International Journal of User-System Interaction 15(2) 2022, 38-50 © MatrixRom

Enhancing an End-to-end Romanian Speech Recognition Model for Persons with Speech Impairments

Elena Pelican, Lucian Odainic

Ovidius University of Constanta
Bd. Mamaia, nr. 124, Constanta 900527
E-mail: epelican@univ-ovidius.ro

Abstract. In this paper we propose an approach to improve a deep learning model, built for speech recognition in Romanian language. The improvements have been made for serving a person with speech difficulties, as already existing model have proved unsatisfactory results. We have updated the weights on new collected data from the person in question. In order to make the model more robust, audio data was modified by adding noise and changing the speed by making it slower and then faster. The system relies on a deep learning model similar to DeepSpeech2 that contains convolutional and recurrent neural networks. The layer with CNN requires image data, so the raw recorded waveform was converted into Mel spectrograms, a visual representation of the sound. The resulting model has shown to be more appropriate for solving the problem under consideration, so we have incorporated it within an Android application. This generates transcriptions of audio files that are registered in real time containing Romanian speech. The Android application was inspired by Google Live Transcribe, which, when testing it, the transcriptions had imperfections. We thought of colliding the enhanced model and the idea of a live transcriber to have the solution that outperforms the existing products and suits perfectly our case.

Keywords: Computational linguistics, Deep learning, Romanian language, Speech recognition, Speech-to-text.

DOI: 10.37789/ijusi.2022.15.2.2

1. Introduction and Motivation

This work was inspired by a real world situation, a problem that a young lady now in her twenties was encountering on a daily basis. The person in question has always been struggling with strangers not understanding her. She was born with Palatoschisis, a type of orofacial clefts, which are characterized by splits or openings in the upper lip, gum line, and roof of the mouth. This medical condition has caused speech impairments such as nasal speaking voice, hazy sounds while pronouncing the words and poor articulation of speech sounds. Even after surgeries and speech-language therapy this problem persisted. So, this paper describes the proposed solution by reusing a deep learning model in order to capture different manners of speaking.

We were inspired to try the Google Live Transcribe after finding the paper published by Loizides, Basson, Kanevsky and et al. (2020). They described various and authentic ways of using the app for deaf and hard of hearing (DHH) people. Another quick discovery was personalizing English Automatic Speech Recognition (ASR) for people with amyotrophic lateral sclerosis (ALS) and accented speech as presented by Shor, Emanuel, Lang and et al. (2019). They also used fine tuning on the existing neural network to grasp the new phonemics and incorporated it in a live transcriber app. In our case, the application would serve the same purpose of using it real time during casual conversation or job interviews to help our person to be understood. Unfortunately, it has shown pretty bad results, meaning low accuracy transcriptions. Thus, it was not helping the situation at all. We decided to build our own speech recognition system.

Using deep learning for building ASR systems instead of hidden Markov models, has proven tremendous quality progress (Hinton et al., 2012; Deng and Yu, 2014; He, Gao and Deng, 2014; Hirschberg and Manning, 2015) and now it is a great technique being fully used in building such systems (Jurafsky and Martin, 2009; Deng and Li, 2013; Benesty, Sondhi and Huang, 2008). Thus, using personal digital assistants, transcribing video or audio files became a common thing and solving daily tasks can be done much easier nowadays. Speech recognition has taken human-computer interaction a step forward making it more natural, but it requires continuous improvement as human's abilities vary. Furthermore, multilingual ASR systems are used within personal assistants like Siri, Alexa, Google assistant, Cortana (Avdic and Vermeulen, 2020; Sainath et al., 2017), or language translation software (Di Gangi, Negri and Turchi, 2019).

The deep learning model for speech recognition for Romanian language was used within an Android app. The app itself takes the form of a live transcriber, as the person is speaking, the actual text will appear on the screen (Nakadai et al., 2015). The work related to Romanian ASR systems considered a project called ROBIN, for human-robot interaction (Tufis et al., 2019) and then a better version of it presented by Avram, Pais & Tufis (2020) deployed as a REST web service.

