INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS

ISSN(print): 2643-9840, ISSN(online): 2643-9875

Volume 05 Issue 08 August 2022

DOI: 10.47191/ijmra/v5-i8-50, Impact Factor: 6.261

Page No. 2287-2294

A Mobile Solution for Speech Content Memorizing

Said LAZRAK[1], Abdelillah SEMMA[2], Noureddine AHMER ELKAAB[3], Driss MENTAGUI[4]

[1,2,3,4]Ibn Tofail University, Kenitra, Morocco

ABSTRACT: The memorization is the process that allows human to acquire new knowledge and retain it in the log-term memory. Many techniques have been developed to optimize the human memorisation process. In this paper, we present a new method for memorising a speech audio content. Our method is based on segmenting and listening. The approach methods are implemented into a mobile application.

KEYWORDS: Humain Memorization, Digital Audio Processing, Supervised Learning, SVM.

1 INTRODUCTION

The purpose of this paper is to present our work relying on the two separate areas: memorization and Audio segmentation.

This paper is organized as fellows: In the first section, we illustrate different learning and memorisation methods, and present our proposed method. The second section is dedicated to the automatic segmentation of audio signals; in this section we present our audio segmentation SVM algorithm. In the third section, we describe our mobile application solution for memorizing.

2 MEMORIZATION

The aim of memorization process, and learning in general, is to acquire knowledge and skills, and ensure long-term retention of all the learned information. As commonly used in the literature, the term *item,* in this paper, refers to the piece of information to be learned.

Through the human history, many memorization techniques have been developed such as: **Rote (or by heart) learning**: This method is based on passive repetition; it is seen as the opposite to meaningful and critical learning [25]. It has been used in Vedic chant since as long as three thousand years ago [29].

Spaced repetition: whitch exploits the psychological spacing effect (vs cramming), and it consists of committing information into long-term memory by means of increasing time intervals between subsequent reviews of the previously learned material [34].

The lag effect [23]: is the related observation that people learn even better if the spacing between practices gradually increases [19].

Active recall: a learning method that exploits the (testing effect) i.e. the fact that memorization is more efficient if the to-be-learned information is optimally selected through testing using proper learner feedback.

Optimal human learning techniques have been extensively studied by researchers in psychology, Pedagogy [13] [11] and computer science [32], [40],
[39].

One of the first mathematical models for memorization is *the human forgetting curve* [14]. It defines the probability, over time, that a given learner will correctly recall a particular learned item. This probability is given by:

$$p = \left(\frac{1}{2}\right)^{\Delta T/h}$$

Where ΔT refers to the lag time i.e. time elapsed since the last exposure to the item, h is the item Half-Life, it corresponds to the value of the lag ΔT where the learner is likely on the verge of forgetting the item: p=0.5. A small value of ΔT means that the item is fresh in learner memory, thus the probability of remembering it is high. The value of h is updated after each learner exposure to the item and based on temporal distribution of the item reviews.

In Pimsleur model [27], the items are repeated at exponentially increasing intervals. And in Leitner System for flashcards [22]: The items are ranged in boxes, and each box has a reviewing periodicity. In this model, the new added items are assigned minimum periodicity. A number of spaced repetition software and online platforms have been developed to aid the learning process especially in language learning replacing the use of physical flashcards such as *Mnemosyne* [1], *Synap* [2], and *Duolingo* [3]. And in [34] the used model: MEMORIZE algorithm is based on stochastic differential equations with jumps.

The goal of an efficient memorization, for a learner, is to maximize the recall probability of all the items through calculation of the optimal reviewing time by determining which item would benefit the most from review.

Our proposed models We define the History of an item reviewing: $H=(p_i,a_i)_i$ witch is the sequence of proposed periodicities p_i and user feedback a_i. The user feedback has two possible values: remember (action=1), and forget (action=0) indicating whether the user succeeded to remember the item in question. The History of an item deduced from a database event-log table, which contains the history of the user events relative to all items.

