Unsupervised Clustering for Detecting New Zealand Bat Echolocation Calls
-
Web Application *Pekapeka*

COMPSCI 380

June 2025

Henri Chemla
Email: hche697@aucklanduni.ac.nz
University of Auckland

Supervised by
Dr Jörg Wicker
Department of Computer Science
University of Auckland

Contents

I Acknowledgments 3

II Unsupervised Clustering for Detecting NZ Bat Calls 3

1 Introduction 3

2 Related Work 4

3 Methodology 4

3.1 Data Collection and Preprocessing 4
3.2 Clustering Procedure 5

4 Results 7

5 Future Work and Limitations 11

6 Conclusion 11

III Web Application *Pekapeka* 12

7 Introduction 12

8 Domain and Requirements 12

8.1 Functional Requirements 12
8.2 Non-Functional Requirements 12

9 Users and Actors 13

10 Architecture Overview 13

11 Database Model 14

12 API Surface 15

13 Deployment and Operations 16

14 Future Work 16

15 Conclusion 17

Part I
Acknowledgments

I would like to thank my supervisor **Dr. Jörg Wicker** for providing eco-acoustic bat data this semester, for his guidance, and for enabling the development of *Pekapeka* by provisioning a virtual machine.

Part II
Unsupervised Clustering for Detecting NZ Bat Calls

1 Introduction

New Zealand's native bats are the country's only native land mammals and are thus of high conservation concern. According to the New Zealand's Department of Conservation (DOC) they are currently at high risk of extinction [1]. There are two extant species: the Long-tailed bat (*Chalinolobus tuberculatus*, classified as "Nationally Critical") and the Lesser short-tailed bat (*Mystacina tuberculata*), with statuses ranging from "Vulnerable" to "Recovering". Bats navigate and hunt in darkness using ultrasonic echolocation calls, which are in a range that is generally inaudible to humans [2].

Acoustic ecology (or "eco-acoustics") offers a non-invasive approach to monitor wildlife by deploying recording devices that can continuously capture their surrounding environments [3].

Ultrasonic recorders can log thousands of audio files over days, weeks or months, capturing bat calls amid background noise like wind, rain, insects, etc... However, large volumes of audio in most situations and in ours (approximately 400GB of data) makes manual analysis unfeasible. This motivates the use of machine learning techniques to detect and identify bat calls within large acoustic datasets. Unsupervised learning can be used to discover patterns and group similar acoustic events without human annotation. This is valuable in cases such as ours, where annotated data is not present.

This report presents an unsupervised clustering approach to automatically identifying candidate bat echolocation calls from a large collection of recordings, with the goal of being able to discern whether a bat call is present in an audio or not and ideally, by what species the noise was produced. A qualitative interpretation of the results is finally done as part of the work in Section 4, to map the resulting clusters to known bat calls types and species.

2 Related Work

A 2023 systematic review of machine-learning in eco-acoustics reports that while unsupervised clustering and dimensionality-reduction methods could identify groups of sounds, there is also a general trend towards CNNs and deep-learning as hardware and libraries improve [4, 7].

BatDetect [6] demonstrated that CNNs could outperform commercial detectors on European datasets. Roemer *et al.* combined 1.15 million labelled recordings containing bat calls from four different continents to train a classifier with 1,000 decision trees, obtaining AUC values between 0.89 and 1 on mid-to-high quality calls [5].

While supervised methods require labels and annotations, unsupervised clustering provides an ability to separate different types of elements and features into different clusters. LAMDA clustering for bat "sonotypes" [11] and density-based analysis of coral-reef soundscapes [12] are all examples of how unsupervised grouping can reveal categories of sounds in an environment. Our attempt at carving out bat calls out of audio files using k-means clustering goes along the lines of the research done here.

3 Methodology

3.1 Data Collection and Preprocessing

The dataset consisted of 40GB of approximately 20,000 full-spectrum audio recordings (in .wav format) collected from ultrasonic bat detectors deployed in the Waitakere Ranges, West of Auckland, in New Zealand by the Community Waitakere group [8]. Each recording was typically of short duration (2 to 15 seconds) and captured in the ultrasonic range (up to 312.5kHz).

To isolate potential bat calls from background noise, all files were iterated through, and a slider of 6ms was passed over each file, with a sliding interval of 3ms. In each window's 20kHz to 100kHz range, a simple energy threshold detector looked for short pulses of ultrasound that exceeded by 4 times the standard deviation of the global average mean-square amplitude of all of the aggregated files. The window of 6ms was chosen as it comfortably held the length of a Long-tailed bat buzz call or a Lesser short-tailed bat call. However, it segments the Long-tailed bat search calls, which are on the order of 1ms to 40ms [9].

