Vroom!: A Search Engine for Sounds by Vocal Imitation Queries

Yichi Zhang
University of Rochester
Rochester, NY
yichi.zhang@rochester.edu

Junbo Hu
University of Rochester
Rochester, NY
junbohu@rochester.edu

Yiting Zhang
Georgia Institute of Technology
Atlanta, GA
yiting.zhang@gatech.edu

Bryan Pardo
Northwestern University
Evanston, IL
pardo@northwestern.edu

Zhiyao Duan
University of Rochester
Rochester, NY
zhiyao.duan@rochester.edu

ABSTRACT

Traditional search through collections of audio recordings compares a text-based query to text metadata associated with each audio file and does not address the actual content of the audio. Text descriptions do not describe all aspects of the audio content in detail. Query by vocal imitation (QBV) is a kind of query by example that lets users imitate the content of the audio they seek, providing an alternative search method to traditional text search. Prior work proposed several neural networks, such as TL-IMINET, for QBV, however, previous systems have not been deployed in an actual search engine nor evaluated by real users. We have developed a state-of-the-art QBV system (*Vroom!*) and a baseline query-by-text search engine (*TextSearch*). We deployed both systems in an experimental framework to perform user experiments with Amazon Mechanical Turk (AMT) workers. Results showed that *Vroom!* received significantly higher search satisfaction ratings than *TextSearch* did for sound categories that were difficult for subjects to describe by text. Results also showed a better overall ease-of-use rating for *Vroom!* than *TextSearch* on the sound library used in our experiments. These findings suggest that QBV, as a complimentary search approach to existing text-based search, can improve both search results and user experience.

CCS CONCEPTS

• **Information systems** → **Search interfaces**; *Speech / audio search*; Similarity measures; Novelty in information retrieval.

KEYWORDS

vocal imitation, sound search, subjective evaluation, Siamese style convolutional recurrent neural networks, text description

ACM Reference Format:
Yichi Zhang, Junbo Hu, Yiting Zhang, Bryan Pardo, and Zhiyao Duan. 2020. Vroom!: A Search Engine for Sounds by Vocal Imitation Queries. In *2020 Conference on Human Information Interaction and Retrieval (CHIIR'20), March 14–18, 2020, Vancouver, BC, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3343413.3377963

CHIIR'20, March 14–18, 2020, Vancouver, BC, Canada

ACM ISBN 978-1-4503-6892-6/20/03…$15.00
https://doi.org/10.1145/3343413.3377963

1 INTRODUCTION

Designing methods to access and manage multimedia documents such as audio recordings is an important information retrieval task. Traditional search engines for audio files use text labels as queries. However this is not always effective. First, it requires users to be familiar with the audio library taxonomy and text labels, which is unrealistic for many users with no or little audio engineering background. Second, text descriptions or metadata are abstract and do not describe the audio content in detail. Third, many sounds, such as those generated by computer synthesizers, lack commonly accepted semantic meanings and text descriptions.

Vocal imitation is commonly known as using voice to mimic sounds. It is widely used in our daily conversations, as it is an effective way to convey sound concepts that are difficult to describe by language. For example, when referring to the "Christmas tree" dog barking sound (i.e., a barking sound with overtones fading out rapidly showing a Christmas-tree-shaped spectrogram) [32], vocal imitation is more intuitive compared to text descriptions. Hence, designing computational systems that allow users to search sounds through vocal imitation [4] goes beyond the current text-based search and enables novel human-computer interactions. It has natural advantages over text-based search as it does not require users to be familiar with the labels of an audio taxonomy and it indexes the detailed audio content instead of abstract text descriptions that not all agree on. Regarding applications, sound search by vocal imitation can be useful in many fields including movie and music production, multimedia retrieval, and security and surveillance.

Recently, a deep learning based model called TL-IMINET [43] was proposed for sound search by vocal imitation. It addresses two main technical challenges: 1) feature learning: what feature representations are appropriate for the vocal imitation and reference sound, and 2) metric learning: how to design the similarity between a vocal imitation and each sound candidate. Experiments on the VocalSketch Data Set [5] have shown promising retrieval performance for this model, however, no user studies have been conducted to validate the model as part of a user-facing search engine and the sound-search-by-vocal-imitation approach in general at the system level. In this paper, we seek to answer the following questions: 1) Is vocal-imitation-based search an acceptable approach to sound search for ordinary users without an extensive audio engineering

background? 2) How does vocal-imitation-based search compare with the traditional text-based search for different kinds of sounds in terms of search effectiveness, efficiency and user satisfaction?

To answer the above questions, in this work, we conduct a user study to compare sound search by vocal imitation and by text description on Amazon Mechanical Turk. Specifically, we designed a web-based search engine called *Vroom!*. The frontend GUI allows a user to record a vocal imitation as a query to search sounds in a sound library, and the backend uses a pre-trained deep learning model as the search algorithm. We also designed a baseline system called *TextSearch* for comparison. It allows a user to search a collection of sounds by comparing a user's text query to the keywords for each sound. We further developed a software experimental framework to record user behaviors and ratings on a cloud database using MongoDB Atlas.

In this study, each system was used by 100 workers, each of which was asked to search for 10 sounds randomly selected from a large sound library that contains 3,602 sounds from eight sound categories with an average length of 4 seconds. Each search was done within the sound's category. Analyses of search results and user behaviors show that subjects gave significantly higher overall ease-of-use scores to *Vroom!* than *TextSearch* in this sound library. Results also show significant advantages of *Vroom!* over *TextSearch* on categories that were difficult to describe by text.

The rest of the paper is organized as the following. We first review related work in Section 2. We then introduce the sound search by vocal imitation system *Vroom!* in Section 3, and the baseline of sound search by text description system *TextSearch* in Section 4. In Section 5, we describe the *FreeSoundIdeas* dataset that we collected for user evaluation. In section 6, we discuss the experimental framework, subject recruitment, and analyze the results. Finally we conclude the paper in Section 7.

2 RELATED WORK

Using text descriptions to search the metadata (e.g. filenames, text tags) in collections of audio files is already widely deployed. For example, Freesound [13] is an online community generated sound database with more than 420,000 sounds. Each audio file in the collection is tagged with text descriptions for text-based search. SoundCloud [34] is another community-based online audio distribution platform that enables users to search sounds by text description.

On the other hand, sound query by vocal imitation (QBV) is drawing increasing attention from the research community to address limitations of text-based search. It is one kind of of Query by Example (QBE) [44]. There are numerous QBE applications in the audio domain, such as content based sound search and retrieval [12, 37], audio fingerprinting of the exact match [36] or live versions [28, 35], cover song detection [3] and spoken document retrieval [6]. Vocal imitation of a sound was first proposed for music retrieval, such as finding songs by humming the melody as a query [9, 15] or beat boxing the rhythm [16, 18]. Recently, it has been extended for general sound retrieval, as summarized below.