Obviously, the more complex the item, the harder it is to remember, and the model learns the item complexity through its reviewing history.

Our first model In our first model, we assume that the two last states of an item (p_{i-1},a_{i-1}) and (p_i,a_i) provide a concise summery of the whole reviewing history for that item. Unlike other models, our model does not take into consideration the total number of times where the learner reviewed item.

Giving the two last reviewing states, the new proposed periodicity p_{i+1} is calculated based on the precedent states (p_{i-1}, a_{i-1}) and (p_i, a_i) as shown in the table below:

The four values in the table stem from the simple following idea: keeping increasing (resp decreasing) the periodicity while the two last user actions are

		a_{i-1}	
		1	0
a_i	1	$p_i \times 2$	$(p_{i-1} + p_i)/2$
	0	$(p_{i-1} + p_i)/2$	$p_i/2$

Fig.1. the periodicity transition

"Remember"(resp forget). If the user action changes, the new proposed periodicity is the mean of the two last periodicities using a Bisection approach [21].

Finally, we assume that all new added item have the same initial maximal difficulty inspired by Leitner System for flashcards [22]. This method of assigning difficulty to items is different for the one used by other online applications [3] where the default difficulty is deduced from other learners' experience.

Our second model When exposed to an item, we calculate the probability p that the user will correctly remember the item, this probability is compared the actual action of the user. In addition, we define the probability error error prob=actionp as the error in estimating the probability that the user will correctly recall the item. The error variable is then used to update the half-life of the reviewed item using this proposed formula:

$$HL \leftarrow \begin{cases} Max(\Delta T,\ (1+error\_prob).\ HL) & if\ action = 1 \\ Min(\Delta T,\ (1+error\_prob).\ HL) & if\ action = 0 \end{cases}$$

The next item to be reviewed in a time T is the one with the minimum probability that the learner remembers the item at time T, and adopting a stochastic -greedy approach [33].

3 SEGMENTATION

Automatic audio segmentation aims to divide a digital audio signal into segments [36], based on silence detection, content, or speaker identity.

One common automatic segmentation method consists on studying the amplitude envelop through detecting the peaks then connecting the successive ones linearly [31], or by interpolation [24]

In [37] the used idea is that the audio time-markers where the signal can be split correspond to sudden changes of values in most of features. A distancemetric between successive frames of the sound is used.

In this section, we will present our mono-channel audio segmentation method that is based on silence detection.

Unlike classical approach where the rules are hard-coded, our approach uses a supervised method using a labeled training dataset. The training dataset contains labeled data that were constructed based on the exhaustive list of silencemarker times in a speech audio file. The time markers were manually established and correspond to elements of the time line silence where the file can be split or segmented. There are two main Python libraries that can be used to import and process digital audio signal: scipy-lib [4], Librosa [5]. In our work, we used Librosa library.

Feature extraction In order to manipulate (i.e. segment) the audio signal data, we start by extracting features. The utility of extracting feature from an audio signal is to reduce the huge sound data of a raw waveform to a smaller set of parameters. Generally, before extracting the features, the audio signal is preceded by a preprocessing phase, which consists of filtering frequencies. One of the most used filtering method is the Low-Pass Filtering (LPF): [9], [28], given, for example, that typical the human have a fundamental frequency from 85 to 255 Hz [8].

Depending upon which audio features are used, we distinguish two main approaches for the feature extraction: time-domain, and spectral (frequency) domain. In the first approach, the features are calculated using a sliding time windows using one of different methods such as Root Mean Square (RMS) [17]. True Amplitude Envelope (TAE) [10]or Zero Crossing Rate (ZCR) [16], [26].