This processing method produced a collection of approximately 250,000 candidate call clips (6ms each) across all recordings. The short window enables focus on short noises, which is suitable for capturing bat calls while excluding longer noise fluctuations. The chosen 6ms window size comfortably captures the short echolocation calls of the Lesser-short tailed bat (typically 3-4ms) and the short feeding buzzes of the Long-tailed bat (typically 1ms). It also enables the detection of the Long-tailed bat's search phase call (typically 20 to 40ms); although roughly 5 times the length of a 6ms window, these longer calls can be identified via their peak energy portion which is in the 35kHz to 40kHz range, allowing for identification similar to the shorter buzz calls of the same species [9].

Each 6ms waveform snippet was then transformed into a 64-length feature vector using a Short Time Fast Fourier Transform method for feature extraction, with each feature being a mel-band of the sound window. Then, on each window, the average amplitude obtain features representing the frequency content of the clip. For clustering (see Section 4) we also extracted summary features such as the peak frequency (frequency of maximum amplitude in the clip) and the bandwidth of the signal. .

Prior to clustering, we applied Principal Component Analysis (PCA) to reduce the dimensionality of the feature space. The raw spectrogram features (each of the 64 features) were reduced to a smaller number of components. PCA helps to reduce the dimensionality of the data, which is beneficial given the large number of clips and the presence of multiple noise dimensions.

3.2 Clustering Procedure

A k-means algorithm was used on this dataset. More specifically, the `scikit-learn` MiniBatch implementation of the k-Means variant was chosen. It can efficiently handle large datasets by using small random batches to update cluster centroids, speeding up clustering on approximately 250,000 data points with only a minor loss in accuracy compared to standard k-means. The algorithm aims to partition the feature vectors into k clusters such that each clip belongs to the nearest cluster centroid in PCA feature space.

The key hyperparameter in k-means is the number of clusters k, which we needed to determine. To select an appropriate k, we ran diagnostics by iterating through values of k and evaluating an "elbow/silhouette" procedure. Total inertia (within cluster sum of squared distances) and the average silhouette value were plotted for different values of k. The inertia generally decreases as k increases (clusters get smaller and tighter), while the silhouette score measures how well-separated the clusters are.

After fixing $k = 9$, and operating PCA reduction on the feature set, we trained the MiniBatch k-Means model using a batch size of 4096 and running for sufficient iterations to ensure convergence of centroid positions. The result was 9 cluster centroids in the feature space, with each of the approximately 250,000 clips assigned a cluster label 0 through 8. We then computed summary statistics for each cluster (such as the number of clips, the average peak frequency, average bandwidth, and average signal amplitude per cluster to help interpret what each cluster represented.

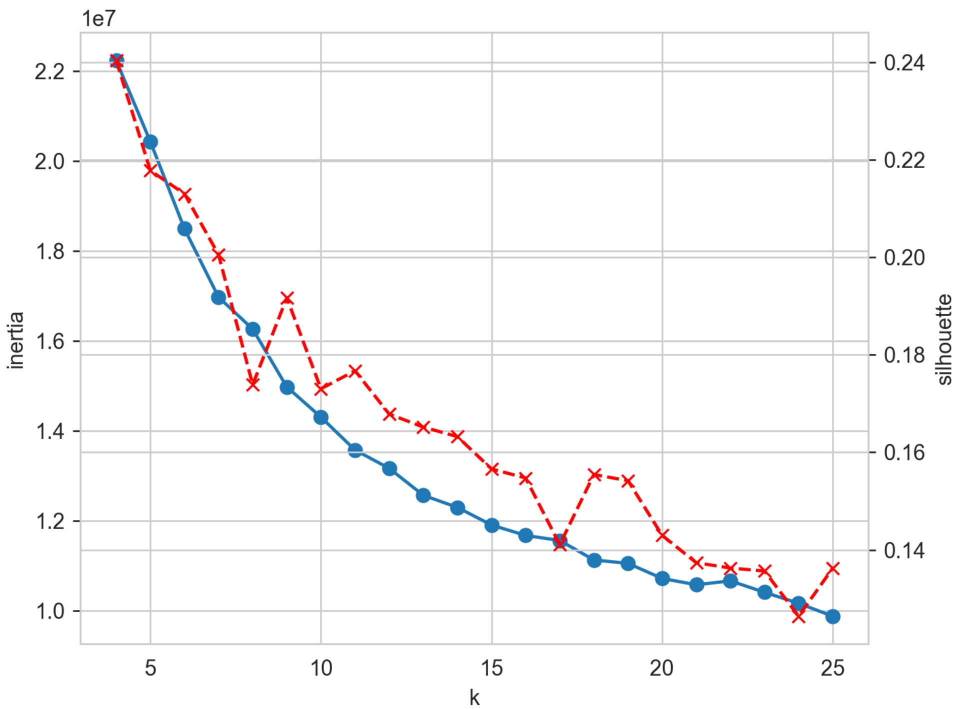

Figure 1: Elbow-Silhouette analysis for choosing the number of clusters k. The plot shows inertia (sum of squared errors, blue curve, left axis) and mean silhouette score (red curve, right axis) relative to number of k chosen. There is an elbow in the inertia curve around $k = 8$ and a peak in silhouette around $k = 9$ suggesting that approximately 8 or 9 clusters could be an appropriate choice. Ultimately it was chosen that 9 clusters would be suitable.