The model structure is based on DeepSpeech2 architecture (Amodei et al., 2016) and it was already trained for Romanian language (Avram, Pais and Tufis, 2020). They have achieved a WER of 9.91% and a CER of 2.81% training the model on 230 hours of audio and using their own language model trained on 10.2 GB of text using KenLM (Heafield, 2011) tool to correct prediction errors. After testing the model on our own prepared dataset with recordings of people with no speech impediments, the model issued a WER of 4.464% and CER of 0.548%. We also tried the new recorded test dataset containing the clips of the person with speech deficiency and the results came out completely different. Thus, the value for WER was 88.75% and for CER 62.81%. So, we started to finetune the whole model with the new data.

The remainder of the paper is organized as follows. Section 2 describes the dataset structure used for retraining the speech recognition model and Section 3 holds the network training process; the results during the evaluation step are presented in Section 4. The model is used within an Android app, described in Section 5 and the sixth section concludes the paper.

2. Dataset

For the beginning, we have collected recordings of the person with speech deficiency in order to upgrade the model performance.

First of all, we have gathered the text that has to be pronounced and recorded, from different web sources, especially Common Voice Corpus (version 6.1). The final text dataset consists of 9697 sentences in Romanian language. To have less symbols to generate and for correctly serving the language model the text was cleaned up. We have erased all the punctuation marks (e.g. "m-am" or "Buna!" became "m am" and "Buna" respectively). Now the sentences are ready to be encoded to fit the model properly.

The text was converted into numbers, so each sentence has its own array of numbers that corresponds to specific letters. Thus, we needed to encode 33 characters equal to 31 symbols in the Romanian alphabet, plus 1 for space and the blank character for the decoding step. Character encoding makes it easier for the speed of the decoding process as it will not operate with thousands of words, but with dozens of characters instead.

The collected text and its encode is stored in a csv file alongside the name of the audio file that contains the recorded voice of the person who has read the sentence.

The person in question recorded 2363 audio files on a mobile phone in wav format. The recordings were done within a sample rate of 16 kHz and bit depth of 16 bits. Sample rate denotes the frequency where the sound was captured and the bit depth of 16 bits means that there are 2^{16} discrete values for the loudness level or a dynamic range of 96 dB.

2.1. Dataset Augmentation

Data augmentation has proven to increase model performance in computer vision (Lecun, Huang and Bottou, 2004; Sapp, Saxena and Ng, 2008; Coates et al., 2011) and speech recognition (Chu et al., 2019; Hannun et al., 2014). Therefore, in order to make the model more robust with less raw data, we have applied some data augmentation techniques to slightly diversify the dataset. For instance, we have used noise injection to introduce the background noise that was not recorded in real time. Another method of data augmentation we have used was speech speed alteration, as the person could speak the same words a bit faster or slower in real life.

Finally, the dataset includes 9452 audio files equivalent to 13 hours of speech.

Artificially increasing the number of audio clips diversifies the data without actually collecting it and contributes to the model improvement. However, the resulting audio clips cannot replace real data. The generation of factitious data should not be made pointlessly, in fact the best augmentation techniques are revealed after analyzing existing dataset.

2.2. Mel Spectrograms

Now that all the audio clips are ready and the csv file holds their encoded text labels, the next step is changing the structure of the audio files in order to feed them to the neural network.

First of all, we have computed the Mel frequency cepstral coefficients (MFCC) (Kamath, Liu and Whitaker, 2019).

Mel spectrograms are a visual representation of the sound and also the most common features used for ASR. Their success is based on the ability to mimic the human auditory system perception of the sound. Each 20 ms windowed audio sample went through Fourier Transform and then the resulting spectrogram was applied the Mel Scale to convert frequencies into mels. Various studies have shown that the human ear cannot perceive sound on a linear scale. Humans can detect the sound signal more efficiently at low frequencies than at high frequencies. For example, it is easier to detect the jump 700-1200 Hz than 12000-12500 Hz, although the distance is the same.