The second approach uses a frequency analysis: Frequency-Domain Linear Prediction (FDLP), and recently Mel-Frequency Cepstral Coefficients (MFCC) [18] [30]. MFCCs are commonly used in combination with hidden Markov models [20], k-NN and SVM [6], or Convolutional Neural Networks (CNN) for audio classification [12] and speech [7]

Other sound features are used in the literature: pitch, loudness timbre and harmonicity. A comparison of 16 primary features belonging to the two approaches is presented in [38].

Our Feature extraction approach is based on RMS witch is widely used in the literature as an estimation of the waveform amplitude envelop.

Root-Mean Square (RMS) Energy

RMS is used to estimate the amplitude envelope (evolution over time of the amplitude of a sound) applying instantaneous root mean square value of the waveform through a sliding window $w_i(t)$:

$$RMS\,(t) = \sqrt{\frac{1}{T}\sum_{i=1}^{T} w_i\,(t)\,x_i^2(t)}$$

Where $x_i(t)$ is the i^{th} sample of the signal centred around t as seen through the window $w_i(t)$, t is the number of samples the analysis, and T is the window length.

For a simple rectangular window, we can use a simplified formula:

$$A = \sqrt{\sum_{n=1}^{N} x^2(n)}$$

In [15] a Speech/Music Discriminator based on RMS is used. If the RMS value in a frame is less than a pre-determined threshold, it is regarded as a silence frame. And in [41] an adaptive threshold is used that depends on the mean value of the absolute amplitudes of a signal.

Far from the RMS approach, our first used method consists on using MFCC and CNN. A key drawback to this approach is that it requires a larger amount of training data to operate; otherwise we have the overfitting problem.

The RMS output sample rate (we used the default value of 2205 per second). And the RMS method is used to generate data in our Machine Learning model as described below.

Data structure

As we use a machine learning supervised algorithm, our dataset is composed of a predictive (input) data: X, and a target data: Y. An element of the X data is a sub-vector of the RMS vector with 12 dimensions, the first element contains the RMS value corresponding to a given time (element of time line). The Y data contains Boolean data; it indicates whether the time interval (time region) corresponding to the X entry does contain a silent.

Constructing the training dataset

In this phase of feature engineering, we describe training data structure. The training dataset is composed of two subsets, the first corresponds to the times whose associated vector contains silence (Yi=0). And the second corresponds to elements relative to speech (Yi=1).

For a given labeled silence time in the audio file, we use the RMS vector relative to the entire audio file. From this vector, we extract the five 12-dimension sub-vectors that are nearest to the element corresponding to this silence time. Each of these five

vectors is considered as containing a silence and thus is assigned the target value (Y=0). The second subsets of the training dataset (Y=1) is created using the same logic for an equidistance sample of time-points between two successive silence-times and sufficiently distant from these two silence-times.

Finally, we used he obtained training dataset to train a Support Vector Machine (SVM) supervised model.

Extracting silence-markers after our predictive model has been trained, it is used to extract silence-markers in an audio file in two steps. In the first step, we extract all 12- dimension sub-vectors of RMS-vector relative to an audio file that is recorded in similar conditions (volume, echo...) as those of the file on the training phase. The SVM binary classifier is applied to each of the sub-vectors. In a second step we calculate, for each element in the vector obtained in the first step, the number of previous elements of the vector considered by the SVM as silent (accumulation of continuous silence). Our classification algorithm predicts that the audio file can be segmented in a time-point if the number calculated in the second step is greater than an empirical threshold value of 10. Finally, we keep only the markers that are, at least, 1.5 seconds apart.

Silence-marker times types

We used a two-tiered hierarchy of silence-markers in an audio-file: the first one corresponds to the beginning of a bloc (eg paragraph, chapter...), and the second corresponds to the beginning of a sub-bloc (eg sentence, verse in a poem...) of a bloc. The information relative to the marker type introduced based on human judgement, which depends on content structure of the audio file. The time interval of a sub-bloc is delimited by times between two successive markers.

