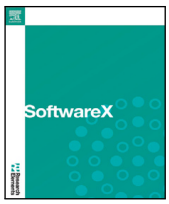
SoftwareX 25 (2024) 101606

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx



Original software publication

Joseph the MoUSE — Mouse Ultrasonic Sound Explorer

Adam Kania [a,1], Weronika Ormaniec [b,1], Dmytro Zhylko [c,1], Leszek Grzanka [d], Diana Piotrowska [e], Andrzej Siódmok [a,*]

[a] Jagiellonian University, Kraków, Poland
[b] ETH Zürich, Switzerland
[c] AGH University of Science and Technology, Kraków, Poland
[d] Institute of Nuclear Physics, Polish Academy of Sciences, Kraków, Poland
[e] Maj Institute of Pharmacology, Polish Academy of Sciences, Kraków, Poland

ARTICLE INFO

Keywords:
Object detection
Image processing
Ultrasonic vocalizations

ABSTRACT

Joseph the MoUSE — Mouse Ultrasonic Sound Explorer (MoUSE) software aims to address the issue of manual analysis of recordings from experiments on rodents by introducing automatic techniques for ultrasonic vocalization (USV) detection. It combines deep learning (DL) methods with classical pattern recognition and computer graphics algorithms. During development, we used a dataset that consisted of recordings from real-world experiments in the open field. Recordings like these pose obstacles to automatic USV detection, one of which is the noise produced by mice in the experimental area or in nearby cages. Therefore, additionally, we conducted research and implemented de-noising methods along with detection algorithms. The project includes Python packages with algorithms for sound noise removal and USV detection, and provides a user-friendly graphical interface.

Code metadata

Current code version	1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00410
Permanent link to Reproducible Capsule	
Legal Code License	MIT
Code versioning system used	git
Software code languages, tools, and services used	Python
Compilation requirements, operating environments & dependencies	Python 3.8
If available Link to developer documentation/manual	github.com/JosephTheMoUSE/MoUSE-docs/wiki
Support email for questions	josephthemouse@googlegroups.com

1. Motivation and significance

Rodents such as mice and rats play an essential role in biomedical research. They are the subject of experiments that allow scientists to observe the course of various diseases close to the human environment. Because of this, they constitute a significant part of the drug discovery process and psychological research. In the latter case, key symptoms cannot be observed conventionally because they usually do not affect the physical state of the rodent. That is why other methods of measuring the well-being of animals have been developed. Several research articles [1–6] confirm that ultrasonic vocalizations (USVs) of rodents provide the possibility to assess their condition and emotional state. These vocalizations typically fall in the 20–90 kHz frequency range and can be divided into two main categories: alarm calls and appetitive calls. Animals make emergency calls when they sense danger or as an effect of a negative stimulus. In contrast, rewards and positive social interactions trigger appetitive calls. Duration of vocalization, frequency modulation, and bandwidth categorize these two main groups further [7].

* Corresponding author.
E-mail address: andrzej.siodmok@uj.edu.pl (Andrzej Siódmok).
[1] These authors contributed equally to this work and share first authorship.

https://doi.org/10.1016/j.softx.2023.101606
Received 2 July 2023; Received in revised form 30 November 2023; Accepted 3 December 2023
Available online 16 December 2023

The typical way to collect data on rodent vocalizations is to record them using an ultrasound microphone (e.g. Avisoft-Bioacoustics, CM16/CMPA). Analysis of recordings involves searching and categorizing vocalizations, with a particular emphasis on disregarding laboratory noise. In an experiment conducted in the Maj Institute of Pharmacology of the Polish Academy of Sciences, where Mouse Ultrasonic Sound Explorer (MoUSE) was first used, rats were placed on an open field arena measuring approximately 55 × 55 cm. Typically, two unfamiliar rodents are placed in the arena for a 10-min social interaction test. An ultrasonic microphone that operates in the range from 10 kHz to 120 kHz was located about 20–30 cm above the box. The microphone recorded rodent sounds overlaid with many background noises, such as human speech, high-frequency sounds generated by a monitor, sounds generated by the animals waiting in their home cages and by the tested rats themselves, e.g., scratching with their claws. After the recordings are collected, the researchers usually manually select and classify the vocalizations of the rodents. This process is laborious and tedious and requires a significant amount of the researcher's time, which is its major disadvantage. This is, in fact, the main reason why researchers often abandon recording vocalizations and fall back to techniques based only on video recordings. Sometimes, custom-made software is used to help with the USVs' analysing process [8]. Joseph the MoUSE — Mouse Ultrasonic Sound Explorer was created to facilitate, speed up, and automate this process. It should be noted that many efforts have been made to automate this process [9–15]. For example in [16], the authors of DeepSqueak used deep learning methods to identify and classify USVs. A detailed comparison of various programmes for vocalization analysis is found in [17,18].

Among the already existing solutions used for USV analysis the most common tools are Avisoft SasLab Pro [19] and Raven Pro [20]. Despite their widespread usage, existing methods suffer from limited support for automated detection and classification. Furthermore in experiments involving open-field settings low-frequency calls can be easily mistaken for background noise, even by human annotators, leading to inaccurate results.

Other software created to tackle the problem of USV detection includes WAAVES [21] and USV detection using template matching [22]. Both of them achieve great success in the experiments described in the original articles but still require manual tuning, which means that they do not generalize across a wide spectrum of experiments. Lastly, there is the software application known as DeepSqueak [16] which uses the power of Deep Learning and Computer Vision to replace human visual analysis of spectrograms.

MoUSE brings the benefits of the OpenSource software: the code is free to use and relies on a freely available library with a permissive licence. It can be deployed and installed in most laboratories without the need to buy any additional software.

2. Software description

We chose to use spectrogram as our sound data representation, since it is the most descriptive format to analyse by humans and is the format in which data was originally analysed in. This however poses technical challenges, due to the size of the input spectrogram (tens of gigabytes per typical experimental day), efficient processing of multiple recordings can pose significant computational challenges. Therefore, our proposed solution is designed to be compatible with an average PC, ensuring that the processing time remains reasonable. Our approach involves the utilization of our own dedicated library with specialized algorithms for audio denoising, ultrasonic vocalization (USV) detection, and classification. This library is seamlessly integrated into an application with a graphical user interface, providing a user-friendly way for conducting these tasks. The application can be found in our GitHub repository together with instructions on how to install it (see Code metadata).