4 Results

After clustering, we obtained 9 distinct clusters of acoustic events. To visualize the clustering outcome, t-distributed Stochastic Neighbor Embedding (t-SNE) was used to project the high-dimensional clip features into two dimensions for plotting.

t-Distributed Stochastic Neighbour Embedding (t-SNE) is a non-linear technique that projects high-dimensional data to 2-D while preserving local neighbourhood structure [10]. Unlike PCA (a linear projection), t-SNE can unfold curved manifolds and reveal cluster boundaries, making it popular for visualising embeddings. The tradeoff is that absolute distances and axes have no physical meaning and where only the relative grouping of points matters.

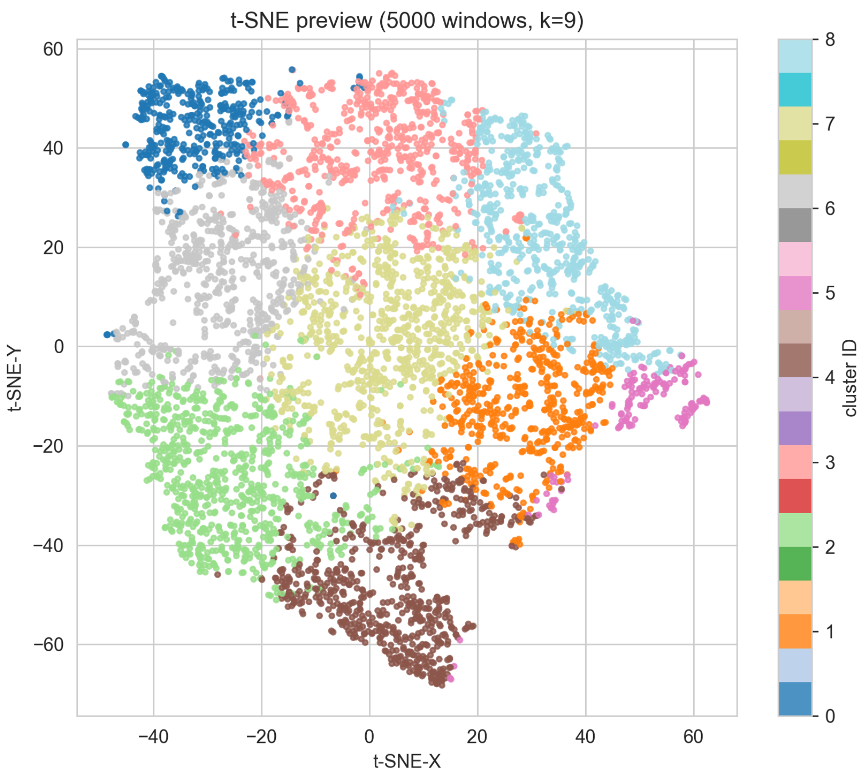

Figure 2: Cluster centroid spectra, illustrating the average frequency content of the audio clips in each cluster. Each colored curve corresponds to one of the 9 clusters (clusters 0–8), showing the mean spectral amplitude across the frequency range (0–100 kHz) for that cluster's centroid.

Some clusters form tight, compact clouds, while others are more diffuse. This indicates that certain types of audio events in the data are very homogeneous (forming dense clusters) whereas others are more variable. Points of the same color generally group together, confirming that the k-means algorithm was able to identify distinct clusters in the feature space. Moderate overlap between a few clusters is present, which could be due to those sound events having more continuous variation or feature overlap (for example, certain clusters may contain very similar kinds of species with relatively similar kinds of calls).

To further clarify which clusters are likely bat calls, we plotted the average peak frequency of each cluster against known bat echolocation frequency bands in Figure 3. We also quantitatively examined the cluster summary metrics below with the help of Table 1 and attempted to interpret the results.

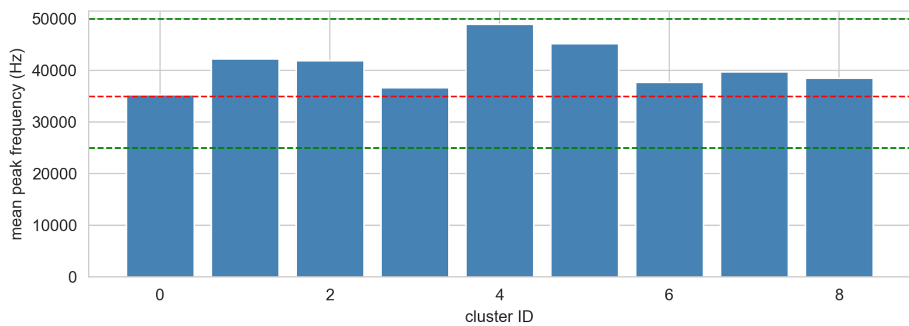

Figure 3: Mean peak frequency for each cluster (clusters 0-8 on the x-axis), with reference lines indicating typical bat call frequencies. The red dashed line (around 35kHz) marks the bottom of the peak frequency range of long-tailed bat echolocation calls [9], and the green dashed lines (around 25kHz and 50kHz) denote the most perceptible harmonics of the lesser short-tailed bat [9].