Although giving good segmentation result for a pure speech audio files, the algorithm described in this section is not usable with files containing noise or mixed content (music, noisy environment...).

The resulting list of times corresponding to silence of the audio file is stored to an external file then to a database which is used later by our mobile application.

4 A MOBILE APPLICATION FOR AUDIO MEMORISATION AND REVISION

Exploiting the widespread access to smartphones, we describe in this section our offline memorization mobile application for Android devices. The application represents a tool for self-learning, that can help memorizing and reviewing the content of audio speech files. The application uses a method based on listening and reciting the audio sequences on an active and adaptive way, by taking into consideration the user capability, expressed as feedback, rather than being restricted to real time sequential listening. The audio files can be of different content: Lectures, poems, religious texts..

The items to be memorized correspond to sequences of the audio file defined using the segmentation algorithm. The idea of memorizing individually the sequences and not the entire audio file is inspired from the Chunking Memorising method [35]. The application version is an implementation of the second method in the memorisation section in an audio context.

Database structure

The application uses a local database that comprises five tables:

Tab File: contains the paths and names of the audio files.

Tab SubBloc: indicates, for each marker, the corresponding time (in millisecond) in the corresponding audio file. The data in this table is the result of applying the algorithm in the section above.

Item: Contains all the audio sequences to be reviewed. Each of the sequences starts in a sub-bloc and finishes in another sub-bloc, and has an estimated half-life.

The user can customize a beginning and end times (T_b, T_e) if the times does not correspond to the beginning of a sub-bloc stemming from the segmentation depicted in the segmentation section. To do that we created two parameters that we named beginning percentage and end percentage. The new beginning and end times T_b, T_e are calculated using the formulates:

$$T_b = S_b + (S_{b+1} - S_b).\frac{P_b}{100}$$

and $$T_e = S_{e-1} + (S_e - S_{e-1}).\frac{P_e}{100}$$

Fig.2. Calculating times based on percentages

The figure above depicts the new beginning and end times corresponding percentages: p_b =50, and p_e=25. Obviously, the beginning and end times have the default values: p_b=0, p_e =100.

tabLog: contains the history of the user actions while revising the different items.

tabParameter: contains miscellaneous parameters of the application.

Memorization & Revision application interfaces

Besides general and auxiliary forms, the application interface contain two main forms, namely Memorization form and Revision form.

Memorization interface

Memorizing form allows defining, and adding a new item i.e. sequence. Through this form, the user first chooses the audio file, and specify the beginning and the end of the passage. For both beginning and end passages, the user specifies the bloc number, the sub-bloc number, and the number of times that the sequence will be played. A short customised silence separate stopping and replaying the sequence. For the number of times of repeating we use three parameters that we call "repeating numbers". These repeating numbers define the initial state of the player whose evolving states follows the state graph automate:

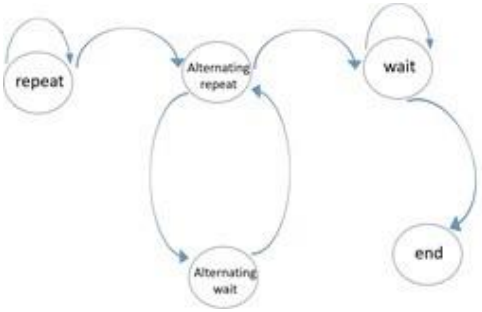

Fig.3. the player states transition

The First repeating number N1 indicates that the player, in a first step, simply plays the specified sequence N1 successive times. *The Second repeating number N2*:

Before defining the number N2, we first introduce the method decrease volume() that belongs to our created Player class. When invoked after playing a sequence, this function allows, after a delay, to decrease progressively the sound volume, using instance of the Java Timer class, after a specific duration, and when the volume is too low, the player is stopped. The charter bellow shows the look of the applied window.