The primary purpose of the application is to aid researchers in the analysis, detection, and classification of USVs in the audio recordings collected during experiments involving rodents.

2.1. Software architecture

We have chosen to split the codebase into a couple of separate repositories grouped under the GitHub organization, https://github.com/JosephTheMoUSE. Major repositories are GUI application, core library and documentation. The repositories are linked by the mechanism of git submodules.

The source code was written in Python programming language, as currently, it seems to be the best solution for data analysis and machine learning projects. Python provides a rich ecosystem of machine-learning related libraries and has decent support for modern cross-platform GUI libraries such as Qt6. The popularity of Python (at the time of writing) simplifies the process of code development and its maintenance.

The core library contains code responsible for major features of MoUSE. This includes audio denoising, USV detection and classification. The architecture of the GUI application follows the popular and successful Model-View-Controller software design pattern [23]. Apart from application source code, both repositories contain configuration files for code formatters and unit tests covering crucial parts of the code.

The main flow of application execution is composed of steps reflecting a typical data analysis process: data loading, denoising, USV detection and classification, filtering and exporting the results (see Fig. 1). Each step of the analysis process is pre-configured with default parameters, which can be easily adjusted by the user. Denoising and detection configuration screens contain a recording preview. This preview visualizes the results of applying a given method, thus providing quick feedback about the selected configuration (see Fig. 2). This is especially helpful, as the user can select a particularly problematic fragment of the recording for configuration tuning.

2.2. Software functionalities

The MoUSE platform can be run on Windows and Linux operating systems. It reads audio data in WAV format and exports detected and annotated USVs in comma-separated values (CSV) format accepted by most data analysis tools. It supports displaying the spectrogram and provides basic operations on it. For example, the user can manually select a fragment of a spectrogram and edit or delete previous selections. A more detailed explanation of all the features can be found in the MoUSE platform documentation (see Code metadata).

In the following section, we outline the key features of the MoUSE platform. Note, that all presented algorithms utilize spectrogram representation of sound data.

2.2.1. Denoising

The MoUSE platform supports three audio denoising methods:

- Bilateral filter is a standard, non-iterative denoising algorithm, which also found its application in bioacoustics [24]. This algorithm can smooth a spectrogram while preserving the edges. Bilateral filter substitutes the intensity value of each pixel with a weighted average of intensities derived from neighbouring pixels. The weight is determined by the Euclidean distance between the pixels in both the spatial and intensity domains.
- Short duration transient suppression (SDTS) filter is a Gaussian-filter-based algorithm that was primarily used for denoising dolphin whistle recordings. According to [24] this method should remove vertical line patterns in the spectrogram. The working principle of the SDTS filter can be explained as follows: If the predominant pattern within the pixel's vicinity is vertically oriented, the pixel's value will be reduced. The determination of direction significance relies on the analysis of intensity relationships in four intermediate images generated through modified Gaussian filters. Each filter intensifies pixels being part of a pattern oriented in one of four main directions: vertical, horizontal and two diagonal ones.

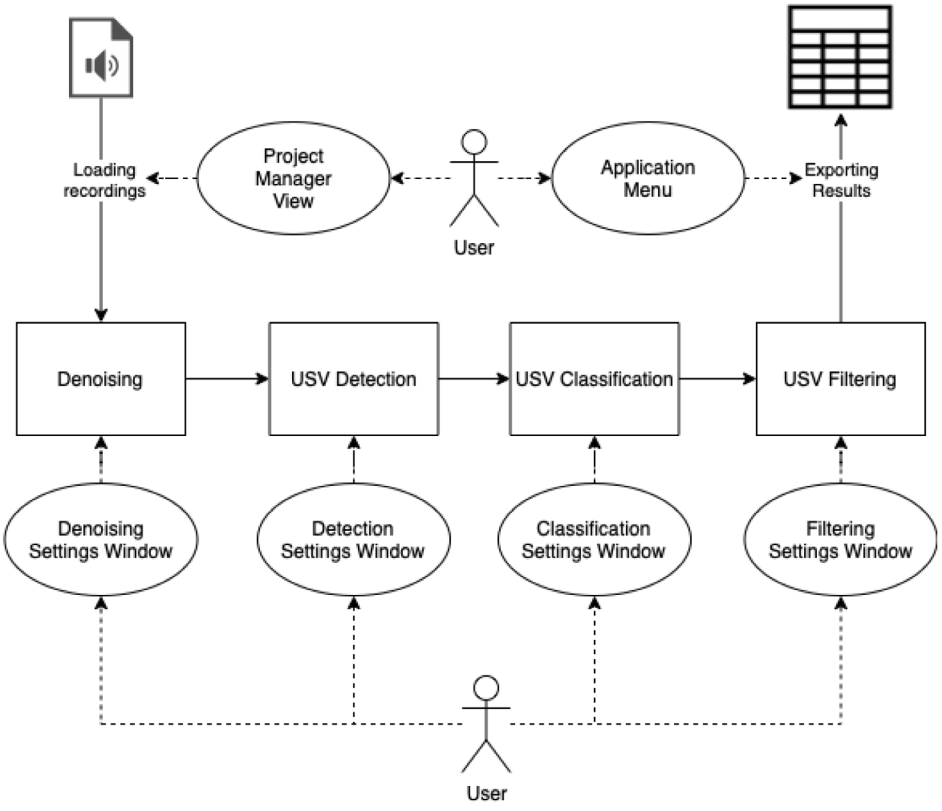


Fig. 1. MoUSE application flow.

