no se continuo con este enfoque y se opto por el uso de mel-espectrogramas como representación principal.

Los espectrogramas fueron usados directamente como entrada a nuestros modelos. Además, aplicamos *t-Distributed Stochastic Neighbor Embedding (t-SNE)* y *Análisis de Componentes Principales (PCA)* para visualizaciones y análisis de distribución, aunque no se incluyeron en el entrenamiento.

III. Metodología

El enfoque utilizado combina representaciones espectro-temporales con modelos generativos para sintetizar ejemplos de ballenas. A continuación se describen los métodos utilizados.

III-A. Mel-espectrogramas

Para representar cada señal de audio se usaron mel-espectrogramas $X \in R^{T \times F}$. Esta representación se construye en dos pasos:

1. Se divide la señal $x(t)$ en ventanas solapadas y se aplica una transformada de Fourier corta (STFT) para obtener el espectrograma $S(f,t)$.
2. Luego se aplica un banco de filtros mel $M(f)$ y se computa la energía en cada banda, como se muestra en la Ecuación (1)

$$X(t,m) = \sum_f |S(f,t)|^2 \cdot M_m(f) \tag{1}$$

$M_m(f)$ representa el filtro mel m aplicado sobre la frecuencia f, y $X(t,m)$ es la energía en el tiempo t para la banda mel m.

Cada fila del mel-espectrograma representa una ventana temporal, y cada columna una banda de frecuencia perceptual. La escala mel concentra resolución en frecuencias bajas (donde suelen encontrarse los up-calls) y reduce el detalle en frecuencias altas, donde hay más ruido. Esta representación compacta permite capturar de forma eficiente la estructura espectro-temporal de los sonidos de ballenas.

III-B. Beta-VAE

El autoencoder variacional (VAE) busca modelar una distribución latente $q(z|x)$ a partir de la cual pueda reconstruirse la entrada x. La función objetivo del VAE se muestra en la Ecuación (2), donde el primer término representa la reconstrucción y el segundo la regularización KL.

$$\mathcal{L} = E_{q(z|x)}[\log p(x|z)] - D_{\mathrm{KL}}(q(z|x)\|p(z)) \tag{2}$$

El Beta-VAE introduce un parámetro β, como se observa en la Ecuación (3), que permite ajustar la fuerza de regularización.

$$\mathcal{L}_\beta = E_{q(z|x)}[\log p(x|z)] - \beta \cdot D_{\mathrm{KL}}(q(z|x)\|p(z)) \tag{3}$$

Con $\beta > 1$, se obtiene una estructura latente más disentangled, que favorece la diversidad y separación de factores de variación en las muestras generadas.

III-C. Adversarial Autoencoder (AAE)

El AAE reemplaza el término KL por un entrenamiento adversarial. Utiliza un discriminador $D(z)$ que fuerza al codificador $q(z|x)$ a producir un espacio latente que se asemeje a una distribución prior $p(z)$. La arquitectura combina:

- Un autoencoder que minimiza el error de reconstrucción.
- Un discriminador que distingue entre $z \sim q(z|x)$ y $z \sim p(z)$.

Este enfoque permite regularizar el espacio latente sin requerir una forma analítica para el KL.

III-D. Spectrogram GAN

La red generativa adversarial (GAN) consiste en un generador $G(z)$ que transforma ruido latente en muestras sintéticas, y un discriminador $D(x)$ que intenta distinguirlas de las reales. El objetivo de entrenamiento se define en la Ecuación (4).

$$\min_G \max_D \; E_{x\sim p_{\text{data}}}[\log D(x)] + E_{z\sim p(z)}[\log(1 - D(G(z)))] \tag{4}$$

En este trabajo se usó una variante conocida como **Spectrogram GAN** [2], que adapta la arquitectura de las GAN convolucionales para trabajar con representaciones espectrales de audio, como los mel-espectrogramas, donde tanto el generador como el discriminador están compuestos por capas convolucionales. Este enfoque busca generar espectrogramas realistas desde ruido latente, que luego pueden ser reconstruidos como señales de audio.

III-E. Reducción de dimensionalidad: PCA y t-SNE

El Análisis de Componentes Principales (PCA) proyecta los datos sobre una base ortogonal que maximiza la varianza, permitiendo una representación lineal reducida. El método t-SNE (t-Distributed Stochastic Neighbor Embedding) es no lineal, y busca preservar la estructura local en proyecciones a 2D o 3D. Ambos métodos fueron empleados para visualizar la distribución de las muestras generadas y reales.

III-F. Modelos de clasificación

Regresión lineal. Busca una frontera de separación lineal entre clases. Se entrenó sobre entradas vectorizadas.