Thus in 1937 Volkman, Newman and Stevens proposed a solution that makes the equal distances between two extremes of frequency to be easily received by the listener. In other words, the Mel Scale presses the frequency domain to detect effortlessly its variation.

The dataset consisting of 2363 original audio files was divided into 2280 files for the training set and the remaining 83 audio clips made up the testing set. After applying data augmentation techniques on the training dataset, our collection grew up to 9120 files. Thus, the network was trained on a bit more than 12 hours of speech.

3. Training

The model architecture is not modified to its originals, as shown in the Figure 1.

It consists of 2 input layers of Convolutional Neural Networks (Lecun et al., 1998), 4 layers of Long Short-Term Memory (Sochreiter and Schmidhuber, 1997) type of Recurrent Neural Networks and one fully connected layer. Additionally, batch normalization is used within the network in order to accelerate its training (Ioffe and Szegedy, 2015). Because predicted text may have repeated characters, the most used loss function used to train speech recognition models is Connectionist Temporal Classification (CTC) (Graves, Fernandez, Gomez, and Schmidhuber, 2006). Keeping the architecture unchanged was significant in order to be able to reuse the model with its weights as a starting point for a new dataset.

The purpose was to update the model on fresh, slightly different data. Thus, we had to decrease the learning rate from its original of $2 * 10^{-4}$ to $2 * 10^{-6}$, after some attempts we have found this value the best for updating the model weights on new data, as shown in Figure 2.

Decreasing the learning rate value keeps the already existing model stable and does not wash the previous accumulated performance as it usually happens when retraining the model on fresh data with the same learning rate.

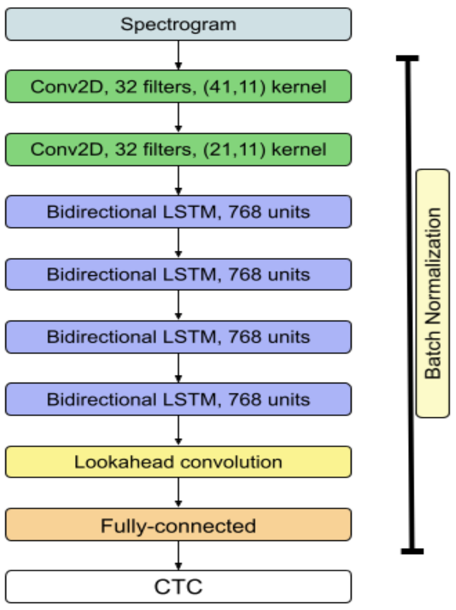

Spectrogram
Conv2D, 32 filters, (41,11) kernel
Conv2D, 32 filters, (21,11) kernel
Bidirectional LSTM, 768 units
Bidirectional LSTM, 768 units
Bidirectional LSTM, 768 units
Bidirectional LSTM, 768 units
Lookahead convolution
Fully-connected
CTC
Batch Normalization

Figure 1. Model architecture

The training process itself took place on Google Colaboratory Free version. It was managed by GPUs like Nvidia K80s, T4s, P4s or P100s with 12 GB of RAM as GPU training has shown breakthrough results (Raina, Madhavan and Ng, 2009; Krizhevsky, Sutskever and Hinton, 2017; Coates, Huval, Wang, Wu, Catanzaro and Ng, 2013) and 60 GB of disk space. The model was trained on 100 epochs with a batch size of 64 that used 6 GB of memory.