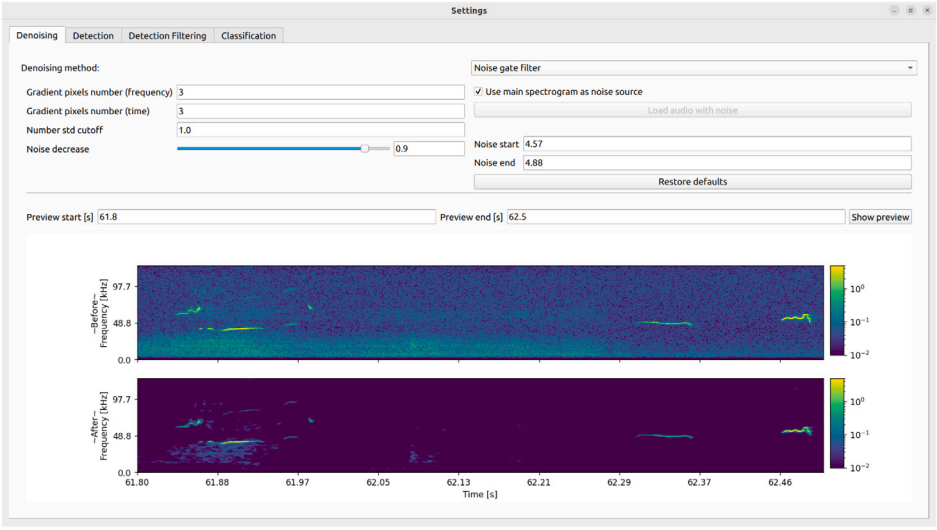


Fig. 2. Denoising settings. The top graph displays the raw spectrogram with USVs and noise in green colour. The bottom graph represents the same spectrogram processed with a noise gate filter, with the configuration presented in the setting panel above the spectrograms. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- Noise gate filter is an algorithm we implemented from scratch based on its description on the Audacity website [25]. This method needs to be initialized by the user with a fragment of a noisy spectrogram which is used to estimate noise statistics. These statistics are used to compute noise thresholds for each spectrogram frequency band. If the pixel intensity value is smaller than the corresponding threshold, it gets attenuated to the minimal value found in the input spectrogram.

2.2.2. Detection

The MoUSE platform offers two distinct approaches for detecting USVs. The first approach involves the use of Morphological Geodesic Active Contour (GAC) [26], which allows users to manually fine-tune the algorithm for optimal precision and recall. Alternatively, users can opt for the Neural Network approach, which relies on a learned model but may not perform optimally in cases where there is a significant

Table 1
Global coverage. All values are multiplied by 100 for better visibility.

	FRCNN	GAC 1	GAC 2	GAC 3	GAC 4	GAC 5
Coverage of ground truth by detected	80.95	47.14	47.34	55.30	55.32	55.92
Coverage of detected by ground truth	68.25	76.68	77.44	60.48	62.02	39.90

Table 2
Global recall and precision. IoU threshold: 0.1. All values are multiplied by 100 for better visibility.

	FRCNN	GAC 1	GAC 2	GAC 3	GAC 4	GAC 5
Recall	97.91	80.46	80.67	70.31	70.31	73.95
Precision	81.28	77.70	78.06	86.04	87.19	83.73
F1	88.46	78.42	78.76	75.88	75.99	78.24

disparity between the training data and the data being analysed. Further information on these approaches can be found in the respective paragraphs.

Morphological geodesic active contour (GAC). GAC is an algorithm that can find image areas with high gradient. The main advantage of this method is that it does not require training. The fact that this method is non-trainable creates the need for configuring the method for the user. To make the configuration more transparent and easy to use, MoUSE provides a preview on which the configuration can be tested. The preview shows a user-selected fragment of the spectrogram, on which each iteration of the algorithm is visualized. After the method finishes execution, detected USVs are shown on the preview.

An alternative method for autoconfiguration of GAC is also available. The method uses a fragment of an annotated spectrogram (e.g., 10 s) and Bayesian optimization in order to find parameters that maximize a selected metric.

Neural network approach. Faster R-CNN [27] — currently the only supported architecture by MoUSE. This is a neural network architecture trained end-to-end with a convolutional NN as a backbone. The model is capable of working without user input and is suitable for preliminary analysis of large amounts of recordings. Feature extractor size and architecture choice are mainly dictated by computational resources, so MoUSE may be used on middle/high-end consumer-grade PCs. We experimented with different variants of ResNet [28] model and settled with a modified resnet-50 variant with half of the blocks removed, essentially leaving 21 convolutional layers across 2 stacks of bottlenecks, to produce wider spacial feature maps, while being memory efficient and capturing different features with increased channel number, for better detection. Increasing the size of the model and/or reducing spacial reduction would improve detection performance, but was prohibitively expensive to train and inference with. Final results are presented in Tables 1 and 2 (labelled FRCNN). We also present the capabilities of FRCNN classification head, which is a part of the detection pipeline, in Fig. 4. We detail our training procedure and hyperparameter choice in Appendix C. Apart from the standard steps of training object detection neural network, we used a custom data augmentation technique, that exploits the fact of vocalizations being time-invariant, so we could safely generate more examples by "sliding" existing boxes across time domain, with the restriction of not causing any unintentional intersections between annotations.

2.2.3. Classification

The MoUSE Platform supports two kinds of classification methods:

- Simple classification — it classifies USVs solely on their mean frequency, which is enough to distinguish emergency calls (22 kHz; near-constant frequency calls between 20 and 25 kHz) from appetitive calls (the 50 kHz, 'happy' calls of much more heterogeneous structure, appearing between 30 and 90 kHz).
- Advanced classification — it is currently supported only as a part of the neural network-based detection algorithm. The classification module is end-to-end learned for detection purposes during training. In advanced classification, the appetitive ('happy') calls are further divided into the following sub-categories: short (*sh*; duration less than 12 ms), flat (*fl*; calls with a near-constant frequency, longer than 12 ms), one-component (*oc*; monosyllable calls), multi-component (*mc*; calls consisting of at least two syllables), trills (*tr*; very highly modulated calls, most characteristic category, consisting of several spike-like syllables), complex trills (*trc*; trills with elements of other call categories e.g. fl or oc). The choice and description of call categories were based mostly on [29].

2.2.4. USV filtering

In our platform, we currently have two options for filtering incorrect detections:

- Rule-based — remove USVs that are mostly (more than half of the box) below a frequency threshold. This can help with removing false positives in low-frequency ranges (scratching, music, etc.)
- Neural Network based — binary classification network trained to distinguish between noise and USVs.

3. Illustrative examples

3.1. Using MoUSE desktop application

In this section, we present a general example of how the MoUSE platform is used. We discuss:

- how the spectrogram can be preprocessed,
- how detection and classification of USVs can be configured,
- what can be done with the results.

For a more detailed usage description please refer the documentation (https://github.com/JosephTheMoUSE/MoUSE-docs/wiki). Additionally, we attach an explanatory video that shows all the steps described in this section (see https://youtu.be/qmGrD-POp10).

After audio files are loaded into MoUSE, a user can proceed to configure audio denoising. Each of the previously described audio-denoising methods can be tested on a user-selected fragment of the spectrogram. In Fig. 2 a spectrogram fragment denoised with a noise gate filter is presented (bottom one).