Roma and Serra [29] designed a system that allows users to search sounds on Freesound by recording audio with a microphone, but no formal evaluation was reported. Blancas et al. [4] built a supervised system using hand-crafted features by the Timbre Toolbox [26] and an SVM classifier. Helén and Virtanen [17] designed a query by example system for generic audio. Hand-crafted frame-level features were extracted from both query and sound samples and the query-sample pairwise similarity was measured by probability distribution of the features.

In our previous work, we first proposed a supervised system using a Stacked Auto-Encoder (SAE) for automatic feature learning followed by an SVM for imitation classification [38]. We then proposed an unsupervised system called IMISOUND [39, 40] that uses SAE to extract features for both imitation queries and sound candidates and calculates their similarity using various measures [10, 22, 30]. IMISOUND learns feature representations independently of the distance metric used to compare sounds in the representation space. Later, we proposed an end-to-end Siamese style convolutional neural networks named IMINET [41] to integrate learning the distance metric and the features. This model was improved by transfer learning from other relevant audio tasks, leading to the state-of-the-art model TL-IMINET [43]. The benefits of applying positive and negative imitations to update the cosine similarity between the query and sound candidate embedding was investigated in [21]. To understand what such a neural network actually learns, visualization and sonification of the input patterns in Siamese style convolutional neural networks using activation maximization [11] was discussed in [42, 43].

To date, research on sound search by vocal imitation has been only conducted at the algorithm development level. No usable search engines have been deployed based on these algorithms, nor have any user studies been conducted to assess the effectiveness of the new search approach in practice. This paper conducts a large-scale user study along this line: evaluating the performance of a vocal-imitation-based search engine built on a best-performing deep learning algorithm, and comparing it with a traditional text-based search engine.

3 PROPOSED SEARCH ENGINE: *VROOM!*

We designed a web-based sound search engine by vocal imitation, called *Vroom!*, which can be accessed via https://vocalimitation.com. It includes frontend design and backend implementation.

3.1 Frontend GUI Design

The frontend GUI is designed using Javascript, HTML, and CSS. It allows a user to record a vocal imitation of sound that he/she is looking for from one of the categories described in Subsection 5.2 using the recorder.js Javascript library [25]. It also allows the user to listen to the recording, inspect the waveform, and re-record imitations. By clicking on the "Go Search!" button, the user can initiate the search request. The recording is then uploaded to the backend server and compared with each sound within the specified category using the CR-IMINET algorithm described later. Top five sound candidates with the highest similarity scores are first returned to the user, and more candidates up to 20 can be returned by clicking on "Show more results". The user can play the returned sounds and make a selection to complete the search. Only sounds that have been played become available for the selection. If not satisfied with

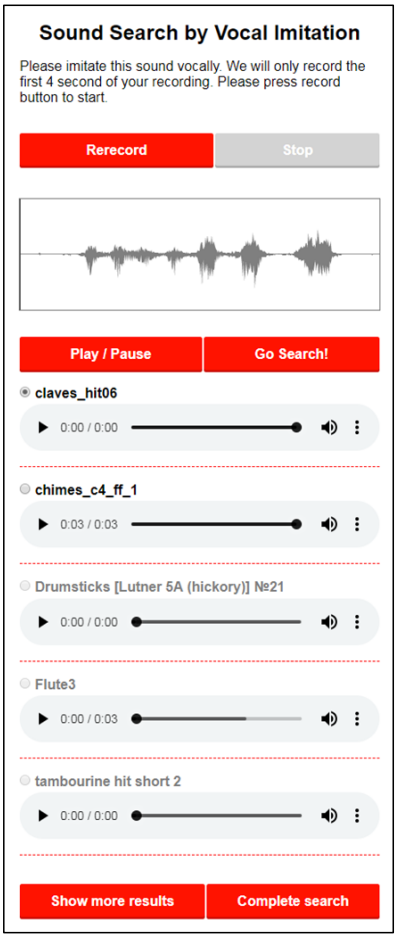


Figure 1: Frontend GUI of the vocal-imitation-based search engine *Vroom!*.

any of the returned sounds, the user can re-record an imitation and re-do the search. The frontend GUI is shown in Figure 1.

3.2 Backend Search Algorithm: CR-IMINET

Hosted on a Ubuntu system, the backend server is designed using Node.js express framework, with Keras v2.2.4 [7] and GPU acceleration supported. It receives the user's vocal imitation from the frontend, pre-processes the audio, then implements a Siamese style convolutional recurrent neural network model called CR-IMINET to calculate the similarity between the vocal imitation and candidate sounds in the sound library. It responds to each frontend search request and retrieves the most similar sounds to each imitation query, within the specified sound category of the sound library.

3.2.1 Architecture. As shown in Figure 2, CR-IMINET contains two identical Convolutional Recurrent Deep Neural Network (CRDNN) towers for feature extraction: One tower receives a vocal imitation (the query) as input. The other receives a sound from the library (the candidate) as input. Each tower outputs a feature embedding.

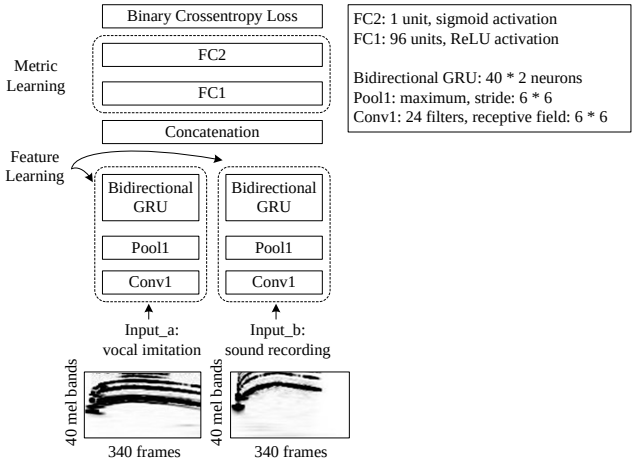


Figure 2: Architecture of the CR-IMINET. The two input spectrograms are of the same size. The two towers for feature extraction are of the same structure with shared weights. In each tower a convolutional layer is followed by a bi-directional GRU layer.

These embeddings are then concatenated and fed into a Fully Connected Network (FCN) for similarity calculation. The final single neuron output from FC2 is the similarity between the query and the candidate. The feature learning and metric learning modules are trained jointly on positive (i.e., related) and negative (i.e., non-related) query-candidate pairs. Through joint optimization, feature embeddings learned by the CRDNNs are better tuned for the FCN's metric learning, compared with isolated feature and metric learning in [40].

The Siamese (two-tower) and integrated feature and metric learning architecture in the proposed CR-IMINET is inspired by the previous state-of-the-art architecture, TL-IMINET [43]. Differently, TL-IMINET only uses convolutional layers in the feature extraction towers, while CR-IMINET uses both a convolutional layer and a bi-directional GRU (Gated Recurrent Unit) [8] layer. This configuration can better model temporal dependencies in the input log-mel spectrograms. Another difference is that TL-IMINET pre-trains the imitation and recording towers on environmental sound classification [31] and spoken language recognition [24] tasks, respectively, while CR-IMINET does not adopt this pre-training for simplicity thanks to its much smaller model size (shown in Table 1).