Random Forest. Ensamble de árboles de decisión entrenados sobre subconjuntos aleatorios de datos y características. Mejora robustez y reduce sobreajuste mediante votación.

Gradient Boosting. Construye árboles secuencialmente, corrigiendo errores residuales mediante gradientes. Captura relaciones complejas y suele lograr alta precisión.

Perceptrón multicapa (MLP). Red neuronal densa con capas ocultas. Modela relaciones no lineales entre las entradas y la salida binaria.

III-G. Métricas de evaluación

Para evaluar la calidad de las muestras generadas, fueron entrenados clasificadores (lineales, MLP y CNN) sobre distintas combinaciones de datos reales y sintéticos. Las métricas utilizadas fueron:

- **Accuracy**: proporción de predicciones correctas sobre el total.
- **F1-score**: media armónica entre precisión y recall, útil ante clases desbalanceadas.
- **AUC-ROC**: área bajo la curva ROC, que mide la capacidad del modelo para separar las clases en distintos umbrales.
- **Matriz de confusión**: matriz que resume el desempeño del modelo mostrando los verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, facilitando el análisis detallado de errores.

Estas métricas se computaron sobre un conjunto de validación, comparando el rendimiento al entrenar con datos reales, sintéticos, o una mezcla de ambos.

IV. Experimentos, resultados y discusión

Comenzamos el proyecto probando clasificadores simples como **MLP**, **Random Forest** y **XGBoost** para establecer un *baseline*. No obstante, no se encontraron grandes diferencias entre ellos. Posteriormente, se integraron convoluciones al MLP, logrando una mejora en performance considerable respecto a los demás modelos, además de ofrecer una velocidad de entrenamiento mayor. Es por esto que se eligió este modelo como baseline para evaluar el rendimiento al entrenar con datos reales y sintéticos.

Con esta base, se avanzó con el eje central del trabajo: utilizar **modelos generativos** para sintetizar ejemplos de ballenas y atacar el desbalance de clases. Se entrenaron un **VAE**, **AAE** y **GAN**, comenzando con versiones simples sin convoluciones. Los resultados iniciales no fueron buenos: los generadores producían muestras casi idénticas, con un *up-call* promedio y sin variabilidad.

Para mejorar, se incorporaron **capas convolucionales**, lo que elevó un poco la calidad, pero todavía no se conseguian buenos resultados. Al revisar el preprocesamiento, se descubrió que la normalización que estaba siendo utilizada no manejaba bien los *outliers*, comprimiendo excesivamente los espectrogramas. Al aplicar *standard normalization*, los modelos empezaron a aprender estructuras útiles. La **Beta-VAE** fue el modelo más estable y consistente durante el entrenamiento. Generó espectrogramas variados que conservan bien la estructura del *up-call*, capturando patrones comunes de la clase real sin colapsar hacia una única forma. Fue además el que menos ajustes finos necesitó y se comportó de forma robusta a través de diferentes combinaciones de hiperparámetros.

La **AAE**, en cambio, resultó menos estable y más sensible a la configuración. Si bien logró generar muestras con la forma general del *up-call*, estas venían muchas veces

acompañadas de ruido residual o distorsiones. El modelo requería mayor regularización y ajustes para no colapsar o generar espectrogramas poco interpretables.

Por último, la **Spectogram GAN** logró generar las muestras de mayor *fidelidad visual*. Tras varias pruebas, incorporar capas convolucionales fue clave para obtener espectrogramas casi indistinguibles de los reales. A pesar de esto, su entrenamiento fue el más desafiante: sensible a la inicialización, propenso a inestabilidad y requería monitoreo constante para evitar colapsos del generador o el discriminador.

Durante los experimentos con GAN y AAE, se intentó utilizar como discriminador un clasificador externo entrenado por uno de los equipos más exitosos del leaderboard. Sin embargo, su complejidad lo volvía excesivamente estricto, penalizando incluso pequeñas desviaciones y dificultando el aprendizaje del generador. Lejos de mejorar la calidad, esto causaba colapsos o aprendizaje nulo. En cambio, un **discriminador simple**, con pocas capas, ofreció una señal de error más tolerante y efectiva, guiando mejor el entrenamiento. Paradójicamente, el modelo del leaderboard —que se esperaba ventajoso— resultó contraproducente frente a un clasificador liviano.

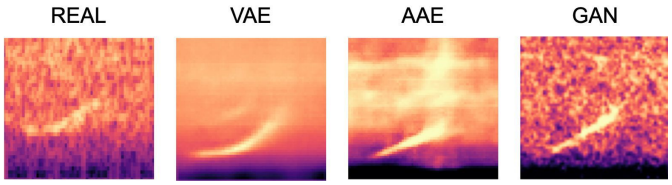

Figura 2: Comparación de una muestra sintética generada por cada modelo (Beta-VAE, AAE y GAN) junto a una muestra real (derecha).