When the amount of noise in the spectrogram is at a satisfactory level, a user can proceed to the configuration of USV detection. MoUSE platform provides a preview spectrogram on which detection methods can be tested. In Fig. 3 we present an example of a configuration window with GAC method selected for detection.

The MoUSE platform incorporates a configurable filtering step, allowing users to eliminate false detections. Users are offered both frequency-based and neural network-based approaches for this purpose.

Users are also provided with a USV classification configuration window which is used in cases where not only detection, but classification

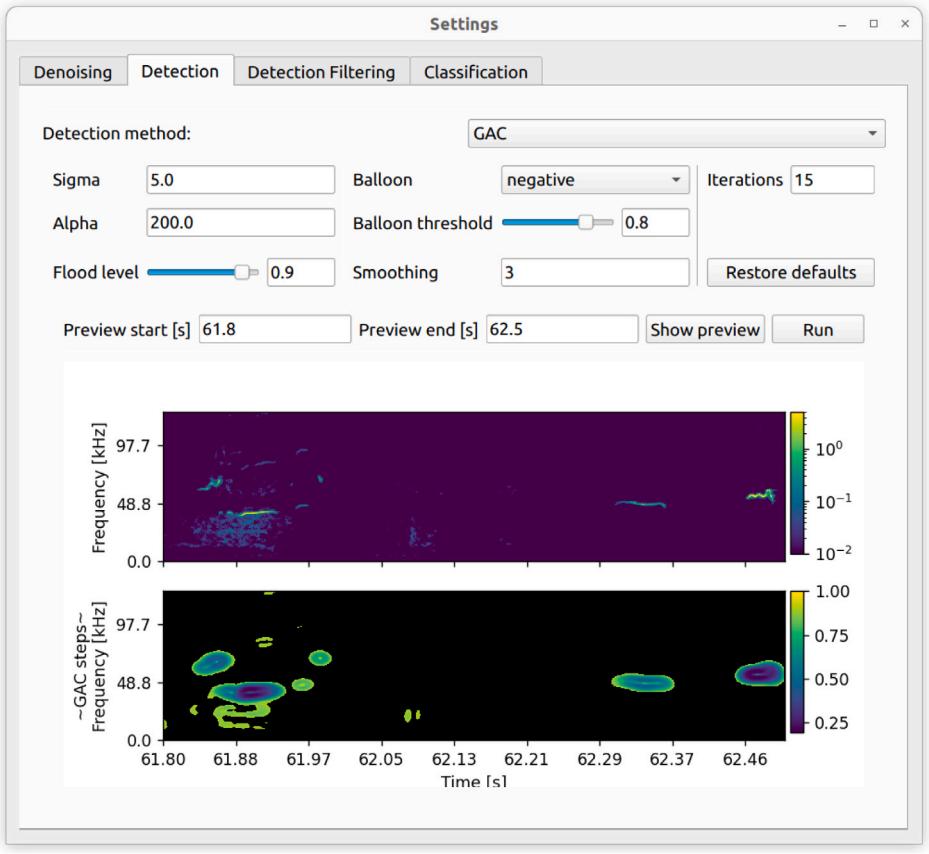


Fig. 3. Detection settings on the "GAC" page. The top graph displays a denoised spectrogram with several USVs. The bottom graph illustrates the initial state of the active contour algorithm's bitmap, which is iteratively updated during the execution of the algorithm. The final state of this bitmap corresponds to the algorithm's 'detections'.

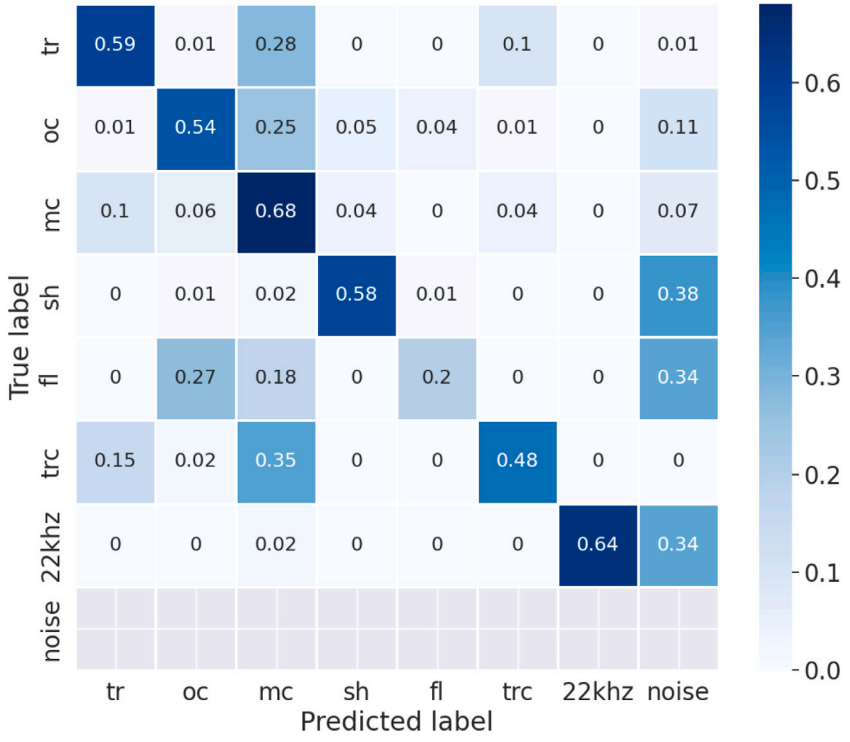


Fig. 4. Confusion matrix for Faster R-CNN classification on the test set.

Table 3
Recall per vocalization category. IoU threshold: 0.1. All values are multiplied by 100 for better visibility.

	FRCNN	GAC 1	GAC 2	GAC 3	GAC 4	GAC 5
tr	83.33	79.21	79.21	74.31	74.31	78.74
oc	77.46	74.53	74.53	67.06	67.06	66.50
mc	98.34	86.35	87.08	83.12	83.12	79.15
sh	99.51	69.02	69.02	61.42	61.42	45.32
fl	33.33	25.69	25.69	27.78	27.78	23.61
trc	66.67	56.33	56.33	57.78	57.78	57.78
22 kHz	31.94	15.28	15.28	0.00	0.00	23.61

of each call is necessary. Currently, users can select between a simple rule that can differentiate between emergency and appetitive calls and a neural network-based classification algorithm.