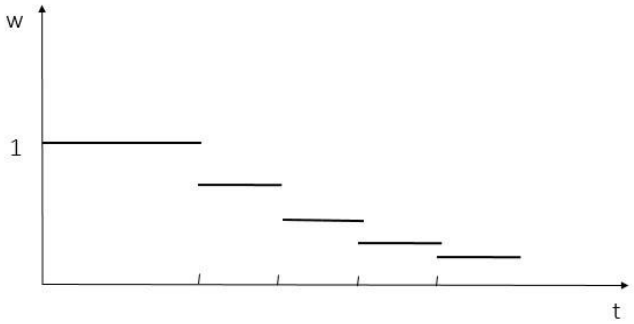

Fig.4. Window for decrease volume () method

This method is applied in both states "wait" and "alternating wait" and it gives to the user the time to try to finish (recite) the started audio sequence.

Fig.5. the wave form corresponding to a sequence

The figure 1 shows the waveform corresponding to the recording, from the device speakerphone, of a sequence. And the figure 2 corresponds to the same sequence when decrease volume () method is called.

In a second step the player alternates simple playing (the state Alternating repeat) and playing with decrease volume (the state alternating wait) N2 times; that is why it is preferable to choose N2 as an even number.

The third repeating number N3 indicates the number of times the sequence is played with application of the decrease volume() method(the state “wait”). Finaly the system move to the “end” state.

Using the button “add”, the sequence with the user parameters defined above is added to a non-activated list. The elements of this list can be activated later through the menu “activation”. The activated sequences list is used by the re-

Fig.6. the wave form with decrease volume () method

Vision interface. When activated, an item is attributed a default haf life=30 seconds.

The button “Next” displays the sub-bloc whose beginning time corresponds to the current sequence end time.

Revision interface

This interface is designed in a dark mode, and it uses swipe events, so it can be used even without looking at the device screen, or when the user is performing other activities; this also allows better accessibility for the visually impaired people. The interface displays a summary of current item, and has five user event listeners that can be triggered in any place in the screen:

1. **Right to Left swipe**: allows playing the current sequence with decrease volume. The user try then to complete the sequence. The played sequence is selected using the getNextItem() method detailed in the next paragraph.
2. **Left to Right swipe**: this event allows playing the current sequence, which is the same as the one played in event above, then the user compares the played sequence with his answer. Depending on whether, his answer is correct or not, the user initiate one of the two events bellow.
3. **Bottom to Top swipe**: The user initiates this event when he estimates that he memorized current sequence, so the periodicity of reappear in the future is reduced.
4. **Top to Bottom swipe**: In the opposite of the event above, this one is called when the sequence needs to be reinforced.
5. **Double tap**: allows to stop playing the sequence.

For performance reasons, if the audio file corresponding the current and precedent item is the same, the audio file is not reloaded since it is already present in the device memory.

The items used in this interface can be introduced either individually using the Revision interface, or automatically loaded in the database using a script variant that extract the times relative to successive overlapping sequences with a customizable minimum duration in an audio file with sequential content, as illustrated in the figure below:

The latter possibility is created essentially to relieve visually impaired people of using the Memorization interface.

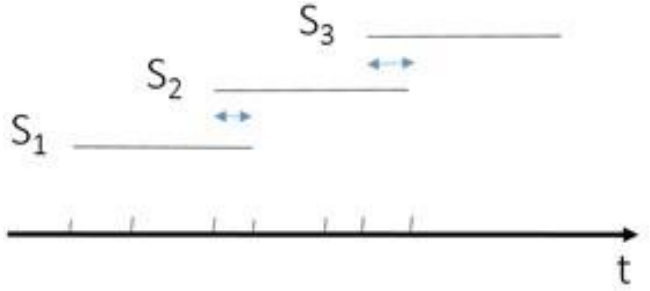

Fig.7. extracting the times relative to successive overlapping sequences

Manipulating the items

The most two interesting method to manipulate item objects are getNextItem() and execute action() methods.

getNextItem()

The getNextItem() method returns the weakest item that currently requires more reinforcement. It is a static method(according to Object-oriented programming) of the class Item, and the returned item is the one with the minimum remembering probability . **execute _action(action)**

When applied to an item object, the execute action() method updates the current item half life illustrated in the first section.