Como se observa en la Fig. 2, las muestras generadas por el Beta-VAE y la AAE presentan una apariencia más suave y continua, mientras que tanto la muestra real como la generada por la GAN tienen un aspecto más cuadriculado o "pixeleado". Esta diferencia se debe a la naturaleza del modelado: tanto el Beta-VAE como la AAE reconstruyen cada pixel como una función continua de los valores vecinos, promoviendo transiciones suaves en la imagen. En cambio, la GAN, al no modelar explícitamente la distribución de los datos sino buscar engañar a un discriminador, tiende a generar detalles más definidos pero también más susceptibles a artefactos visuales, como patrones cuadriculados.

Más allá de la fidelidad visual que se observa en los espectrogramas, también fue realizada una evaluación auditiva, reconstruyendo los audios correspondientes para escucharlos. Al hacerlo, se pudo notar que que los patrones característicos de la clase objetivo —en particular el típico *up-call*, que se percibe como un tono agudo que va subiendo— también están presentes en los audios generados, aunque con diferencias según el modelo.

En particular, el audio generado por el AAE conserva la forma del *up-call*, pero viene acompañado de un ruido de fondo que se mantiene durante toda la señal, lo mismo en el caso del GAN. Este ruido coincide con lo que se ve en sus espectrogramas, donde aparecen zonas con textura o distorsión visual. No obstante, los audios reales también contienen ruido estático y del oceáno, por ende estos fueron resultados esperables. En cambio, los audios generados por el VAE modelos no mostraron ese tipo de ruido audible.

Para analizar si las muestras sintéticas replicaban adecuadamente la distribución de las reales, se aplicó reducción de dimensionalidad con **PCA** y **t-SNE**, cuyos resultados se muestran en la Fig. 3. Para este análisis se usaron las muestras generadas por la **GAN**, dado que fue el método de generación que mejor se asemejaba a los datos reales.

En PCA, las muestras generadas y reales se superponen completamente, compartiendo forma y rango de dispersión, lo que indica que el generador logra replicar la estructura global de la distribución de los datos originales en el espacio lineal.

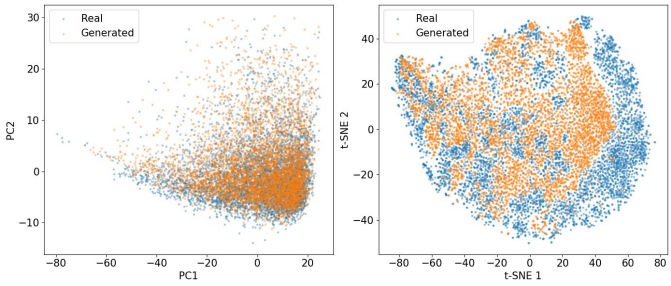

Figura 3: t-SNE y PCA mostrando la superposición de muestras sintéticas y reales en el espacio latente.

En t-SNE, las muestras sintéticas (naranja) y reales (azul) forman juntas una figura bien definida en la que se encuentran completamente mezcladas, con sintéticas embebidas entre las reales y viceversa. Aunque existen zonas donde predomina un color, las muestras coexisten en toda la figura, mostrando que las generadas logran insertarse de manera realista dentro del espacio local de las muestras reales. Esto sugiere que el generador no solo cubre la estructura global de la distribución, sino que también produce ejemplos plausibles que se ubican en posiciones intermedias a nivel informativo entre distintas muestras reales, manteniendo la coherencia de la estructura latente.

Una vez hecho esto, se usaron las muestras sintéticas para balancear el dataset y entrenar nuevamente el **MLP** convolucional en tres escenarios: sin datos sintéticos, con ponderación de clases, y con datos sintéticos agregados. En todos los casos, los rendimientos fueron muy similares, lo que sugiere que las muestras generadas no aportaron mejoras significativas en este contexto.

Luego, dado que los generadores utilizados incluían capas convolucionales y que los datos de entrada eran

espectrogramas (estructuras con fuertes correlaciones espaciales locales), decidimos reemplazar el MLP por una **CNN**. Incorporamos convoluciones tanto en la entrada como en el clasificador final, buscando aprovechar mejor la estructura espacial de los datos. Además, aplicamos una combinación de estrategias para robustecer el entrenamiento: **ponderación de clases**, **data augmentation acústico** (con transformaciones como cambio de velocidad y mezcla con ruido), y **datos sintéticos** generados por los tres modelos (Beta-VAE, AAE y GAN).