Detection and classification results can be fine-grained by the user manually through interacting with graphical annotation representation (bounding boxes) and through changing values in the annotations table. When the user is satisfied, MoUSE allows exporting data to a tab-separated values (TSV) file, including results and metadata.

3.2. Performance of detection and classification algorithms

In this section, we evaluate the performance of the algorithms supported by MoUSE. The dataset used for our analysis is a 6-min recording with manually selected and classified vocalizations that we treat as ground truth. A manual selection is a rectangular bounding box around the vocalization. Similarly, our algorithms also return rectangular bounding boxes.

MoUSE offers multiple configurations of algorithms for denoising and detection. In this section, we present results for the following configurations: FRCNN, GAC 1, GAC 2, GAC 3, GAC 4, GAC 5 all defined in Appendix A. The first metric we present here is the bounding box coverage. Having two bounding boxes A and B we define coverage of A by B as $cov(A, B) := \frac{A \cap B}{A}$. In Table 1 we present the coverage of ground truth bounding boxes by detected bounding boxes and vice versa for the experiments described above. One can notice that FRCNN is better at creating bounding boxes that cover ground truth bounding boxes than any GAC configuration we tested. It is expected behaviour as FRCNN interacts with human-made annotations during training. GAC on the other hand is a fully unsupervised method. By comparing these 2 types of coverage we can conclude that GAC in general creates smaller bounding boxes than FRCNN or human annotators. For the purpose of further analysis, we define Intersection over Union (IoU) between 2 bounding boxes A and B as $IoU(A, B) := \frac{A \cap B}{A \cup B}$. Now given a threshold T we can define the following categories for bounding boxes:

- TP (true positives) — a bounding box A returned by the detection algorithm is considered a true positive when there exists a ground truth bounding box B such as $IoU(A, B) \geq T$.
- FP (false positives) — a bounding box A returned by the detection algorithm is considered false positive when there is no ground truth bounding box B such as $IoU(A, B) \geq T$.
- FN (false negatives) — a ground truth bounding box B is considered a false negative when there is no bounding box A returned by a detection algorithm such as $IoU(A, B) \geq T$.

Using these definitions, in Table 2 we report global precision, recall and F1 scores achieved by different algorithms. Because of the discrete nature of the spectrogram and annotation process, we chose a low threshold to fairly compare different methods. We observed that GAC tends to produce the smallest bounding boxes. This is a desirable trait but tends to be unfavourable in cases where small inconsistency in annotations based on discrete pixels (due to potentially different spectrogram configurations and/or manual annotation) constitutes a substantial difference in bounding box surface area, like extremely long or extremely short USVs. In this section we present results for threshold $T = 0.1$, extended results are presented in Appendix B. Hence, we consider even a small intersection over union a success. One can see that all the methods achieve a rather high global recall with FRCNN performing the best. Further analysis reveals that we, in fact, have 3 distinct families of methods. The first is a neural network with a great capacity to learn from annotated data, which produces the best results but may not be transferable to out-of-domain data. And second and third families are represented by different parametrizations of GAC method. GAC allows more control over recall vs. precision trade-off and is highly adaptable. We observe GAC 1 reaching high recall, while additional postprocessing filtering helps boost the precision of the model without hurting recall (GAC 2). In GAC 3–4 we can observe the same effect of postprocessing. But with the addition of noise-gate filtering, we can significantly increase recall as demonstrated by GAC 5, by denoising low-frequency range and uncovering 22 kHz USVs covered in noise. Filtering, however, can also introduce some issues with short (sh) USVs, making them less visible or removing them entirely, so it should be used with consideration and might require some experience to properly work. All effects on recall caused by denoising can be observed in Table 3. In this table, we have categorized the automatically detected bounding boxes according to the labels of the ground truth bounding boxes with which they intersect the most.

Finally, in order to quantify Faster R-CNN classification performance we define predictor as a bounding box with the greatest IoU, but no less than 10% relative to each ground truth box. In this case, a correct prediction is a predicted bounding box with the same label as the corresponding ground truth box. In cases where no prediction boxes satisfy the IoU requirement, we consider the predicted label to be “noise”. In Fig. 4 you can find the confusion matrix for the Faster R-CNN model. As expected, the model performs well for distinct classes and we observe an accuracy drop for classes that can be misclassified by human annotators. For example, mc and tr vocalization appear to be very similar, which is reflected in many confusions. Also, a performance drop is observed in low cardinality classes, which is expected considering that Neural Networks learn from data and tend to perform worse for underrepresented classes.

In summary, MoUSE presents users with an easy-to-use black box solution, that will work very well for users with a similar experimental setup and a manual human-in-the-loop method, which requires some degree of expertise but can perform just as well. MoUSE also provides methods that can boost the performance of detection algorithms.

4. Impact

Measuring rodents’ ultrasonic vocalizations has become a popular tool in behavioural research within the last two decades, but the currently used methods have some serious limitations. As mentioned above, the analysis of rodent USVs recordings is usually done manually and consumes a large amount of time. This fact discourages many laboratories from recording vocalizations. MoUSE was created to facilitate, speed up, and automate this process. Therefore, the programme will allow the exploration of novel research enquiries that were previously unattainable due to limited resources.

Recording ultrasonic communication is most commonly done in research focused on the sociality and sociability of laboratory rodents, but USVs provide insight also into the emotional state of these animals, making USVs recordings a useful tool for monitoring their well-being. Understanding rodents' emotional state is important not only in the fundamental research but also during the preclinical development of new drugs for psychiatric disorders, for example, lowering anxiety levels or aversion to certain situations will have an impact on rodents' vocalizations [30]. The pattern of emitted USVs also changes under the influence of drugs of abuse such as amphetamine, cocaine or morphine (for a review, see [31]). By recording and analysing USVs during free exploration of testing apparatus, researchers can, for instance, gain insight into individual tendencies of animals to vocalize, differentiating between low and high vocalizers [32,33]. Through automated analysis, researchers can obtain valuable data that can contribute to a better understanding of animal behaviour. Our tool contributes to this research in two key ways. First, it is open access, ensuring accessibility to a wider audience. Second, it provides two distinct methods for USV detection: a more advanced Faster R-CNN and a simpler one named GAC. GAC does not require a manually annotated dataset, making it adaptable across various experiments. Additionally, our tool can serve as an initial detection tool, offering potential value in preliminary detection tasks.