3.2.2 Training. We use the VimSketch dataset [19] to train the proposed CR-IMINET model. It is a combination of VocalSketch Data Set v1.0.4 [5] and Vocal Imitation Set [20]. The VocalSketch Data Set v1.0.4 contains 240 sounds with distinct concepts and 10 vocal imitations for each sound collected from different Amazon Mechanical Turkers. These sounds are from four categories: Acoustic Instruments (AI), Everyday (ED), Single Synthesizer (SS) and Commercial Synthesizers (CS). The number of sounds in these categories is 40, 120, 40 and 40, respectively. The Vocal Imitation Set is curated based on Google's AudioSet ontology [14], containing six categories of sounds in the first layer of the AudioSet ontology tree: Animal, Channel, Human Sounds, Music, Natural Sounds, Sounds

Table 1: Model size (# trainable parameters) and retrieval performance (MRR) comparisons between the proposed CR-IMINET and the previous state of the art, TL-IMINET.

Config.	Model Size	MRR (mean ± std)
TL-IMINET [43]	799.9k	0.325 ± 0.03
CR-IMINET	55.1k	0.348 ± 0.03

of Things, and Source Ambiguous Sounds. The number of sounds in these categories is 31, 4, 38, 65, 10, 134, and 20, respectively. Considering the relatively small number of sound recordings in the Channel and Natural Sounds categories (i.e., 4 and 10, respectively) and non-obvious association between the recording and corresponding imitations after listening to them, we remove these two categories in training.

We used in total 528 sounds from the remaining categories in VimSketch dataset for training. These sounds are of distinct concepts, and each has 10 to 20 vocal imitations from different people. All sounds and imitations are trimmed to about 4-second long each. We used 10-fold cross validation to train and validate the CR-IMINET model. In each fold, the number of sound concepts for training and validation is 476 and 52, respectively. Sounds from each concept are paired with imitations from the same concept as positive pairs, and paired with imitations from other concepts as negative pairs.

The ground-truth similarity labels are 1 for positive pairs and 0 for negative pairs. The loss function to minimize is the binary cross-entropy between the network output and the binary labels. Adam is used as the optimizer. The batch size is 128. Model training terminates after 20 epochs as the validation loss begins to increase afterwards.

3.2.3 Performance. We use Mean Reciprocal Rank (MRR) [27] to evaluate the retrieval performance.

$$MRR = \frac{1}{Q}\sum_{i=1}^{Q}\frac{1}{rank_i}, \tag{1}$$

where $rank_i$ is the rank of the target sound among all sounds in the library available for retrieval for the i-th imitation query; Q is the number of imitation queries. MRR ranges from 0 to 1 with a higher value indicating a better sound retrieval performance. We report the average MRR across 10 folds with 52 sound recordings in each fold to search from. The results are shown in Table 1.

It can be seen that CR-IMINET outperforms TL-IMINET in terms of MRR. An unpaired t-test shows that this improvement is statistically significant, at the significance level of 0.05 (p = 4.45e-2). An MRR of 0.348 suggests that within the 52 sound candidates in each fold, on average, the target sound is ranked as the top 3 candidate in the returned list. This suggests that CR-IMINET becomes the new state-of-the-art algorithm for sound search by vocal imitation.

4 BASELINE SEARCH ENGINE: *TEXTSEARCH*

To evaluate the performance of the proposed *Vroom!* search engine, we also designed a web-based sound search engine by text description as the baseline system, called *TextSearch*. It also includes frontend design and backend implementation described as the following.

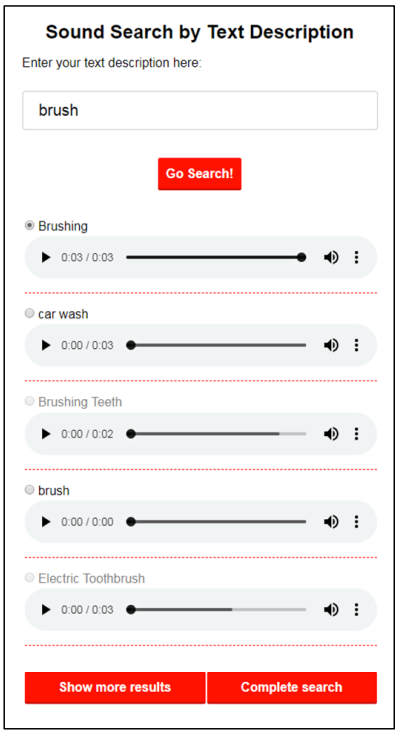


Figure 3: Frontend GUI of the text-description-based search engine *TextSearch*.

4.1 Frontend GUI Design

Similar to *Vroom!*, the frontend GUI of *TextSearch* is designed using Javascript, HTML, and CSS languages as well. As shown in Figure 3, the GUI provides the user with a text box to enter one or multiple keywords as the query for a sound. The user can then click on the "Go Search!" button to search.

The query string is uploaded to the backend server, and then compared with the keyword list associated to each sound candidate to find matches in the Solr database described in Subsection 4.2. Returned sound candidates are ranked in an order based on the internal matching mechanism of Solr. In order to have a comparable experimental setup with *Vroom!*, only the file names but not the keyword lists associated with the returned sounds are presented to the user. By clicking on the "Show more results" button, up to 20 sound candidates can be returned. The user can play the returned sounds and make a selection to complete the search, and only sounds that have been played can be selected. If not satisfied with any of the returned sounds, the user can re-type the query keywords and re-do the search.

4.2 Backend Search Algorithm

The backend is realized by designing a main server that receives and responds to requests from the frontend, and utilizing a separate

search engine service called Solr [1] specialized in text search and matching. The entire backend is hosted on a Ubuntu system. The overall process is as the following. The query request from the frontend users is first received by the main server. Then the main server resends query requests to Solr for text search and ranking. The ranked list is then returned to the main server from Solr, and finally returned to the frontend user.

Specifically, Solr is an open-source enterprise search platform built upon Apache Lucene. Each sound in *FreeSoundIdeas* has the following information, namely, sound filename, descriptive tags provided by the original Freesound.org uploader (i.e., keywords), and the unique sound ID. We organize this information of all sounds within each category into a separate JSON format file. Solr reads in the JSON file for each sound category and organizes the sound keywords into a tree structure for fast indexing given a query input. The retrieval similarity is calculated by Lucene's Practical Scoring Function [2]. Query parsing and partial word matching functions are also supported by Solr for a more user-friendly search experience.

4.3 Baseline Validity

Our designed baseline search engine *TextSearch* is comparable to other text-based search engines such as Freesound.org in the following two aspects. First, the workflow of *TextSearch* is the same as Freesound.org. Both search engines provide the user with a text box to type in the query strings, and return the user with a list of sounds ranked by relevance that are available for playback over multiple pages. Second, the backend Solr-based search algorithm used in *TextSearch* is also used in Freesound.org. It guarantees the search effectiveness and efficiency of the baseline.