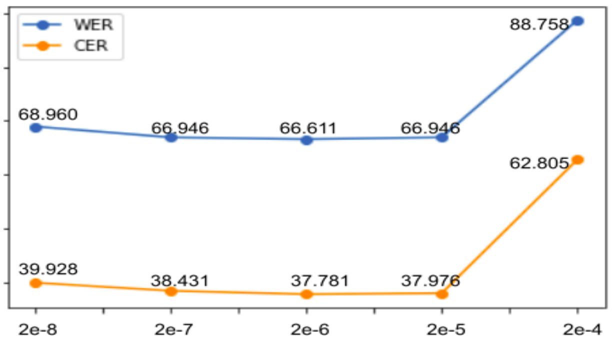


Figure 2. Results after training with different values of learning rate

The training process took about 30 hours, but with the platform limit of 12 hours of continuous training, we splitted it into 3 sessions. In case of interrupting the training trial due to losing internet connection or passing the time limit, all the weights were saved in a .pth file after each epoch. Saving the model state helps to resume its training easier and to use it for further evaluation and deployment.

4. Evaluation

In order to evaluate the system performance, we made use of the Levenshtein distance, which is applied especially on text (Haldar and Mukhopadhyay, 2011). The Levenshtein distance between two words is the minimum number of character edits (inserts, deletions, additions) needed to change a word into another.

The following discussed WER and CER are the performance metrics we have benefited from and they are derived from this kind of distance. WER is word-level and CER is character-level Levenshtein distance between two text sequences.

4.1. Word Error Rate (WER)

WER aligns the actual and predicted text to calculate the difference of these sequences by their words (Graves and Jaitly, 2014) and its value can be computed with the following formula 1.

$$WER = \frac{S+I+D}{N} \tag{1}$$

where

- I = number of insertions,
- D = number of deletions,
- S = number of substitutions,
- N = number of words in the text sequence.

4.1. Character Error Rate (CER)

CER assigns the actual text and the text resulted by the model to calculate how different the sequence is by characters (Graves and Jaitly, 2014) and the following formula 2 can manage this.

$$CER = \frac{S+I+D}{N} \tag{2}$$

where
- I = number of insertions,
- D = number of deletions,
- S = number of substitutions,
- N = number of characters in the text sequence.

Thus, after running the model on the test dataset along with the language model, we have obtained a WER value of 18.42 and CER equal to 6.04. Some successful and less successful snippets of the resulting text are demonstrated in the following table 1.

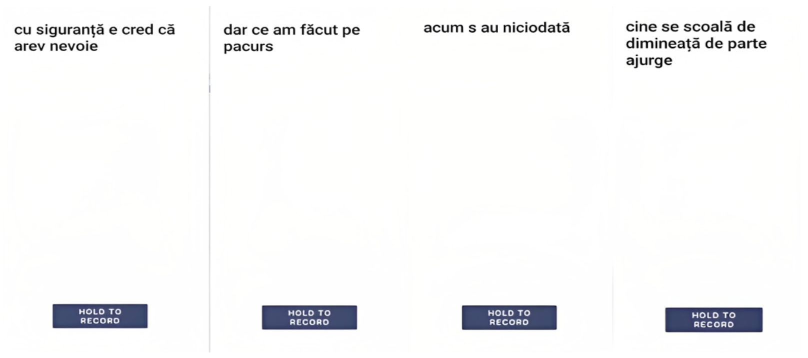
Table 1. Predictions during evaluation

The actual text	The model predicted text
	With no errors
fiecare stat are dreptul să-și declare rezervele	fiecare stat are dreptul să şi declare rezervele
să fim realiști nu există securitate absolută	să fim realiști nu există securitate absolută
să îl lăsăm să acţioneze conform acestui principiu	să îl lăsăm să acţioneze conform acestui principiu
acum discutăm desigur despre preţul combustibililor	acum discutăm desigur despre preţul combustibililor
acest lucru trebuie să se schimbe domnule președinte	acest lucru trebuie să se schimbe domnule președinte
eu însumi voi prezida această dezbatere	eu însumi voi prezida această dezbatere
	With errors
aceste şase puncte includ alte propuneri	acestea şase putin mundi alte propuneri
acesta este un mesaj extrem de periculos	acesta este un sariputra de periculos
ce oameni celebri s au născut în martie	ce o am în celemi s a născut în martie
sunt membru al unei săli de sport	suntem al unei să de sport
desigur până acum nimic nu este nou	desigur până acum nimeni nu este nu
nu aruncaţi materiale plastice în mare	nu arati materiale flasce în mare