Another goal of MoUSE is to provide a flexible system for the automated classification of detected USVs. This is important because the calls differ from each other and their 'meaning' is probably also different. Typically, the USVs are analysed and classified by the experimenter or by students, sometimes by several people per one larger project. This carries the risk of differences in classification by different people since there are many categories and the classification is arbitrary. Automated classification is less prone to such mistakes and is more objective since there is no risk of human bias. For now, there are as many ways of classifying calls as laboratories which record them. Unified classification allows for a more accurate comparison of the collected data between different experiments in different laboratories as well as for the retrospective re-analysis of some old datasets.

In contrast to some other similar tools, MoUSE is designed as an open-source, free equipment, which allows other research groups to use it without extra costs. The installation process is simple and available for all major operating systems (Windows and Linux). Additionally, no programming experience is required to use the MoUSE tool. The interface is user-friendly and intuitive. Users can easily adjust the parameters of automated detection and classification to suit their needs.

In the future, we plan to apply MoUSE to other animal species such as dolphins. MoUSE also opens the possibility of being coupled with information about mouse/rat behaviour taken from another device (i.e., MS Kinect device or webcam), which we plan to add in the future.

5. Conclusions

We presented a software tool that is capable of efficient detection and classification of rodents' ultrasonic vocalizations. Our platform combines deep learning methods with ideas from classical pattern recognition and computer graphics. We investigated the accuracy and performance of several detection and classification algorithms. Applying the MoUSE platform to real experimental data we obtained a decent performance. Contrary to other approaches it is based solely on open-source frameworks and programming languages, that does not require expensive licenses.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

The work was funded by grant no. 2019/34/E/ST2/00457 of the National Science Centre, Poland and the Priority Research Area Digiworld under the programme Excellence Initiative – Research University at the Jagiellonian University in Cracow, Poland. Data used to develop this software was collected during a project funded by the Polish Ministry of Education and Science, Poland grant DIAMENTOWY GRANT 2019, no. 0049/DIA/2019/48 to Diana Piotrowska. This research was supported in part by PL-Grid Infrastructure, Poland. For compute-intensive tasks, like training neural networks, we used High-Performance Computing platforms such as PLGrid Infrastructure and Google Cloud Platform.

Appendix A. Configurations of algorithms

In Section 3.2 we present the results for the following configurations of algorithms for denoising and detection:

- FRCNN — Faster R-CNN model (NN with $\sim 10^7$ parameters) without any additional spectrogram preprocessing or postprocessing. For this method, we filter all the bounding boxes with prediction scores less than the threshold of 0.5. The value of this threshold can be adjusted, nevertheless, the default value works reasonably across a variety of examples.
- GAC 1
  - Sigma: 7.0,
  - Alpha: 50.0,
  - Flood level: 0.9,
  - Balloon: negative,
  - Balloon threshold: 0.9,
  - Smoothing: 3,
  - Iterations: 15.
- GAC 2 — configuration as in GAC 1, but with post-detection filtering of lower-frequency detections (threshold: 18 kHz).
- GAC 3
  - Sigma: 10.0,
  - Alpha: 500.0,
  - Flood level: 0.7,
  - Balloon: negative,
  - Balloon threshold: 0.7,
  - Smoothing: 3,
  - Iterations: 15.
- GAC 4 — configuration as in GAC 3, but with post-detection filtering of lower-frequency detections (threshold: 18 kHz).
- GAC 5 — configuration as in GAC 4 with noise-gate filtering. The noise source was a 300 ms piece of the same audio file, automatically selected so it does not contain any vocalizations. Noise-gate denoising has the following parameters:
  - Gradient pixels number (frequency): 3.0,
  - Gradient pixels number (time): 3.0,
  - Number std cutoff: 1.0,
  - Noise decrease: 0.3.

A more detailed explanation of all the GAC settings can be found in the MoUSE platform documentation (see Code metadata). For all GAC 1–5 detections we have also applied additional filtering, that removes artefacts that are frequently produced by the method, namely: overly large detections that are covering almost the whole spectrogram in time

Table B.4
Global recall and precision. For multiple IoU thresholds: [0.1 0.3 0.5 0.8 0.9]. All values are multiplied by 100 for better visibility.

	FRCNN	GAC 1	GAC 2	GAC 3	GAC 4	GAC 5
precision@10	81.28	77.70	78.06	86.04	87.19	83.73
recall@10	97.91	80.46	80.67	70.31	70.31	73.95
f1@10	88.46	78.42	78.76	75.88	75.99	78.24
precision@30	68.68	61.21	62.00	77.56	76.64	69.89
recall@30	84.81	63.89	64.93	63.74	63.19	61.52
f1@30	75.70	62.15	63.03	68.52	67.75	65.20
precision@50	60.19	42.98	42.65	52.74	53.44	47.22
recall@50	76.96	44.20	43.87	44.36	44.36	41.69
f1@50	67.44	43.40	43.08	47.26	47.32	44.12
precision@80	23.61	6.02	6.42	4.64	4.64	3.19
recall@80	33.44	6.02	6.42	4.39	4.39	2.96
f1@80	27.66	6.01	6.41	4.50	4.50	3.06
precision@90	5.78	0.74	0.84	0.00	0.00	0.18
recall@90	8.53	0.73	0.83	0.00	0.00	0.17
f1@90	6.88	0.73	0.83	0.00	0.00	0.17

Table B.5
Recall per vocalization category. For multiple IoU thresholds: [0.1 0.3 0.5 0.8 0.9]. All values are multiplied by 100 for better visibility.