CONCLUSION

Through this paper, we have presented our audio memorisation approach. Some questions need more attention: In our solution as well as the other solutions, to the best of our knowledge, the learner feedback are restricted to only 2 choices “remember” and “forget”. So an other alternative can consist on introducing a scale value allowing to learner to indicate his degree easiness to remember the learned item.

Also our solution does not take into consideration the interrelation between the learned items where reviewing an item can increase, or decrease, the half-life relative to an other item.

References

1) mnemosyne-proj.org
2) www.synap.ac
3) www.duolingo.com
4) https://www.scipy.org/docs.html
5) https://librosa.org/doc
6) Ali, M.A., Siddiqui, Z.A.: Automatic music genres classification using machine learning. International Journal of Advanced Computer Science and Applications (IJACSA) **8**(8), 337–344 (2017)
7) Ashar, A., Bhatti, M.S., Mushtaq, U.: Speaker identification using a hybrid cnnmfcc approach. In: 2020 International Conference on Emerging Trends in Smart Technologies (ICETST). pp. 1–4. IEEE (2020)
8) Baken, R.J., Orlikoff, R.F.: Clinical measurement of speech and voice. Cengage Learning (2000)
9) Bello, J.P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., Sandler, M.B.: A tutorial on onset detection in music signals. IEEE Transactions on speech and audio processing **13**(5), 1035–1047 (2005)
10) Caetano, M., Rodet, X.: Improved estimation of the amplitude envelope of timedomain signals using true envelope cepstral smoothing. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 4244–4247. IEEE (2011)
11) Dempster, F.N.: Spacing effects and their implications for theory and practice. Educational Psychology Review **1**(4), 309–330 (1989)
12) Dong, M.: Convolutional neural network achieves human-level accuracy in music genre classification. arXiv preprint arXiv:1802.09697 (2018)
13) Dunlosky, J., Rawson, K.A., Marsh, E.J., Nathan, M.J., Willingham, and D.T.: Improving students’ learning with effective learning techniques: Promising directions from cognitive and educational psychology. Psychological Science in the Public interest **14**(1), 4–58 (2013)
14) Ebbinghaus, H.: Memory: a contribution to experimental psychology. Teachers college, columbia university, new york. Trans. HA Ruger and CE Bussenius. Original work published (1885)
15) Ganapathiraju, A., Webster, L., Trimble, J., Bush, K., Kornman, P.: Comparison of energy-based endpoint detectors for speech signal processing. In: Proceedings of SOUTHEASTCON’96. pp. 500–503. IEEE (1996)
16) Gouyon, F., Pachet, F., Delerue, O., et al.: On the use of zero-crossing rate for an application of classification of percussive sounds. In: Proceedings of the COST G-6 conference on Digital Audio Effects (DAFX-00), Verona, Italy. vol. 5, p. 16. Citeseer (2000)
17) Hajda, J.: A new model for segmenting the envelope of musical signals: The relative salience of steady state versus attack, revisited. In: Audio Engineering Society Convention 101. Audio Engineering Society (1996)
18) Hunt, M., Lennig, M., Mermelstein, P.: Experiments in syllable-based recognition of continuous speech. In: ICASSP’80. IEEE International Conference on Acoustics, Speech, and Signal Processing. vol. 5, pp. 880–883. IEEE (1980)
19) Kahana, M.J., Howard, M.W.: Spacing and lag effects in free recall of pure lists. Psychonomic Bulletin & Review **12**(1), 159–164 (2005)
20) Kimber, D., Wilcox, L., et al.: Acoustic segmentation for audio browsers. Computing Science and Statistics pp. 295–304 (1997)
21) Kincaid, D., Kincaid, D.R., Cheney, E.W.: Numerical analysis: mathematics of scientific computing, vol. 2. American Mathematical Soc. (2009)
22) Leitner, S.: So lernt man lernen:[angewandte Lernpsychologie-ein Weg zum Erfolg]. Herder (1991)
23) Melton, A.W.: The situation with respect to the spacing of repetitions and memory. Journal of Verbal Learning and Verbal Behavior **9**(5), 596–606 (1970)
24) Meng, Q., Yuan, M., Yang, Z., Feng, H.: An empirical envelope estimation algorithm. In: 2013 6th International Congress on Image and Signal Processing (CISP). vol. 2, pp. 1132–1136. IEEE (2013)