TextSearch still has its limitations, for example, the searchable space is within each category of sounds in *FreeSoundIdeas*. This may prevent the user from finding sounds that are keyword-relevant to the query string but belong to other categories, while Freesound.org does not have this constraint. In addition, Freesound.org displays the associated keywords, the user quality rating, and the uploader information of the returned sounds, while *TextSearch* does not. Such metadata may aid the user in searching the target high quality sound more efficiently. Nevertheless, we believe that *TextSearch* implements the key features of a text-based search engine of sounds, and serves as a sufficient baseline for assessing the feasibility of vocal-imitation-based search.

5 SUBJECTIVE EVALUATION

5.1 Research Questions

By designing the proposed search engine *Vroom!* and the baseline *TextSearch*, we would like to understand and answer the following questions. 1) Is vocal-imitation-based search an acceptable approach to sound search for ordinary users without an extensive audio engineering background? 2) How does vocal-imitation-based search compare with the traditional text-based search for different kinds of sounds in terms of search effectiveness and efficiency? In this section, we present a large-scale user study on Amazon Mechanical Turk to answer these questions.

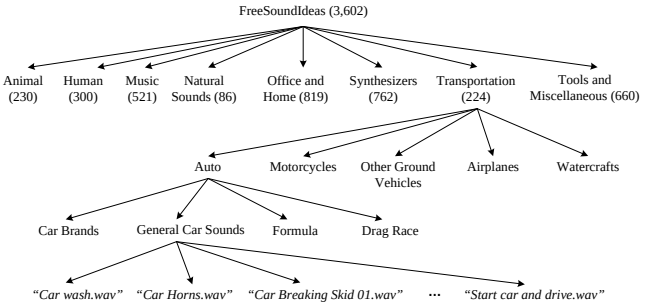


Figure 4: *FreeSoundIdeas* dataset ontology. Numbers in parentheses indicate the number of sound concepts in the category at the first level of the ontology tree.

5.2 Evaluation Dataset Collection

To carry out the subjective evaluation, we create a new dataset called *FreeSoundIdeas* as our sound library. The sounds of this dataset are from Freesound.org [13], while we reference sound descriptions and the structure of how sounds are organized in Sound Ideas [33] to form the *FreeSoundIdeas* ontology. Specifically, the ontology has a multi-level tree structure and is derived from two libraries of Sound Ideas: "General Series 6000 - Sound Effect Library" and "Series 8000 Science Fiction Sound Effects Library", where the former has more than 7,500 sound effects covering a large scope, and the latter has 534 sound effects created by Hollywood's best science fiction sound designers. We copied the indexing keywords from 837 relatively distinct sounds in these two libraries and formed eight categories of sound concepts, namely, Animal (ANI), Human (HUM), Music (MSC), Natural Sounds (NTR), Office and Home (OFF), Synthesizers (SYN), Tools and Miscellaneous (TOL), and Transportation (TRA).

We do not use sounds from Sound Ideas because of copyright issues, instead, we use keywords of each sound track from the abovementioned ontology as queries to search similar sound from Freesound.org. For each query, the first 5 to 30 returned sounds from Freesound.org are downloaded and stored as elements for our *FreeSoundIdeas* dataset. Keywords of these sounds from Freesound.org instead of the queried keywords to find these sounds are stored together with these sounds for a more accurate description. It is noted that this FreeSoundIdeas dataset has no overlap with the VimSketch dataset which is used to train the search algorithm for Vroom!.

In total the *FreeSoundIdeas* dataset includes 3,602 sounds. There are 230, 300, 521, 86, 819, 762, and 660 sound concepts in the category of ANI, HUM, MSC, NTR, OFF, SYN, TOL, and TRA, respectively. Its ontology is shown in Figure 4, with the Transportation category being expanded to leaf nodes to illustrate the granularity.

5.3 Experimental Framework

To quantify search behaviors and user experiences and to make quantitative comparisons between *Vroom!* and *TextSearch*, we designed an experimental framework that wraps around each search engine. The experimental framework is another web application.

The framework guides each subject to make 10 searches, rate their satisfaction score about each search, and rate the ease-of-use score for the search engine after completing all 10 searches. For each search, it guides the subject through three steps. In Step 1, the subject listens to a reference sound randomly chosen from a category of the sound library. The category name is visible while the keywords of the sound is not provided. This sound will be the target sound to search in following steps. In Step 2, the reference sound is hidden from the subject, and the subject uses the search engine (*Vroom!* or *TextSearch*) to search for the reference sound in the specified category of the sound library. In Step 3, the reference sound appears again. The user compares it with their retrieved sound to rate their satisfaction about the search. These three steps, for the *Vroom!* search engine, are shown in Figure 5 for illustration.

The experimental framework tries to mimic the search processes in practice as much as possible. For example, searches are conducted in each of the eight root categories instead of over the entire library to reduce complexity, as the root-level categories show clear distinctions on their semantics. However, certain modifications still have to be made to allow quantitative analysis. In practice, a user rarely listens to the exact target sound before a search; they usually only have a rough idea about the target sound in their mind to cast their query (imitation or text). In our experimental framework, however, before each search, the subject listens to the target sound to cast their query. While this may positively bias the quality of the query (especially for the imitation query), this is necessary to control what sound to search by the subjects. For example, the library may simply not contain the target sound if we allowed subjects to search freely. To reduce this positive bias, we hide the target sound during the search (Step 2).

The backend of this experimental framework records statistics of important search behaviors listed as the following. These statistics are then sent to MongoDB Atlas cloud database for storage and analysis.

- User satisfaction rating for each search
- Ease-of-use rating for the search engine
- Number of "Go Search!" button clicked
- Number of returned candidate sounds played
- Total time spent for each search
- Rank of the target sound in the returned list for each search

5.4 Subject Recruitment

We recruited a total of 200 workers (i.e., 100 for each search engine) from the crowdsourcing platform Amazon Mechanical Turk (AMT) as our subjects. Our sound search tasks were released through cloudresearch.com [23] as it provides several more advanced and convenient features compared with the task releasing mechanism in AMT. Our recruiting criteria are summarized as the following.

(1) AMT worker's historical HIT performance. We required that each worker had a HIT approval rate higher than 97% and the number of HITs higher than 50, and the worker was located in the United States.

(2) Duplicate submission prevention. We blocked multiple submissions from the same IP address, and verified that worker's IP was consistent with the United States region setting. Finally, we strictly controlled that there was no overlap between workers in the two groups. This was to prevent a worker from becoming familiar with the sound library, which might cause a positive bias to the second search engine that the user tested.

(3) Equipment requirement. We asked workers to use Chrome or Firefox to complete the task, as a comprehensive internal test was conducted on these two browsers. Those finished with other browsers (i.e., IE, Safari, etc.) were rejected to rule out any unexpected issues. Also, we asked workers to sit in a quiet environment and make sure their speaker and microphone were on.