	FRCNN	GAC 1	GAC 2	GAC 3	GAC 4	GAC 5
sh@10	99.51	69.02	69.02	61.42	61.42	45.32
oc@10	77.46	74.53	74.53	67.06	67.06	66.50
mc@10	98.34	86.35	87.08	83.12	83.12	79.15
tr@10	83.33	79.21	79.21	74.31	74.31	78.74
trc@10	66.67	56.33	56.33	57.78	57.78	57.78
sh@30	85.87	61.97	64.06	43.38	35.05	30.64
oc@30	77.46	73.02	73.02	67.06	67.06	66.50
mc@30	98.03	68.58	68.58	77.40	77.40	73.64
tr@30	82.29	67.82	67.82	70.92	70.92	73.77
trc@30	63.89	40.44	40.44	48.78	48.78	43.22
sh@50	71.29	28.62	27.58	0.00	0.00	0.49
oc@50	72.50	61.87	61.55	41.94	41.94	47.15
mc@50	92.92	50.54	50.85	59.21	59.21	49.98
tr@50	78.09	50.63	50.63	53.33	53.33	57.51
trc@50	63.89	28.67	28.67	39.11	39.11	33.56
sh@80	7.34	0.00	0.00	0.00	0.00	0.00
oc@80	25.60	3.99	2.48	0.00	0.00	0.00
mc@80	39.73	8.35	9.81	3.91	3.91	2.56
tr@80	34.09	5.76	5.76	5.41	5.41	6.05
trc@80	42.78	4.67	4.67	9.00	9.00	0.67
sh@90	2.45	0.00	0.00	0.00	0.00	0.00
oc@90	3.35	0.00	0.00	0.00	0.00	0.00
mc@90	13.72	0.54	0.84	0.00	0.00	0.33
tr@90	10.14	0.41	0.41	0.00	0.00	0.00
trc@90	0.00	2.00	2.00	0.00	0.00	0.00
22kHz@10	31.94	15.28	15.28	0.00	0.00	23.61
22kHz@30	18.75	9.72	10.42	0.00	0.00	8.33
22kHz@50	16.67	8.33	8.33	0.00	0.00	8.33
22kHz@80	0.00	0.00	0.00	0.00	0.00	0.00
22kHz@90	0.00	0.00	0.00	0.00	0.00	0.00
fl@10	33.33	25.69	25.69	27.78	27.78	23.61
fl@30	33.33	25.69	25.69	18.06	18.06	21.53
fl@50	27.78	13.19	7.64	2.08	2.08	2.08
fl@80	0.00	0.00	0.00	0.00	0.00	0.00
fl@90	0.00	0.00	0.00	0.00	0.00	0.00

and/or frequency axis. We eliminate them with the simple assumption that detections cannot span more than $\sim 40\%$ of the frequency axis (100 pixels) and cannot last longer than ~ 5 seconds in the time axis (5000 pixels).

Appendix B. Extended results

In this section, we demonstrate model performance for different values of threshold values. We can observe a drastic decrease in performance for classes, where annotation imprecision constitutes big differences in annotation surface area. 22 kHz is the hardest category to detect correctly, as it usually spans across multiple sliding windows, which makes partial coverage more likely, due to the process of merging boxes based on probability threshold (e.g. one missing sufficiently large bounding box in the middle window will split detection into two separate, decreasing IoU, while we consider this detection to be correct, tho undesirable) (see Tables B.4 and B.5).

Appendix C. Hypermarameters

C.1. Spectrogram

We are using unnormalized spectrograms as the basic data representation for the detection and classification algorithms. The spectrograms

Table C.6
FRCNN training parameters.

Parameter	Value
Optimizer	AdamW
Learning rate	5e−3
Weight decay	5e−4
Learning Rate Scheduling	linear with warmup
Warmup steps	300
Max Epochs	200 (best checkpoint arrived at 15)
Example duration	0.25[s]
Overlap	0.125[s]

Table C.7
FRCNN model hyperparameters.

Parameter	Value
Backbone	resnet-50/2
Anchor generator aspect ratios	[0.0078125, 0.015625, 0.25, 0.5, 1.0, 2.0, 6.0]
Anchor generator sizes	[32, 64, 128, 256]
RoI output size	7
RoI sampling ratio	2
FRCNN box head representation size	256
Normalization mean	−0.8314
Normalization std	0.2978

are generated using the FFT algorithm from time windows of 512 samples with hop length 256 and n_fft parameter set to 512. The frequency domain reaches about 120 kHz with ~ 400 Hz/pixel resolution and the time domain resolution is $\sim 1e-3$ s/pixel. Spectrograms are generated using torchaudio package.

C.2. FRCNN training

Training and architectural parameters are listed in Tables C.6 and C.7. We use ResNet-50 variant with only 2 stacks of bottleneck layers, we use this bottleneck, because it captures a large number of different features with a large number of output channels (512), while special reduction is low enough to produce accurate detection of small bounding boxes, that would otherwise be lost in a smaller feature map. We call this modification 'resnet-50/2'. During training we use a sliding window of examples, meaning that we select a chunk of recording of length 'Example duration' and move the sliding window in accordance with the 'Overlap' parameter. We drop all chunks that only contain noise, to moderate the number of easy negative examples in train and validation datasets. Then each example is passed through an augmentation pipeline, that clips the spectrogram to the 18 kHz+ range, applies log transform to manage very small and very large values in the spectrogram and finally applies sliding transform with 20% probability to each bounding box. We do not use random rescaling, because we found that it does not improve the training performance of our model. Unmentioned parameters are set to torchvision defaults.

References

[1] Panksepp J, Burgdorf J. 50-kHz chirping (laughter?) in response to conditioned and unconditioned tickle-induced reward in rats: effects of social housing and genetic variables. Behav. Brain Res. 2000;115(1):25–38. http://dx.doi.org/10.1016/s0166-4328(00)00238-2, [PubMed: 10996405].

[2] Wohr M. Measuring mania-like elevated mood through amphetamine-induced 50-kHz ultrasonic vocalizations in rats. Br. J. Pharmacol. 2022;179(17):4201–19. http://dx.doi.org/10.1111/bph.15487, [PubMed: 33830495].

[3] Neunuebel JP, Taylor AL, Arthur BJ, Egnor SR. Female mice ultrasonically interact with males during courtship displays. In: Mason P, editor. eLife 2015;4:e06203. http://dx.doi.org/10.7554/eLife.06203.

[4] Lahvis GP, Alleva E, Scattoni ML. Translating mouse vocalizations: prosody and frequency modulation. Genes Brain Behav. 2011;10(1):4–16. http://dx.doi.org/10.1111/j.1601-183X.2010.00603.x, arXiv:20497235.

[5] Lepri JJ, Theodorides M, Wysocki CJ. Ultrasonic vocalizations by adult prairie voles, Microtus ochrogaster. Experientia 1988;44:271–3.

[6] Kapusta J, Marchlewska-Koj A, Sales GD. Home bedding modifies ultrasonic vocalization of infant bank voles. J Chem Ecol 1995;21:577–82.