ID	Peak (Hz)	BW (kHz)	RMS (dB)	Windows	Interpretation
0	35.3k	9.8	-38.2	19,476	LTB quieter echolocation calls
1	42.2k	28.8	-32.0	24,928	LTB loud echolocation calls
2	41.9k	64.4	-39.2	39,372	LTB echolocation calls
3	36.7k	15.7	-33.7	29,124	LTB loud echolocation calls
4	49.0k	74.8	-35.2	32,428	LTB search w/ rain or interference at 33kHz
5	45.2k	55.8	-26.4	7,558	LTB group or very strong echolocation calls w/ rain
6	37.8k	18.5	-38.7	28,027	LTB search and echolocation
7	39.8k	22.9	-35.4	39,156	LTB search calls
8	38.5k	19.6	-30.1	27,495	LTB loud echolocation calls w/ rain

Table 1: Cluster statistics and qualitative interpretation. BW = bandwidth in kHz (mean); RMS = mean window loudness relative to data average. "LTB" = Long-tailed bat *Chalinolobus tuberculatus*.

Upon closer inspection of the cluster statistics, examination of 9 sample members of each cluster (see Figure 4 and 5) and their associated audio recordings, manual qualitative analysis was done to try and match each cluster to a type of call and a species.

As a result of that process, it appears that clusters 0, 1, 2, 3, 6, 7, 8 all indicate the presence mainly of echolocation calls, and more rarely, search calls. Cluster 4 could be characterized by the presence of rain and recordings which feature a continuous uninterrupted tone around the 33kHz frequency mark. Cluster 5 contains very strong calls. They range from calls from possibly multiple bats that overlap, or echolocation calls from a single bat that are very loud or that present a "duplicate" aspect, where one single call seems to contain two calls in one with a few ms of separation.