The demographic information of the recruited subjects is summarized in Figure 6. We can see that the gender distribution is quite even, and a large portion of subjects were born in the 1980s and 1990s. For race distribution, most subjects are White/Caucasian, followed by Black/African American and Asian.

Before the worker starts, he/she is welcomed with the task portal including instructions, an external link directed to our web based experimental framework hosting *Vroom!* and *TextSearch*, and a text box for entering the completion code, which will be available to copy and paste on the last page of the experimental framework, after the user finishes his/her sound search task.

The two groups of subjects were asked to perform 10 sound searches using *Vroom!* and *TextSearch*, respectively. Subjects were informed about the collection of their search behaviors and ratings before the experiments. After the user finished 10 sound searches and provided ease-of-use score and general feedback, then the completion code would be available for the user to paste into the text box from the task portal. Finally, we verified the submitted completion code from each subject to approve his/her job.

Our internal pilot trials show that each experiment took about 25 and 15 minutes for *Vroom!* and *TextSearch*, respectively. Therefore, we paid each subject 1.5 US dollars for *Vroom!* and 1 US dollar for *TextSearch*. To encourage the subjects to treat the experiments more seriously, we made an extra 50% bonus payment based on the worker's performance. Subjects were informed about this compensation policy including the bonus payment before they started the experiments.

5.5 Experimental Results

5.5.1 User Feedback. Figure 7 compares two types of user ratings between *Vroom!* and *TextSearch*: 1) User's satisfaction rating (SAT) indicates how satisfied a user is with each search by comparing the finally retrieved sound to the reference sound (collected in Step 3 in Figure 5); 2) ease-of-use rating evaluates a user's overall experience of each search engine upon the completion of all 10 searches.

It can be seen that *Vroom!* shows a statistically significantly higher ease-of-use rating than *TextSearch* at the significance level of 0.05 (p=0.0324, unpaired t-test). This suggests a positive answer to the first research question raised in Section 5.1, i.e., vocal-imitation-based search can be accepted by ordinary users without an extensive audio engineering background. The average satisfaction rating of all categories shows slightly better performance of *Vroom!* than *TextSearch*. However, a further inspection reveals that the average satisfaction rating varies much from one category to another. For MSC, NTR, SYN, and TOL categories, *Vroom!* receives a statistically significantly higher satisfaction rating than *TextSearch* does, at the significance level of 0.1 (MSC p = 9.8e-2), 0.1 (NTR p = 6.1e-2), 0.001

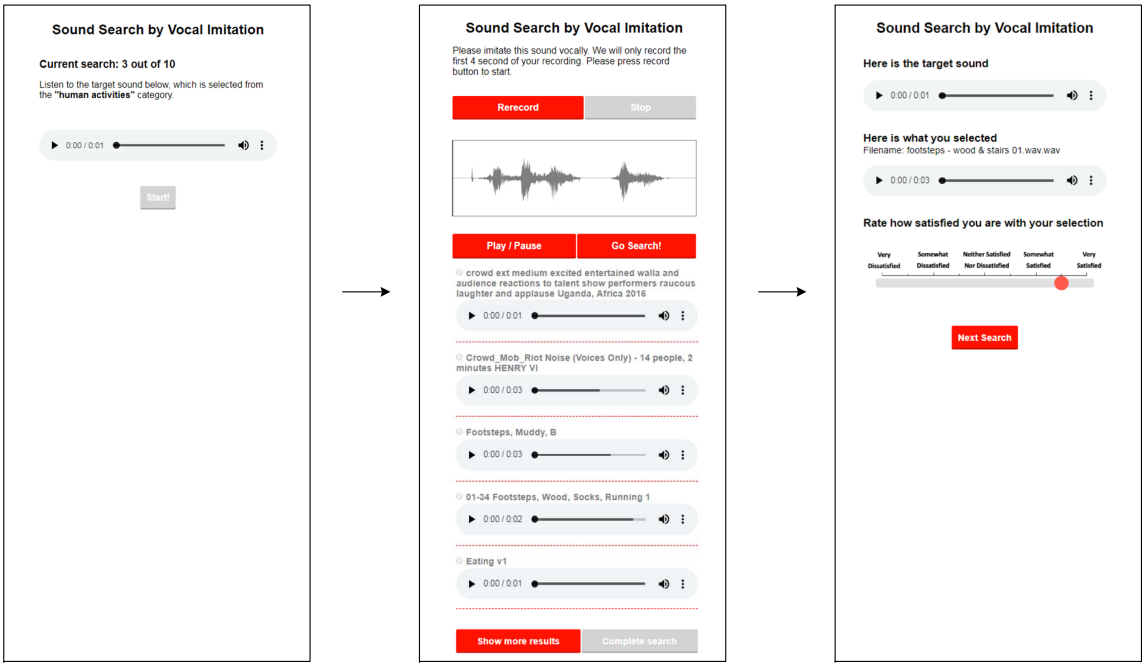
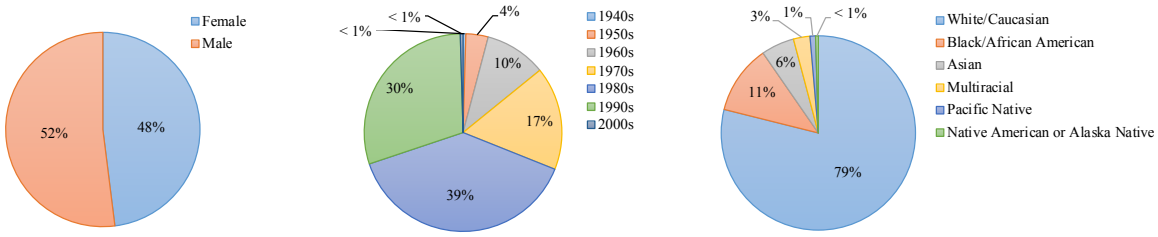


Figure 5: Experimental framework hosting the proposed vocal imitation based search engine *Vroom!*. The framework hosting the text description based search engine *TextSearch* is exactly the same except that Step 2 is replaced with the *TextSearch* engine.



(a) Gender (b) Birth Decade (c) Race

Figure 6: Pie charts showing the demographic information of the subjects. Charts are created based on the information collected from cloudresearch.com. Please note that cloudresearch.com may not have every demographic for every subject.

(SYN p = 8.22e-10), and 0.1 (TOL p = 8.79e-2), respectively, under unpaired t-tests. This is because many subjects could not recognize sounds from these categories nor find appropriate keywords to search in *TextSearch*. This is especially significant for the SYN category, as many sounds simply do not have semantically meaningful or commonly agreeable keywords, while imitating such sounds was not too difficult for many subjects. Also please note that in conducting *Vroom!* experiments with AMT workers, TOL Category was named as a much boarder concept called "Sound of Things", while in *TextSearch* this category was renamed to the current "Tools and Miscellaneous" to provide more information and help the worker better understand the sounds they heard. This may slightly bias the experimental results to *TextSearch* in TOL category. Nevertheless, in the figure we see that *Vroom!* still outperforms *TextSearch* in TOL category in terms of user satisfaction rating.