5. Deployment

The model is embedded in an Android application in order to benefit from its functionality. The development of the application took place in Android Studio with the Java programming language. The app can run on Android 8.0 version and up. Because the deep learning model requires special computing resources we have included the access to the Google Cloud Platform. The Android connection with Google Cloud Platform (GCP) is made using the Firebase intermediary, in result, the application can not work without an internet connection. Firebase and GCP projects have to be linked in order to sync the storage needed to keep the audio files.

The app functionality pipeline is as follows, while holding the button pressed the microphone is recording an mp3 file. After, the function located on GCP is triggered by a HTTP request. The .json file sent within a POST method contains the audio file name. The cloud function reads the file with specified name from storage, converts it to wav format, then runs the deep learning model and sends back a .json with the resulting text. The final prediction is shown on the screen as shown in Figure 3.


cu siguranță e cred că arev nevoie

dar ce am făcut pe pacurs

acum s au niciodată

cine se scoală de dimineață de parte ajurge

Figure 3. Result in the Android app

6. Discussion and Conclusion

The paper presents the process of developing a speech recognition system for people with speech impairments by carefully fine tuning a Romanian healthy-speech model and using it in an Android app similar to a Live Transcriber. The related work including the existing Romanian ASR model has shown unsatisfactory results on impaired speech. It had a value of 88.75% for WER and 62.81% for CER being trained on 210 hours of speech. We showed that by collecting some recordings and fine tuning the model we can get better results and become more inclusive towards people with speech impairments. We saw a significant improvement of roughly five times lower error rates such as WER value of 18.42% and CER equal to 6.04% with only 12 hours of speech.

The ASR system is embedded in a user-friendly Android app interface, ensuring ease of use and accessibility for a wide range of users. One of its significant strengths lies in its ability to enhance real time dialogue for individuals with speech impairments. The model itself can be used in many other ways than within a live transcriber, with the purpose of improving various areas of use for people with speech deficiencies. For instance, it can help generate captions for videos in which people with speech impairments talk or for improving the personal assistants voice recognition to have better results when requesting.

While the ASR system presents valuable strengths, it also possesses certain weaknesses that impact its performance and utility. The resulting text may exhibit imperfections, including occasional misspellings, due to the small amount of training data we provided. Also, the accuracy of transcriptions is threatened by loud and noisy background environments, affecting the overall user experience. The system's model has been fine-tuned primarily using data from one individual with Palatoschisis. This narrow focus may lead to inaccuracies and errors when used with other individuals, as testing on other individuals has not been conducted.

The Android application faces certain challenges and threats that could impede its effectiveness and user experience. The neural network's complexity poses a challenge for offline, on-device use. The requirement for an internet connection to operate the app limits its accessibility and utility in scenarios with no connectivity. Also, all the steps and computational resources required slows down the overall speed of using the application, the average latency being 1-2 seconds.

The model has potential avenues for growth and improvement that could further enhance its capabilities and impact. It is very flexible and efficiently built. Thus, it can be trained on data taken from other people with speech difficulties. The model can be taken as a core that can be grown to capture more speech styles. By aggregating various impaired voices into a single model, the system's accuracy and performance could be significantly improved. In order to do that it is enough to pass through the path we have discussed above.

As future work we consider implementing a trigger behind the application that will start training the network when the generated text has a value of WER or CER associated in order to continue enhancing the model. As more training data is the key that will help improve the overall accuracy.

Also, we are looking to adjust the cloud computing resources to reduce the latency, or even changing the model architecture to suit the smartphones hardware in order not to depend on internet connectivity.