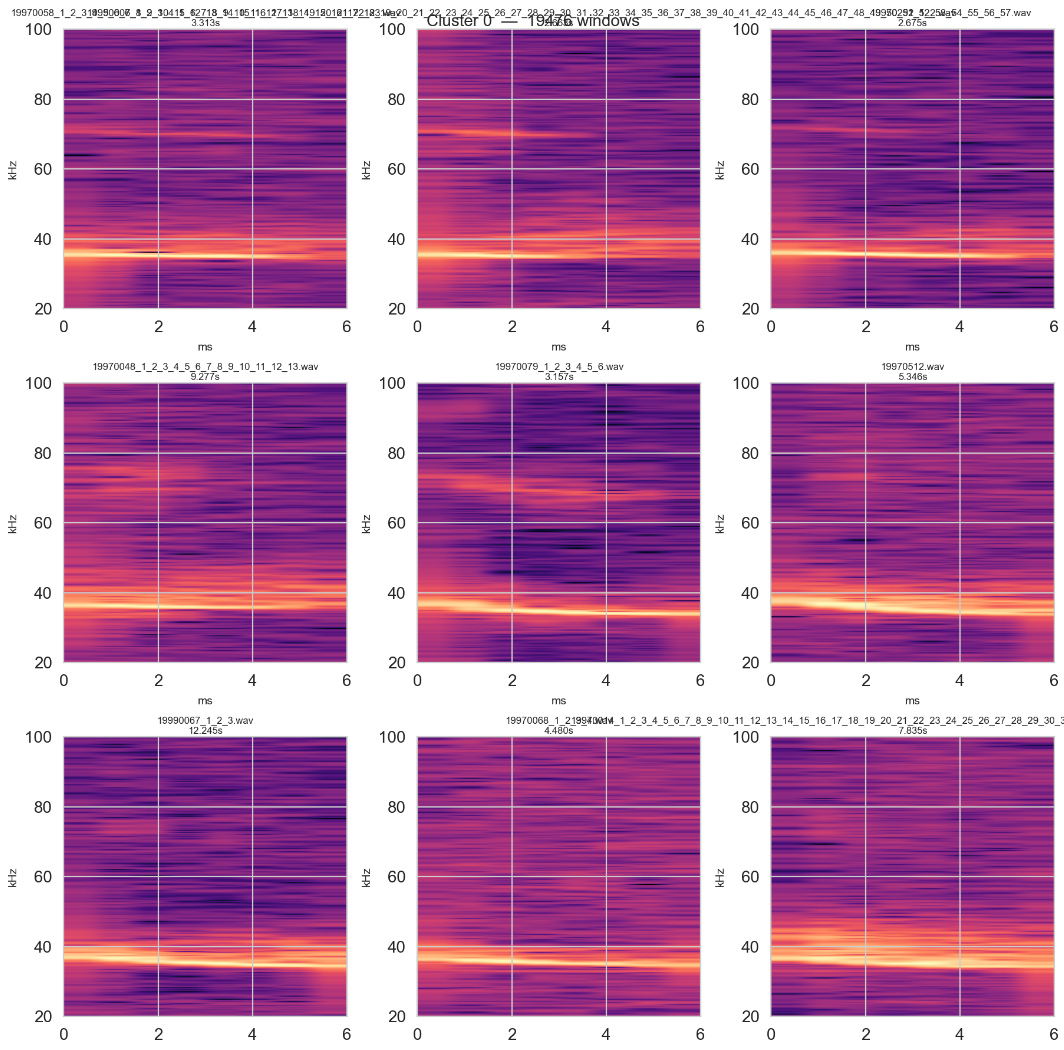

Figure 4: A 3x3 grid of 9 individual windows each from a different recording all mapped to cluster 0. The relative quietness of this cluster's calls and the frequency at which they are hints at these calls being search or echolocation calls from the Long-tailed bat

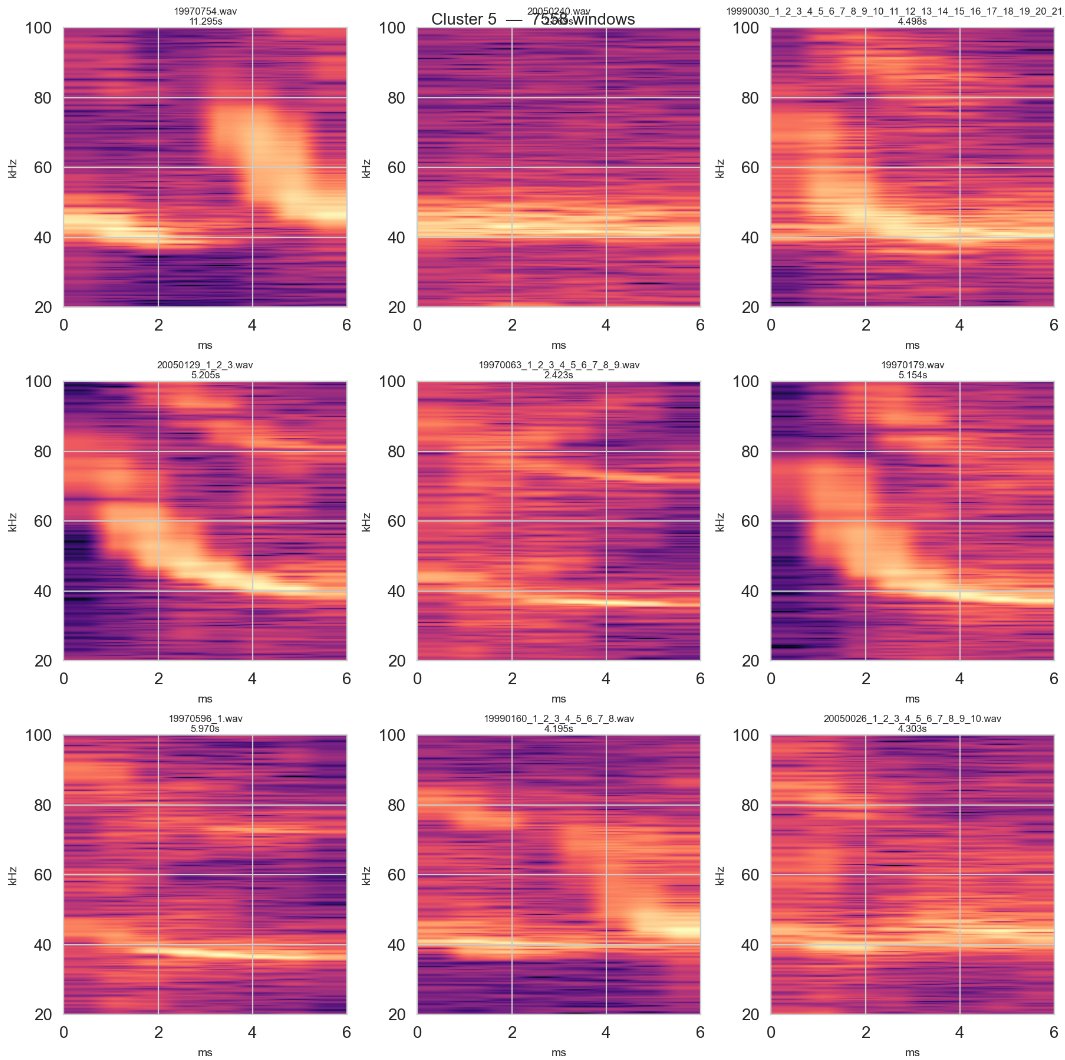

Figure 5: A 3x3 grid of 9 individual windows each from a different recording all mapped to cluster 5. Overlap of descending call frequencies can be observed, indicating potentially multiple overlapping bat calls, or single very loud bat echolocation calls

In all cases, the frequencies of the bat calls are located in the 35kHz to 45kHz range, consistent with the known echolocation and search call frequencies of the Long-tailed bat. There was no indication of any presence of the Lesser short-tailed bat.