La Fig. 4 muestra los resultados obtenidos en validación con esta CNN. Las métricas fueron prácticamente idénticas en todos los casos, alcanzando un **F1-score de 0.975**, lo que refuerza la hipótesis de que las muestras sintéticas no aportan información adicional significativa para el clasificador, al menos en este problema y con este nivel de separabilidad de clases.

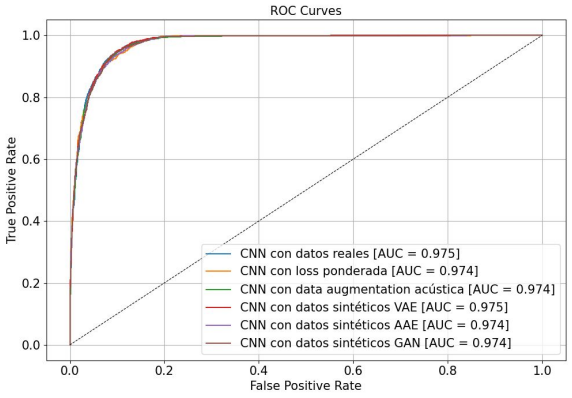

Figura 4: Comparación de precisión, recall y F1-score para la CNN bajo diferentes estrategias: solo datos reales, con ponderación de clases, con data augmentation acústico, y con datos sintéticos generados por los tres modelos.

Si bien el objetivo principal del trabajo no fue diseñar el mejor clasificador posible, sino analizar el valor de los datos generados, cabe destacar que este modelo final alcanza un rendimiento que lo ubicaría dentro del **top 15 del leaderboard oficial** de la competencia, con un **AUC de 0.975**.

Los modelos entrenados presentaron una loss sobre el conjunto de validación de 0.17, una accuracy de 0.92 %, un F1-Score promedio de 0.84. Este fue el caso para todos los modelos menos para la red entrenada con datos reales y la loss ponderada, el cual presentó un rendimiento un poco peor con una loss sobre el conjunto de validación de 0.19.

Buscando forzar al modelo a necesitar las muestras sintéticas, intentamos múltiples estrategias:

- **Reducir drásticamente el tamaño del conjunto de entrenamiento**, utilizando solo un 10 % de los datos originales para entrenar y completando el resto con muestras sintéticas, para evaluar si lograban aportar información faltante.
- **Probar con un clasificador de menor capacidad** (regresión lineal), esperando que al requerir más datos para separar las clases pudiera aprovechar las muestras generadas. Sin embargo, observamos que el problema era *linealmente separable* incluso con pocos datos, aprendiendo de manera perfecta con un subconjunto reducido del dataset.
- **Agregar ruido al conjunto de validación**, buscando que las diferencias en balance y representación se reflejaran de forma más marcada y obligaran al modelo a usar mejor la información adicional.

A pesar de estos intentos, los resultados permanecieron prácticamente inalterados en términos de métricas. Ninguna de estas estrategias permitió a los modelos extraer información significativa de las muestras sintéticas. Esto refuerza la hipótesis de que, dado el nivel de separabilidad de los datos originales y la calidad del preprocesamiento, las muestras generadas no aportaron diversidad relevante al espacio de representación.

V. Conclusión y trabajo futuro

La experiencia confirmó que, en este problema, el **preprocesamiento cuidadoso y la arquitectura del clasificador** son factores decisivos. La CNN superó con claridad al MLP y al resto de los métodos probados, demostrando una mayor capacidad para aprovechar la estructura de los datos en formato de espectrograma.

Como trabajo futuro, sería interesante explorar **transfer learning**, utilizando redes preentrenadas en datos de audio o modelos de espectrogramas para mejorar la representación de las muestras de ballenas. También podría investigarse la aplicación de modelos generativos condicionados para diversificar los ejemplos sintéticos, incrementando así la variabilidad en las muestras generadas.

Referencias

[1] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," 2016. [Online]. Available: https://arxiv.org/pdf/1606.05579

[2] C. Donahue, J. McAuley, and M. Puckette, "Synthesizing audio with generative adversarial networks," 2018. [Online]. Available: https://arxiv.org/abs/1802.04208v3

[3] TarinZ, "Whale detector: Right whale upcall classification," GitHub repository, 2023. [Online]. Available: https://github.com/TarinZ/whale-detector

[4] Kaggle, "Whale Detection Challenge Dataset," 2023. [Online]. Available: https://www.kaggle.com/competitions/whale-detection-challenge/data

[5] C. Chin, "Right Whale Convolutional Neural Network," GitHub repository, 2023. [Online]. Available: https://github.com/cchinchristopherj/Right-Whale-Convolutional-Neural-Network