25) Mulnix, J.W.: Thinking critically about critical thinking. Educational Philosophy and theory **44**(5), 464–479 (2012)
26) Panagiotakis, C., Tziritas, G.: A speech/music discriminator based on rms and zero-crossings. IEEE Transactions on multimedia **7**(1), 155–166 (2005)
27) Pimsleur, P.: A memory schedule. The Modern Language Journal **51**(2), 73–75 (1967)
28) Potamianos, A., Maragos, P.: A comparison of the energy operator and the hilbert transform approach to signal and speech demodulation. Signal processing **37**(1), 95–120 (1994)
29) Scharfe, H.: Education in Ancient India. Handbuch der Orientalistik: Indien, E.J. Brill (2002), https://books.google.co.ma/books?id=BmK-wAEACAAJ
30) Scheirer, E., Slaney, M.: Construction and evaluation of a robust multifeature speech/music discriminator. In: 1997 IEEE international conference on acoustics, speech, and signal processing. vol. 2, pp. 1331–1334. IEEE (1997)
31) Schloss, W.A.: On the Automatic Transcription of Percussive Music–From Acoustic Signal to High-level Analysis. Ph.D. thesis, Stanford University (1985)
32) Settles, B., Meeder, B.: A trainable spaced repetition model for language learning. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers). pp. 1848–1858 (2016)
33) Sutton, R.S.: andrew g. barto. reinforcement learning (1998)
34) Tabibian, B., Upadhyay, U., De, A., Zarezade, A., Scho¨lkopf, B., Gomez-Rodriguez, M.: Enhancing human learning via spaced repetition optimization. Proceedings of the National Academy of Sciences **116**(10), 3988–3993 (2019)
35) Thalmann, M., Souza, A.S., Oberauer, K.: How does chunking help working memory? Journal of Experimental Psychology: Learning, Memory, and Cognition **45**(1), 37 (2019)
36) Theodorou, T., Mporas, I., Fakotakis, N.: An overview of automatic audio segmentation. International Journal of Information Technology and Computer Science (IJITCS) **6**(11) (2014)
37) Tzanetakis, G., Cook, F.: A framework for audio analysis based on classification and temporal segmentation. In: Proceedings 25th EUROMICRO Conference. Informatics: Theory and Practice for the New Millennium. vol. 2, pp. 61–67. IEEE (1999)
38) Won, M., Alsaadan, H., Eun, Y.: Adaptive multi-class audio classification in noisy in-vehicle environment. arXiv preprint arXiv:1703.07065 (2017)
39) Zaidi, A., Caines, A., Moore, R., Buttery, P., Rice, A.: Adaptive forgetting curves for spaced repetition language learning. In: International Conference on Artificial Intelligence in Education. pp. 358–363. Springer (2020)
40) Zaidi, A.H., Moore, R., Briscoe, T.: Curriculum q-learning for visual vocabulary acquisition. arXiv preprint arXiv:1711.10837 (2017)
41) Zhang, J.X., Whalley, J., Brooks, S.: A two phase method for general audio segmentation. In: 2009 IEEE International Conference on Multimedia and Expo. Pp.626–629. IEEE (2009)