On the other hand, for the ANI, HUM, and OFF category, however, *TextSearch* outperforms *Vroom!* significantly in terms of satisfaction rating, at the significance level of 0.005 (ANI p = 1.2e-3), 0.005 (HUM p = 1.2e-3), 0.05 (OFF p = 2.1e-2), respectively, under unpaired t-tests. Subjects were more familiar with these sounds that can be easily identified in everyday environments and knew how to describe them with keywords, while some sounds could be difficult to imitate, e.g., shuffling cards, toilet flushing, and cutting credit card.

For the remaining TRA category, the average satisfaction rating of *TextSearch* is slightly better than our proposed *Vroom!*, however, such outperformance is not statistically significant.

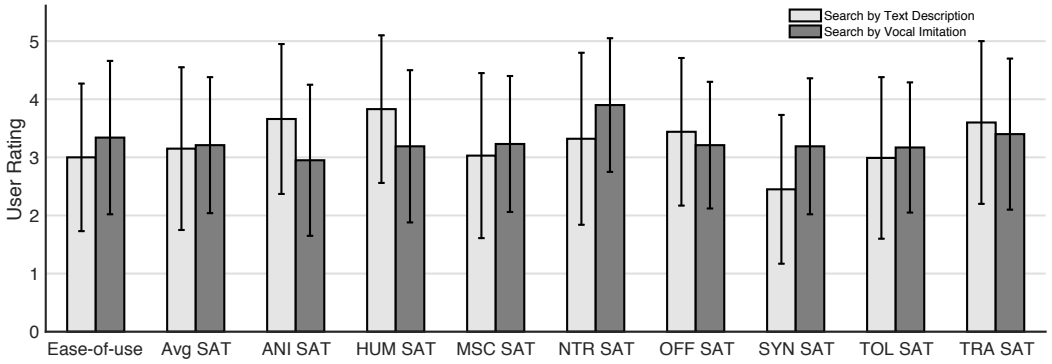


Figure 7: Average user ratings of sound search by text description (*TextSearch*) and vocal imitation (*Vroom!*). Ratings include the overall ease-of-use rating of the two search engines, and search satisfaction (SAT) within each sound category and across all categories. Error bars show standard deviations

5.5.2 User Behaviors. Figure 8 further compares user behaviors between *Vroom!* and *TextSearch*. First, for both search engines, it is obvious to observe the trend of positive correlation among the number of search trials, the number of sounds played, and the total time spend in one sound search.

Second, *Vroom!* has significantly fewer search trials than *TextSearch* in all categories except HUM as shown in Table 2. Note that in HUM category the mean number of search trials in *Vroom!* is still lower than *TextSearch*, although it is not statistically significant. Considering that user satisfaction ratings for *Vroom!* in MSC, NTR, SYN, and TOL categories are significantly higher than those for *TextSearch*, this suggests that fewer search trials may lead to better search experience. Higher search efficiency can be achieved by requesting less queries from the user. This answers the second research question in Subsection 5.1 in terms of search efficiency.

On the other hand, the average number of played sound candidates in each search using *Vroom!* is much larger than that of using *TextSearch*. As file names of returned sound candidates often contain semantic meanings, we believe that users can often skip listening to sound candidates when their file names seem irrelevant to the text query in *TextSearch*. For *Vroom!*, such "short cut" is not available and listening is often the only way to assess the relevancy.

Finally, the overall time spent on each search in *Vroom!* is significantly longer than that in *TextSearch*. This can be explained by the larger number of sounds played in *Vroom!* as well as the additional time spent to record and playback vocal imitations compared to typing in keywords.

5.5.3 Ranking of Target Sound. We visualize the target sound ranking distribution for both the proposed *Vroom!* and baseline *TextSearch* across different categories. Complimentary to User Feedback in Section 5.5.1, it is an objective evaluation measure to compare the two search engine performance.

Please note that in *Vroom!* the target sound is always in the returned candidate list. But in *TextSearch*, given the user's query keywords, the target sound may or may not be in the returned candidate list. If the target sound is not in the candidate list, we treat the target sound rank as 999, which is greater than the number of sounds in each category. As shown in Figure 9, black and white

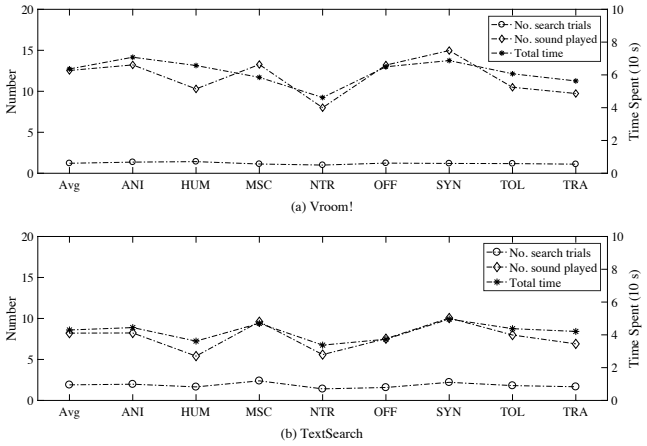


Figure 8: User behavior statistics for *Vroom!* and *TextSearch*. Each point in the plot is the mean value with std omitted for better visualization.

bars indicate rank counts for *Vroom!* and *TextSearch*, respectively. The overlapping portion between the two systems is shown in grey. Some interesting findings can be observed as the following.

First, target ranks in *Vroom!* are distributed more flattened within the range of maximal number of sounds in that category. But *TextSearch* shows a more polarized result that either the target sound ranks very high or has no rank at all. It indicates that the user may choose highly matched keywords of the target sound or cannot come up with relevant descriptions for that sound entirely. This is obvious in MSC, OFF, and SYN categories, by comparing the leftmost and rightmost white bars in the figure. For example in the SYN category, for users without music or audio engineering background, describing a synthesizer sound effect by text is very challenging (e.g., a sound effect named "eerie-metallic-noise.mp3" annotated with keywords of "alien", "eerie", "glass", and "metallic").

Second, in HUM, MSC, OFF, and TOL categories, *Vroom!* shows a smaller proportion of high ranks compared with *TextSearch*, while

Table 2: P-values of unpaired t-tests verifying if the hypotheses in the first column are statistically significant, at the significance level of 0.05.