However, as discussed in the future work and limitations section (see Section 5), absence of other kinds of calls in our clustering results may simply be explained by the choice of hyperparameter set for this attempt.

In summary, by combining the cluster feature statistics with knowledge of bat call frequencies, we can attempt to infer that most of the clusters are of bat echolocation calls are of Long-tailed bats. This unsupervised separation is an experiment in showing the potential that clustering has to distinguish bat call events from an unlabeled dataset.

5 Future Work and Limitations

While the clustering approach allows us to differentiate many bat calls from noise, there are several areas for improvement.

The current primary limitation facing this approach is the requirement for hyperparameter fine-tuning. We chose a specific filter type, and specific window size. However, all along the process, various different filters and threshold algorithms can be chosen. When sectioning files into windows, various window sizes can be chosen from. Stronger filters may filter out any species that are not as loud as others. They may also exclude distant and quieter calls. This may create an inaccurate representation of the environment we are clustering.

Another area of improvement is in the feature representation. We used spectrograms and features from them, which is effective but could be improved upon. Other learning techniques could yield other types of features that better capture subtle differences between call types and noise. We could also experiment with other clustering algorithms or decision trees to see if other methods are able to capture better information about bat calls.

In terms of noise, treating noise in the same windows (from rain, or wind, or non-bat noises) as additional clusters is potentially confusing. Future work might develop a noise filtering step before clustering to recognize insects, non-bat noises, rain or wind, to help improve model accuracy.

6 Conclusion

All of the nine produced clusters appear to be clusters of sounds made by the Long-tailed bat *Chalinolobus tuberculatus*. The clustering method does not separate species, requires manual interpretation and could be biased by our choice of hyperparameters. It demonstrates that an unsupervised model can faciliate classification of large scale eco-acoustic data.

Part III
Web Application *Pekapeka*

Project currently live via: waraki.wickerlab.org

7 Introduction

Waraki is an app for individuals and researchers operating in the field of acoustic ecology (eco-acoustics) to upload, store and share their audio recordings with other users. This web application addresses storage, sharing and collaboration needs of eco-acoustics-focused scientific communities. It was developed initially to enable collaboration between the Community Waitakere group and bat-focused researchers at the University of Auckland.

8 Domain and Requirements

8.1 Functional Requirements

Upload, Play and Browse Audio Files

1. Users can upload single/multiple audio files or ZIP archives via a dedicated upload page.
2. Users can view all files in their personal library and see length, size, format, date and name.
3. Users can download or delete any audio they have uploaded.

Share Audio Files via Posts

1. Users create a *Post* to share files with the public and supply quantitative/qualitative metadata.
2. Users can browse or filter community posts.
3. Users can comment on and like other users' posts.

8.2 Non-Functional Requirements

High Availability and Reliability

1. Deploy in a stable environment with sufficient compute resources.
2. Incorporate redundancy in storage and database design to minimise data loss.

Low Latency and High Performance

1. Optimise routes, objects and queries for performance.
2. Plan for future horizontal scaling.

9 Users and Actors

Researcher

Typically works with gigabytes or possibly terabytes of data and requires a reliable platform for eco-acoustic storage.

Wildlife Enthusiast

Enthusiasts or students who capture megabytes or possibly gigabytes of recordings and share them via Posts.

Administrator or developer

Maintains application stability, helps users, updates application, takes feedback

10 Architecture Overview

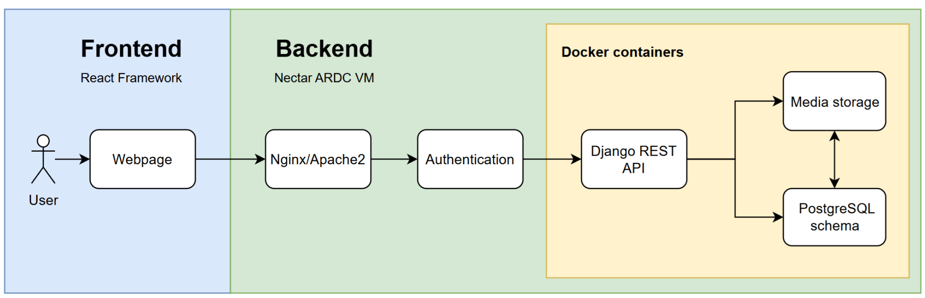

Figure 6: High-level overview of the flow of action in the web application

Frontend

Built with React (JS), chosen for its popularity and long-term maintainability.

Backend

Implemented in Django REST, which accelerates CRUD development and provides WebSocket support for future live-streaming functionality.

Database

Relational schema in PostgreSQL; ACID properties ensure data integrity.

Hosting

Containerised deployment (frontend, backend, database) on a Nectar ARDC VM (Ubuntu 22.04, 16 cores, 32 GB RAM, 500 GB SSD).