References

Amodei D., Ananthanarayanan S., Anubhai R., Bai J., Battenberg E., et al. (2016) Deep Speech 2: End-to-end speech recognition in English and Mandarin. In Proceedings of The 33rd International Conference on Machine Learning, 48: 173–182, PMLR, https://doi.org/10.48550/arXiv.1512.02595

Avram A.-M., Pais V., Tufis D. (2020) Towards a Romanian end-to-end automatic speech recognition based on deepspeech2. In *Proceedings of the Romanian Academy*, Series A, 395–402. Romanian Academy, Publishing House of the Romanian Academy.

Avdic M., Vermeulen J. (2020) Intelligibility issues faced by smart speaker enthusiasts in understanding what their devices do and why. In *32nd Australian Conference on Human-Computer Interaction*, 314–328. Association for Computing Machinery.

Benesty J., Sondhi M.M., Huang Y. (eds.) (2008) *Springer Handbook of Speech Processing*. Springer.

Coates A., Huval B., Wang T., Wu D., Catanzaro B., Ng A. (2013) Deep learning with COTS HPC systems. In *Proceedings of the 30th International Conference on Machine Learning*, 28: 1337–1345, PMLR.

Coates A., Carpenter B., Case C., Satheesh S., Suresh B., Wang T., Wu D.J., Ng A.Y. (2011) Text detection and character recognition in scene images with unsupervised feature learning. In *International Conference on Document Analysis and Recognition*, 440–445. IEEE, DOI: 10.1109/ICDAR.2011.95

Chu M., Liu X., Gong R., Zhao J. (2019) Support vector machine with quantile hyper-spheres for pattern classification. *PLOS ONE*, 14: 1–29, DOI: 10.1371/journal.pone.0212361

Deng L., Yu D. (2014) Deep learning: Methods and applications. Foundations and Trends® in *Signal Processing*, 7: 197–387, DOI:10.1561/2000000039.

Deng L., Li X. (2013) Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21: 1060–1089, DOI: 10.1109/TASL.2013.2244083.

Di Gangi M., Negri M., Turchi M. (2019) Adapting Transformer to end-to-end spoken language translation. In Proc. *Interspeech 2019*, 1133–1137, International Speech Communication Association, DOI: 10.21437/Interspeech.2019-3045.

Graves A., Fernandez S., Gomez F., Schmidhuber J. (2006) Connectionist temporal classification: Labeling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, 369–376, DOI: 10.1145/1143844.1143891

Graves A., Jaitly N. (2014) Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning*, 32:

1764–1772, PMLR

Haldar R., Mukhopadhyay D. (2011) Levenshtein distance technique in dictionary lookup methods: An improved approach, *arXiv*, https://doi.org/10.48550/arXiv.1101.1232

Hannun A.Y., Case C., Casper J., Catanzaro B., Diamos G., Elsen E., Prenger R., Satheesh S., Sengupta S., Coates A., Ng A.Y. (2014) Deep Speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567, https://doi.org/10.48550/arXiv.1412.5567

Heafield K. (2011) KenLM: Faster and smaller language model queries. In Proceedings of the *Sixth Workshop on Statistical Machine Translation*, 187–197, Association for Computational Linguistics.

He X., Gao J., Deng L. (2014) Deep learning for natural language processing and related applications (tutorial at icassp). In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP).

Hirschberg J., Manning C.D. (2015) Advances in natural language processing. *Science*, 349: 261–266.

Hinton G., Deng L., Yi D., Dahl D.E., Mohamed A.R., Jaintly N., Senior A., Vanhoucke V., Nguyen P., Sainath N., Kingsbury B. (2012) Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29: 82–97, DOI: 10.1109/MSP.2012.2205597.

Hochreiter S., Schmidhuber J. (1997) Long Short-Term Memory. *Neural Computation*, 9: 1735–1780.

Ioffe S., Szegedy C. (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Jurafsky D., Martin J.H. (2009) *Speech and Language Processing* (2nd Edition). Prentice-Hall Inc.