[7] Brudzynski SM. Ethotransmission: communication of emotional states through ultrasonic vocalization in rats. Curr. Opin. Neurobiol. 2013;23(3):310–7. http://dx.doi.org/10.1016/j.conb.2013.01.014, [PubMed: 23375168].

[8] Hamed A, Taracha E, Szyndler J, Krzaścik P, Lehner M, Maciejak P, et al. The effects of morphine and morphine conditioned context on 50 kHz ultrasonic vocalisation in rats. Behav. Brain Res. 2012;229(2):447–50.

[9] Zala SM, Reitschmidt D, Noll A, Balazs P, Penn DJ. Automatic mouse ultrasound detector (A-MUD): A new tool for processing rodent vocalizations. PLoS One 2017;12(7):e0181200.

[10] Tachibana RO, Kanno K, Okabe S, Kobayasi KI, Okanoya K. USVSEG: A robust method for segmentation of ultrasonic vocalizations in rodents. PLoS One 2020;15(2):e0228907.

[11] Premoli M, Baggi D, Bianchetti M, Gnutti A, Bondaschi M, Mastinu A, et al. Automatic classification of mice vocalizations using machine learning techniques and convolutional neural networks. PLoS One 2021;16(1):e0244636.

[12] Goussha Y, Bar K, Netser S, Cohen L, Hel-Or Y, Wagner S. HybridMouse: a hybrid convolutional-recurrent neural network-based model for identification of mouse ultrasonic vocalizations. Front Behav Neurosci 2022;15:810590.

[13] Netser S, Nahardiya G, Weiss-Dicker G, Dadush R, Goussha Y, John SR, et al. TrackUSF, a novel tool for automated ultrasonic vocalization analysis, reveals modified calls in a rat model of autism. BMC Biol. 2022;20(1):159.

[14] Baggi D, Premoli M, Gnutti A, Bonini SA, Leonardi R, Memo M, et al. Extended performance analysis of deep-learning algorithms for mice vocalization segmentation. Sci Rep 2023;13(1):11238.

[15] Van Segbroeck M, Knoll AT, Levitt P, Narayanan S. MUPET—mouse ultrasonic profile extraction: a signal processing tool for rapid and unsupervised analysis of ultrasonic vocalizations. Neuron 2017;94(3):465–85.

[16] Coffey KR, Marx RG, Neumaier JF. DeepSqueak: a deep learning-based system for detection and analysis of ultrasonic vocalizations. Neuropsychopharmacology 2019;44(5):859–68. http://dx.doi.org/10.1038/s41386-018-0303-6.

[17] Binder M, Nolan SO, Lugo JN. A comparison of the Avisoft (v.5.2) and MATLAB mouse song analyzer (v.1.3) vocalization analysis systems in C57BL/6, Fmr1-FVB.129, NS-Pten-FVB, and 129 mice. J Neurosci Methods 2020;346:108913. http://dx.doi.org/10.1016/j.jneumeth.2020.108913.

[18] Binder MS, Pranske ZJ, Lugo JN. The deepsqueak analysis system is as accurate, yet more efficient, than the Avisoft system across C57BL/6, FVB.129, and FVB mice. Brain Disorders 2022;8:100055. http://dx.doi.org/10.1016/j.dscb.2022.100055.

[19] Avisoft Saslab Pro. URL http://www.avisoft.com/.

[20] Raven Pro. URL https://ravensoundsoftware.com/software/raven-pro/.

[21] Reno JM, Marker B, Cormack LK, Schallert T, Duvauchelle CL. Automating ultrasonic vocalization analyses: The WAAVES program. J Neurosci Methods 2013;219(1):155–61. http://dx.doi.org/10.1016/j.jneumeth.2013.06.006, URL https://www.sciencedirect.com/science/article/pii/S0165027013002239.

[22] Barker DJ, Herrera C, West MO. Automated detection of 50-khz ultrasonic vocalizations using template matching in XBAT. J Neurosci Methods 2014;236:68–75. http://dx.doi.org/10.1016/j.jneumeth.2014.08.007.

[23] Reenskaug T. MODELS - VIEWS - CONTROLLERS. Technical note Xerox Palo Alto Research Center; 1979.

[24] Mallawaarachchi A, Ong S, Chitre M, Taylor E. Spectrogram denoising and automated extraction of the fundamental frequency variation of dolphin whistles. J Acoust Soc Am 2008;124:1159–70. http://dx.doi.org/10.1121/1.2945711.

[25] Audacity Wiki - How Audacity Noise Reduction Works. URL https://wiki.audacityteam.org/wiki/How_Audacity_Noise_Reduction_Works#algorithm.

[26] Márquez-Neila P, Baumela L, Alvarez L. A morphological approach to curvature-based evolution of curves and surfaces. IEEE Trans Pattern Anal Mach Intell 2014;36(1):2–17. http://dx.doi.org/10.1109/TPAMI.2013.106.

[27] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. Adv Neural Inf Process Syst 2015;28.

[28] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition. 2016, p. 770–8. http://dx.doi.org/10.1109/CVPR.2016.90.

[29] Wright JM, Gourdon JC, Clarke PB. Identification of multiple call categories within the rich repertoire of adult rat 50-kHz ultrasonic vocalizations: effects of amphetamine and social context. Psychopharmacology (Berl) 2010;211(1):1–13.

[30] Simola N, Granon S. Ultrasonic vocalizations as a tool in studying emotional states in rodent models of social behavior and brain disease. Neuropharmacology 2019;159:107420. http://dx.doi.org/10.1016/j.neuropharm.2018.11.008, [PubMed: 30445100].

[31] Wohr M, Schwarting RK. Affective communication in rodents: ultrasonic vocalizations as a tool for research on emotion and motivation. Cell Tissue Res. 2013;354(1):81–97. http://dx.doi.org/10.1007/s00441-013-1607-9, [PubMed: 23576070].

[32] Brunelli SA. Selective breeding for an infant phenotype: rat pup ultrasonic vocalization (USV). Behav. Genet. 2005;35(1):53–65. http://dx.doi.org/10.1007/s10519-004-0855-6, [PubMed: 15674532].

[33] Schwarting RKW. Ultrasonic vocalization in juvenile and adult male rats: A comparison among stocks. Physiol. Behav. 2018;191:1–11. http://dx.doi.org/10.1016/j.physbeh.2018.03.023, [PubMed: 29574044].