11 Database Model

Schema / ERD

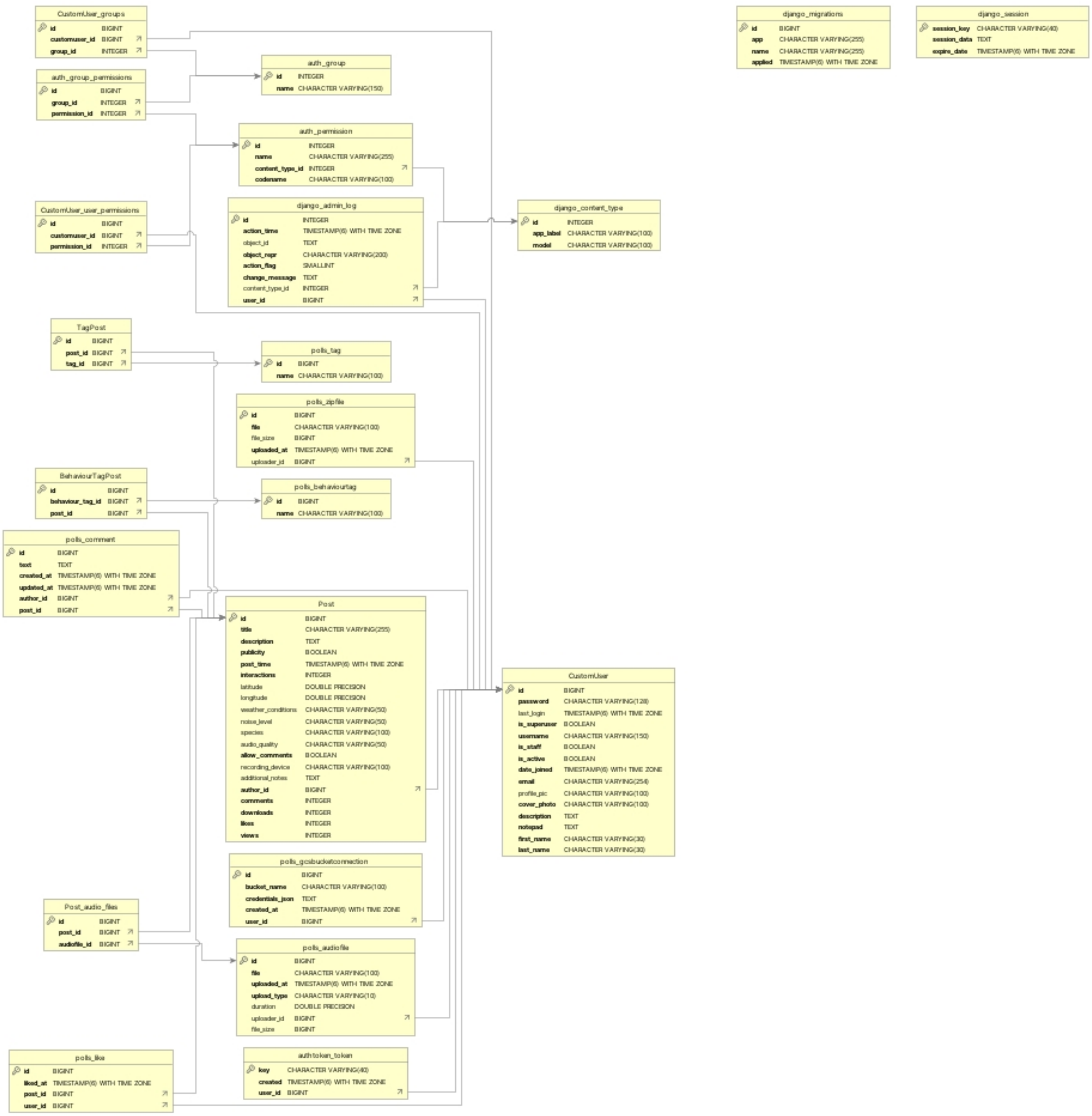

Figure 7: Diagram of the current relations in the PostGreSQL database

Main Entities

Entity	Attributes
polls_audiofile	id, title, uploaded_at, file_key, duration, file_size, upload_type, uploader_id
CustomUser	id, password, last_login, is_superuser, username, first_name, last_name, is_staff, is_active, date_joined, email, profile_pic, cover_photo, description, text, etc.
Post	id, title, description, publicity, post_time, interactions, latitude, longitude, weather_conditions, noise_level, species, audio_quality, allow_comments, recording_device, additional_notes, author_id, comments, downloads, likes, views
polls_zipfile	id, file, file_size, uploaded_at, uploader_id

12 API Surface

User / Account Routes

- `POST /register/` – Register a new user
- `POST /login/` – Log in and receive a token
- `POST /logout/` – Log out (invalidate token)
- `GET /profile/get` – Get current profile
- `POST /profile/edit` – Edit profile
- `POST /profile/notepad` – Update notepad
- `GET /user/info/<id>` – Public profile by ID
- `GET /user/check_username` – Check username availability