Krizhevsky A., Sutskever I., Hinton G.E. (2017) ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84–90, DOI: 10.1145/3065386,

Kamath U., Liu J., Whitaker J. (2019) *Deep Learning for NLP and Speech Recognition, volume 84, chapter 8*. Springer Publishing Company, Incorporated.

LeCun Y., Bottou L., Bengio Y., Haffner P. (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86: 2278–2324, DOI: 10.1109/5.726791

LeCun Y., Huang F., Bottou L. (2004) Learning methods for generic object recognition with invariance to pose and lighting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2: II–97-104, IEEE, DOI:10.1109/CVPR.2004.1315150.

Li B., Sainath T., Narayanan A., Caroselli J., Bacchiani M., Misra A., Shafran I., Sak H., Pundak G., Chin K., Sim K., Weiss R., Wilson K., Variani E., Kim C., Siohan O., Weintraub M., McDermott E., Rose R., Shannon M. (2017) Acoustic modeling for Google Home. In *INTERSPEECH*, 399–403, International Speech Communication Association.

Loizides, F., Basson, S., Kanevsky, D., Prilepova, O., Savla, S., & Zaraysky, S. (2020, October). Breaking boundaries with live transcribe: Expanding use cases beyond standard captioning scenarios. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility* (pp. 1-6), DOI: 10.1145/3373625.3417300.

Nakadai K., Okuno H.G., Ogata T., Noda K., Yamaguchi Y. (2015) Audio-visual speech recognition using deep learning. *Applied Intelligence*, 42: 722–737, https://doi.org/10.1007/s10489-014-0629-7

Raina R., Madhavan A., Ng A.Y. (2009) Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 873–880. Association for Computing Machinery, DOI: 10.1145/1553374.1553486

Sapp B., Saxena A., Ng A. (2008) A fast data collection and augmentation procedure for object recognition, 1402–1408, in AAAI Press.

Shor, J., Emanuel, D., Lang, O., Tuval, O., Brenner, M., Cattiau, J., ... & Matias, Y. (2019). Personalizing ASR for dysarthric and accented speech with limited data, *Proc. INTERSPEECH 2019*, DOI: 10.21437/interspeech.2019-1427.

Tufis D., Mititelu V.B., Irimia E., Mitrofan M., Ion R., Cioroiu G. (2019) Making pepper understand and respond in Romanian. In 2019 *22nd International Conference on Control Systems and Computer Science*, 682–688. Institute of Electrical and Electronics Engineers (IEEE), DOI: 10.1109/CSCS.2019.00122.

Recommendations for authors

The submissions should be written according to the format of the journal (http://rochi.utcluj.ro/ijusi/authors_instructions), and sent for review by e-mail, in PDF format, to one of the editors-in-chief. The final version of accepted papers must be sent in DOC or DOCX format.

Each paper is reviewed by at least 3 independent reviewers. The authors will receive a notification regarding the paper's acceptance/rejection. The acceptance could be conditioned by minor/major changes required by the reviewers. If major changes are required then the paper will be reviewed again.

The length of the paper should not exceed 20 pages (7000 words). The recommended length is 12-16 pages (even number of pages). Submissions should be drafted according to the template file available for download on this web page

Contact

Dorian Gorgan, *Technical University of Cluj-Napoca, Romania*
Phone: +40 264 401478, E-mail: dorian.gorgan@cs.utcluj.ro
Costin Pribeanu*, AOSR Bucharest, Romania*
Phone: +40 31 1070659*, E-mail: costin.pribeanu@gmail.com*
Jean Vanderonckt*, Universite Catholique de Louvain, Belgium*
Phone: +32 10 478525, E-mail: jean.vanderdonckt@uclouvain.be

Publisher

MATRIX ROM, C.P. 16-162, 062510 – Bucharest, Romania,
Phone: +40 21 4113617, Fax.+40 21 4114280, E-mail: office@matrixrom.ro