User behavior hypothesis	Avg	ANI	HUM	MSC	NTR	OFF	SYN	TOL	TRA
No. search trials (*TextSearch* > *Vroom!*)	7.66e-24	2.78e-2	Not significant	3.79e-8	4.76e-2	1.60e-3	2.64e-8	6.74e-8	5.40e-3
No. sound played (*Vroom!* > *TextSearch*)	5.87e-27	6.86e-4	3.86e-5	7.14e-4	4.81e-2	9.86e-12	7.24e-7	6.79e-4	1.10e-2
Total time (*Vroom!* > *TextSearch*)	1.69e-33	6.03e-4	2.58e-6	3.80e-3	4.26e-2	2.55e-14	9.84e-7	2.22e-6	3.70e-3

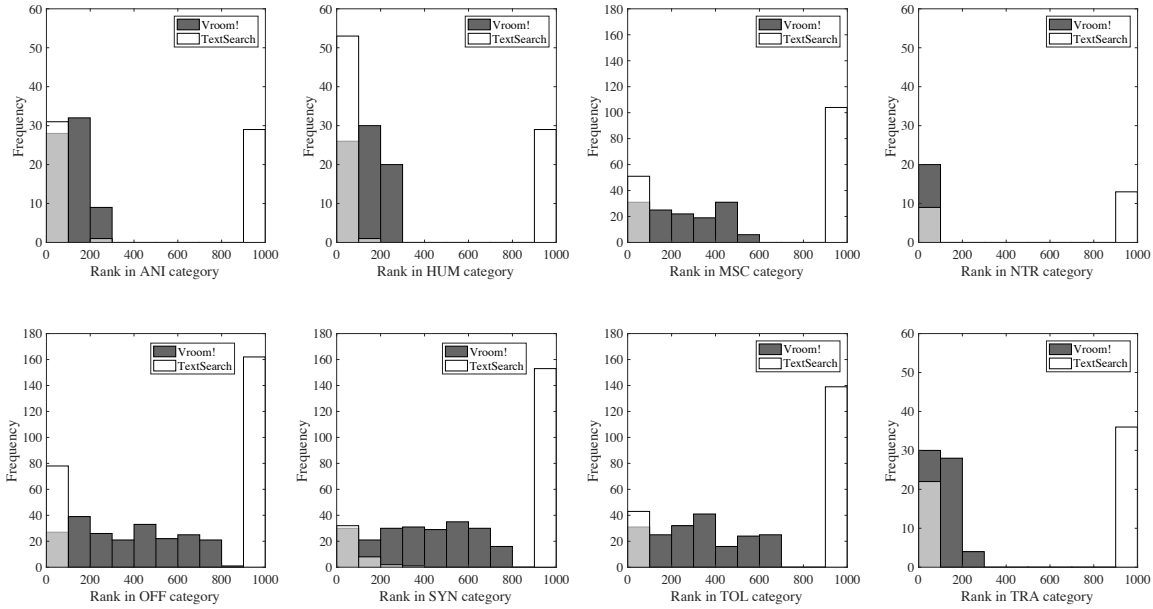


Figure 9: Comparisons of target sound rank in the returned sound list between *Vroom!* and *TextSearch* within each category.

in other categories like NTR, SYN, TRA, *Vroom!* shows a higher proportion of high ranks. In both cases the target sound from *Vroom!* can be low ranked around several hundreds out of the entire returning candidate sound list. We argue that this is still different from the out-of-scope situation if no matched keywords can be found in *TextSearch*. In practice, *Vroom!* could return more than 20 sounds for the user to choose from, and could work with text-based search to reduce the candidate pool, target sounds in low ranks could still be possible to discover. Furthermore, as the returned candidate sounds are ranked based on the content similarity with the vocal imitation query, even if the target sound is low ranked, other high ranked candidate sounds may still align well with the user's taste.

SAT rating and target sound ranking indicate the subjective feeling and objective evaluation about sound search effectiveness of *Vroom!* compared with *TextSearch*. It answers the second research question in terms of search effectiveness.

6 CONCLUSIONS

This paper presented a search engine for sounds by vocal imitation queries called *Vroom!*. It has a frontend GUI allowing the user to record his/her vocal imitation of a sound concept and search for the sound in a library. Its backend hosts a Siamese convolutional recurrent neural network model called CR-IMINET to calculate the similarity between the user's vocal imitation with sound candidates in the library. We conducted a comprehensive subjective study on Amazon Mechanical Turk with 200 workers to evaluate the performance of the vocal-imitation-based search engine and compare with a text-based sound search engine *TextSearch* as the baseline. We developed an experimental framework to wrap around *Vroom!* and *TextSearch* to conduct this user study. User ratings and behavioral data collected from the workers showed that vocal-imitation-based search has significant advantages over text-based search for certain categories (e.g., Synthesizers, Music, Natural Sounds, and Tools and Miscellaneous) of sounds in our collected FreeSoundIdeas sound effect library. Ease-of-use ratings of the vocal-imitation-based engine is also significantly higher than that of the text-based engine. Nonetheless, we can still benefit from text-based sound search engines in categories that we are familiar with (e.g., Animal, Human, and Office and Home). For future work, we would like to further improve the performance of the Vroom! search algorithm by incorporating the attention mechanism and to design search paradigms to combine vocal-imitation-based and text-based search together.

ACKNOWLEDGMENTS

This work is funded by the National Science Foundation grants No. 1617107 and No. 1617497. We also acknowledge NVIDIA's GPU donation for this research.

REFERENCES

[1] [n.d.]. Apache Solr. Retrieved September 30, 2019 from http://lucene.apache.org/solr/

[2] [n.d.]. Class Similarity. Retrieved September 30, 2019 from http://lucene.apache.org/core/3_5_0/api/core/org/apache/lucene/search/Similarity.html

[3] Thierry Bertin-Mahieux and Daniel PW Ellis. 2011. Large-scale cover song recognition using hashed chroma landmarks. In *Proc. Applications of Signal Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on*. 117–120. https://doi.org/10.1109/ASPAA.2011.6082307

[4] David S. Blancas and Jordi Janer. 2014. Sound retrieval from voice imitation queries in collaborative databases. In *Proc. Audio Engineering Society 53rd International Conference on Semantic Audio*. 1–6.

[5] Mark Cartwright and Bryan Pardo. 2015. VocalSketch: Vocally imitating audio concepts. In *Proc. the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI)*. 43–46. https://doi.org/10.1145/2702123.2702387

[6] Tee Kiah Chia, Khe Chai Sim, Haizhou Li, and Hwee Tou Ng. 2008. A lattice-based approach to query-by-example spoken document retrieval. In *Proc. the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 363–370. https://doi.org/10.1145/1390334.1390397

[7] François Chollet et al. 2015. Keras. https://github.com/fchollet/keras.

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, , and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[9] Roger B Dannenberg, William P Birmingham, Bryan Pardo, Ning Hu, Colin Meek, and George Tzanetakis. 2007. A comparative evaluation of search techniques for query-by-humming using the MUSART testbed. *Journal of the Association for Information Science and Technology* 8, 5 (2007), 687–701. https://doi.org/10.1002/asi.20532

[10] Najim Dehak, Reda Dehak, James R Glass, Douglas A Reynolds, and Patrick Kenny. 2010. Cosine similarity scoring without score normalization techniques. In *Odyssey*. 1–5.

[11] Dumitru Erhan, Aaron Courville, and Yoshua Bengio. 2010. Understanding representations learned in deep architectures. *Department d'Informatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep 1355* (2010), 1–25.