Audio / ZIP File Routes

- `GET /audio/my_uploads/` – List user uploads
- `DELETE /audio/delete/<id>/` – Delete audio
- `GET /audio/<id>/download/` – Download audio
- `POST /audio/<id>/stream/` – Stream audio
- `POST /upload/bulk` – Bulk upload
- `POST /upload/chunk` – Chunk upload
- `POST /upload/complete` – Finalise chunks
- `DELETE /zip/delete/<id>/` – Delete ZIP
- `GET /zip/<id>/download/` – Download ZIP

Post Routes

- `POST /post/create` – Create post
- `PUT /post/edit` – Edit post
- `DELETE /post/delete` – Delete post
- `GET /post/get` – Search posts
- `GET /post/get_all` – All post IDs
- `GET /post/get_all_objects` – All posts detailed
- `GET /post/get_by_id` – Post by ID
- `GET /post/get_post_by_tag` – Posts by tag
- `GET /post/search` – Advanced search
- `GET /posts/trending` – Trending posts
- `GET /post/species` – Species list
- `GET /post/recording_devices` – Device list
- `POST /post/<id>/like` – Like post
- `POST /post/<id>/unlike` – Unlike post
- `POST /post/<id>/comment` – Add comment
- `GET /post/<id>/get_comments` – Post comments
- `GET /post/tags` – All tag names

13 Deployment and Operations

Environment

Ubuntu 22.04 LTS VM (16vCPU, 32GB RAM, 2TB Disk) on Nectar ARDC.

Current Deployment Flow

(on the VM, in opt/waraki_codebase/waraki) git pull -> docker-compose down -> docker-compose up --build

14 Future Work

Building on the requirements already identified in the PDF, the following concrete tasks could be planned:

CI/CD and Quality Assurance

- Establishing a GitHub Actions pipeline that runs ESLint, Pytest and various tests on every push/pull request, followed by automatic deployment
- Adding versioned Docker images and storing them for redundancy.

Performance and Scalability

- Introducing caching for frequently requested assets (e.g. profile pictures, small MP3 previews) to prevent large wait times when users try to access their files if they have a lot of them.

Sharing & Permissions

- Implementing file-level permission so that private uploads can be selectively shared with collaborators or project groups. This will be very important for facilitating data sharing between users. Currently, users have no real way of sharing files between themselves.
- Improving role-based access control to better distinguish administrators/developers, researchers and public users.

Classifier Integration

- Implementing an inference service that receives spectrogram images or other kinds of features and returns information about the provided bat audio data.
- Exposing the service through the existing API and overlay detections on the front-end spectrogram viewer.

Data Governance and Observability

- Improving account management and password/authentication management.
- Implementing database backups and retention rules. Currently there are no redundancies for user data.

15 Conclusion

Waraki is at its core a repository where eco-acoustic data can be stored and used immediately in research. Next steps involve live testing and integration into researchers' daily workflow. Completion of the points mentioned in the Future Work section would help turn Pekapeka from a functional prototype into a stable and scalable platform.

References

1. "Bats/pekapeka." Department of Conservation (DOC). https://www.doc.govt.nz/nature/native-animals/bats-pekapeka/ (accessed May 10, 2025)

2. "Sending up the bat signal: inside the world of bat monitoring." Predator Free NZ. https://predatorfreenz.org/stories/sending-up-the-bat-signal-inside-the-world-of-bat-monitoring/ (accessed May 23, 2025)

3. A. Farina, B. Krause, T.C. Mullet, "An exploration of ecoacoustics and its applications in conservation ecology", *BioSystems*, Volume 245, Nov. 2024,

4. V. Zamora-Gutierrez *et al.*, "The evolution of acoustic methods for the study of bats," in *50 Years of Bat Research: Foundations and New Frontiers*, Springer, 2021, pp. 43–59.

5. C. Roemer *et al.*, "An automatic classifier of bat sonotypes around the world," *Methods Ecol. Evol.*, vol. 12, pp. 2432–2444, 2021.

6. O. Mac Aodha *et al.*, "Bat detective—Deep learning tools for bat acoustic signal detection," PLoS Comput. Biol., vol. 14, no. 3, Art. no. e1005995, Mar. 2018, doi: 10.1371/journal.pcbi.1005995

7. D. A. Nieto-Mora *et al.*, "Systematic review of machine learning methods applied to ecoacoustics and soundscape monitoring," *Heliyon*, vol. 9, Art. e20275, 2023.

8. "Welcome to Community Waitakere." Community Waitakere. https://www.communitywaitakere.org.nz/ (accessed Jun. 10, 2025)

9. B. Lloyd, "Bat Call Identification Manual for DOC's Spectral Bat Detectors," Lloyd's Ecological Consulting, Wellington, New Zealand, technical manual, Jun. 2017.

10. L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

11. M. J. Guerrero *et al.*, "Acoustic animal identification using unsupervised learning," *Methods Ecol. Evol.*, vol. 14, no. 4, pp. 934–946, 2023.

12. A. E. Noble *et al.*, "Unsupervised clustering reveals acoustic diversity and niche differentiation in pulsed calls from a coral reef ecosystem," *Frontiers Remote Sensing*, vol. 5, 2024.