[12] Jonathan T. Foote. 1997. Content-based retrieval of music and audio. In *Proc. Multimedia Storage and Archiving Systems II, International Society for Optics and Photonics*, Vol. 3229. 138–147. https://doi.org/10.1117/12.290336

[13] Freesound.Org. 2005. Freesound. Retrieved September 30, 2019 from http://www.freesound.org/

[14] Jort F. Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *Proc. Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. 776–780. https://doi.org/10.1109/ICASSP.2017.7952261

[15] Asif Ghias, Jonathan Logan, David. Chamberlin, and Brian C. Smith. 1995. Query by humming: musical information retrieval in an audio database. In *Proc. the 3rd ACM International Conference on Multimedia*. 231–236. https://doi.org/10.1145/217279.215273

[16] Oliver Gillet and Gaël Richard. 2005. Drum loops retrieval from spoken queries. *Journal of Intelligent Information Systems* 24 (2005), 159–177. https://doi.org/10.1007/s10844-005-0321-9

[17] Marko Helén and Tuomas Virtanen. 2009. Audio query by example using similarity measures between probability density functions of features. *EURASIP Journal on Audio, Speech, and Music Processing* 2010, 1 (2009), 179303. https://doi.org/10.1155/2010/179303

[18] Ajay Kapur, Manj Benning, and George Tzanetakis. 2004. Query-by-beatboxing: Music retrieval for the DJ. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*. 170–177.

[19] Bongjun Kim, Mark Cartwright, and Bryan Pardo. 2019. VimSketch Dataset. https://doi.org/10.5281/zenodo.2596911

[20] Bongjun Kim, Madhav Ghei, Bryan Pardo, and Zhiyao Duan. 2018. Vocal Imitation Set: a dataset of vocally imitated sound events using the AudioSet ontology. In *Proc. Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE)*. 1–5.

[21] Bongjun Kim and Bryan Pardo. 2019. Improving Content-based Audio Retrieval by Vocal Imitation Feedback. In *Proc. Acoustics, Speech and Signal Processing (ICASSP), 2019 IEEE International Conference on*. 4100–4104. https://doi.org/10.1109/ICASSP.2019.8683461

[22] Solomon Kullback and Richard A. Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics* 22, 1 (1951), 79–86. https://doi.org/10.1214/aoms/1177729694

[23] Leib Litman, Jonathan Robinson, and Tzvi Abberbock. 2017. TurkPrime.com: A versatile crowdsourcing data acquisition platform for the behavioral sciences. *Behavior Research Methods* 49, 2 (2017), 433–442. https://doi.org/10.3758/s13428-016-0727-z

[24] Gregoire Montavon. 2009. Deep learning for spoken language identification. In *Proc. NIPS Workshop on deep learning for Speech Recognition and Related Applications*. 1–4.

[25] Octavian Naicu. 2018. Using Recorder.js to capture WAV audio in HTML5 and upload it to your server or download locally. Retrieved September 30, 2019 from https://blog.addpipe.com/using-recorder-js-to-capture-wav-audio-in-your-html5-web-site/

[26] Geoffroy Peeters, Bruno L. Giordano, Patrick Susini, Nicolas Misdariis, and Stephen McAdams. 2011. The timbre toolbox: Extracting audio descriptors from musical signal. *The Journal of the Acoustical Society of America* 130, 5 (2011), 2902–2916. https://doi.org/10.1121/1.3642604

[27] Dragomir R. Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating web-based question answering systems. *Ann Arbor* 1001 (2002), 48109.

[28] Zafar Rafii, Bob Coover, and Jinyu Han. 2014. An audio fingerprinting system for live version identification using image processing techniques. In *Proc. Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. 644–648. https://doi.org/10.1109/ICASSP.2014.6853675

[29] Gerard Roma and Xavier Serra. 2015. Querying freesound with a microphone. In *Proc. the 1st Web Audio Conference (WAC)*.

[30] Hiroaki Sakoe and Seibi Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transaction on* 26, 1 (1978), 43–49. https://doi.org/10.1109/TASSP.1978.1163055

[31] Justin Salamon and Juan Pablo Bello. 2017. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters* 24, 3 (2017), 279–283. https://doi.org/10.1109/LSP.2017.2657381

[32] John P. Scott. 1976. Genetic variation and the evolution of communication. In *Communicative Behavior and Evolution* (1st ed.), Martin E. Hahn and Edward C. Simmel (Eds.). ACM Press, New York, NY, 39–58.

[33] Sound-Ideas.Com. 1978. Sound Ideas. Retrieved September 30, 2019 from https://www.sound-ideas.com/

[34] SoundCloud.Com. 2008. SoundCloud. Retrieved September 30, 2019 from https://www.soundcloud.com/

[35] Timothy J. Tsai, Thomas Prätzlich, and Meinard Müller. 2016. Known artist live song ID: A hashprint approach. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*. 427–433.

[36] Avery Wang. 2003. An industrial strength audio search algorithm. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*. 7–13.

[37] Erling Wold, Thom Blum, Douglas Keislar, and James Wheaten. 1996. Content-based classification, search, and retrieval of audio. *IEEE Multimedia* 3, 3 (1996), 27–36. https://doi.org/10.1109/93.556537

[38] Yichi Zhang and Zhiyao Duan. 2015. Retrieving sounds by vocal imitation recognition. In *Proc. Machine Learning for Signal Processing (MLSP), 2015 IEEE International Workshop on*. 1–6. https://doi.org/10.1109/MLSP.2015.7324316

[39] Yichi Zhang and Zhiyao Duan. 2016. IMISOUND: An unsupervised system for sound query by vocal imitation. In *Proc. Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. 2269–2273. https://doi.org/10.1109/ICASSP.2016.7472081

[40] Yichi Zhang and Zhiyao Duan. 2016. Supervised and unsupervised sound retrieval by vocal imitation. *Journal of the Audio Engineering Society* 64, 7/8 (2016), 533–543. https://doi.org/10.17743/jaes.2016

[41] Yichi Zhang and Zhiyao Duan. 2017. IMINET: Convolutional semi-Siamese networks for sound search by vocal imitation. In *Proc. Applications of Signal Processing to Audio and Acoustics (WASPAA), 2017 IEEE Workshop on*. 304–308.

[42] Yichi Zhang and Zhiyao Duan. 2018. Visualization and interpretation of Siamese style convolutional neural networks for sound search by vocal imitation. In *Proc. Acoustics, Speech and Signal Processing (ICASSP), 2018 IEEE International Conference on*. 2406–2410. https://doi.org/10.1109/ICASSP.2018.8461729

[43] Yichi Zhang, Bryan Pardo, and Zhiyao Duan. 2019. Siamese style convolutional neural networks for sound search by vocal imitation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27, 2 (2019), 429–441. https://doi.org/10.1109/TASLP.2018.2868428

[44] Moshe M. Zloof. 1977. Query-by-example: A data base language. *IBM Systems Journal* 16, 4 (1977), 324–343.