EDUCACIÓN

SECRETARÍA DE EDUCACIÓN PÚBLICA

Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

Research in Computing Science

Vol. 150 No. 9
September 2021

Research in Computing Science

Research in Computing Science, Año 20, Volumen 150, No. 9, septiembre de 2021, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de septiembre de 2021.

Research in Computing Science, year 20, Volume 150, No. 9, September 2021, is published monthly by the Center for Computing Research of IPN.

Volume 150(9)

Advances in Artificial Intelligence

Grigori Sidorov (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2021

ISSN: in process

Formerly ISSNs: 1870-4069, 1665-9899

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.rcs.cic.ipn.mx
http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

ISSN 1870-4069

Prototype of a Robotic System to Assist the Learning Process of English Language with Text Generation through DNN

Carlos Morales-Torres[1], Mario Campos-Soberanis[1,2], Diego Campos-Sobrino[1,2]

[1] Universidad Politécnica de Yucatán,
Ucú, Yucatán,
Mexico

[2] SoldAI Research, Mérida, Yucatán,
Mexico

danniel.torres99@gmail.com,
{mcampos,dcampos}@soldai.com

Abstract. In the last ongoing years, there has been a significant ascending on the field of Natural Language Processing (NLP) for performing multiple tasks including English Language Teaching (ELT). An effective strategy to favor the learning process uses interactive devices to engage learners in their self-learning process. In this work, we present a working prototype of a humanoid robotic system to assist English language self-learners through text generation using Long Short Term Memory (LSTM) Neural Networks. The learners interact with the system using a Graphic User Interface that generates text according to the English level of the user. The experimentation was conducted using English learners and the results were measured accordingly to International English Language Testing System (IELTS) rubric. Preliminary results show an increment in the Grammatical Range of learners who interacted with the system.

Keywords: Robotic systems, natural language processing, text generation, long short term memory networks.

1 Introduction

As Artificial Intelligence (AI) becomes more equipped to comprehend human communication, more institutions will adopt this technology for areas where Natural Language Processing (NLP) would make a difference. AI technology is already being used in smart home and office assistants, customer service, healthcare, and human robotics, among others.

There are multiple aspects of AI and NLP that generate the opportunity of having machines offering engaging, interactive capabilities. However, the current state of the art in NLP lacks reasoning and empathy capabilities, making complex interactions difficult. One way to exploit NLP technology engagement potential is the application of assistive technology. A particularly interesting field is the use of such systems in interactive robotics.

Humanoid robots are useful with tedious and risky errands for people, including tasks that can result in exhausting for human beings. Jobs that require a lot of concentration and feedback, like tutoring and guidance, can benefit from incorporating autonomous robotic systems to let the students interact with learning about a specific field. Robotic systems will require the capacity to understand human lexis to achieve these goals, making characteristic language handling more significant.

In the educational context, there are systems capable of teaching or assisting individuals in a self-learning process, such as Conversational Intelligent Tutoring Systems. However, they are still not optimal enough to automatically provide knowledge to help students in the learning process of a language without the need of human assistance [2]. Also, there have been interesting studies that show that interactive robotic systems are beneficial for learning [3]. The previous characteristics devise a synergy opportunity of a robotic system that incorporates an NLP component to be helpful in the self-learning process [20].

This article presents a functional prototype of a robotic system to assist the English language learning process through text-generation using Deep Neural Networks (DNN). A humanoid robot was designed and manufactured to promote learners' engagement with the assisting tool. The interaction was conducted using a Graphical User Interface (GUI) incorporated in the robot. A text-generation component was included to allow the users to interact with the system and generate language using different English levels. The experimentation was conducted with English learners and measured using the International English Language Testing System (IELTS) rubric. Preliminary results show an improvement of the subjects' current English level through regular usage of the system. However, there is a need for further and deeper experimentation to generalize the findings in this work.

The article is structured as follows: Section 2 describes the state of the art of robotic systems implemented to assist self-learning; Section 3 presents the research methodology; Section 4 describes the experimental work carried out, presenting its results in Section 5. Finally, conclusions and lines of experimentation for future work are provided in Section 6.

2 Background

A humanoid robot is a robotic system capable of presenting similar features to resemble human anatomy. These robots are usually presented and utilized as a research tool in scientific fields aimed to understand the human body structure and behavior to build. It has been proposed that robotics will be helpful in various education scenarios [3].

Previous studies indicate that robotics is providing benefits as a teaching tool in particular in the STEM fields [16], and English learning [10]. Robotic systems also provide a learning environment that seeks to improve the interdisciplinary process of learning, promoting the engagement of students in their learning activities [9, 17]. There are examples where the use of a robot for assisting the learning process is appropriate to use in language skill development as it allows a richer interaction than digital platforms [15, 17].

A significant challenge to incorporate robots as a tool to assist the self-learning process of a language is to design an engaging experience tightly related to the language the learner is using. NLP is particularly well suited to close this gap. NLP has evolved from simple classification methods like logistic regression to more complex language statistical methods and DNN [14]. Neural Networks are the dominant paradigm in NLP and have increased the research of end-to-end systems for understating human language, leading to complex applications as conversational chatbots [21].

The current and approachable theory of already-existing NLP models makes extensive use of transformers, which are topologies that use an encoder-decoder architecture incorporating an attention mechanism [26]. Many state of the art results make use of this architecture training with vast amounts of information. Models like BERT [7], T5 [22] and GPT-3 [6] are examples of big transformers delivering state-of-the-art results for various NLP tasks. Nevertheless, the field of NLP is still underdeveloped in terms of using low data quantities to perform fine-tuning in big transformers models.

One way to deal with low quantity data for NLP tasks is using RNNs. These models are effective for predicting sequence analysis tasks [12], as they store the information for the current feature based on previous information, including within the model forecasting and conditioned output capabilities [19].

Recurrent architectures learn the relative importance of different parts of the sequence; nevertheless, transformers substitute recurrent mechanisms with attention mechanisms [26], which allows the capture of longer size dependencies while reinforcing training.

There exist studies that favor traditional models like Conditional Random Fields (CRF) and LSTM networks over big transformers models in settings where the amount of data is not enough to perform fine-tuning, or the language specificity makes generalization difficult [13,23]. Additionally, LSTM runs faster, making it well suited for real-time systems interaction [4].

Language models (LM) also have been used for text-generation either using large transformers [25] or LSTM like in [5, 18]. In this research, an LM is generated using an LSTM trained on a specific dataset, and it is used to predict the succeeding word. The predicted output word is then appended with the existing input words and given as new input. This process is continuously repeated by shifting the window to generate text.

In the presented work, a humanoid robotic system was designed and manufactured to help engage in the self-learning process of English language students. A text-generation module to expose users to a variety of vocabulary and sentences was developed, thorough the experimentation, selection, and fine-tuning of LSTM models, transformers, and encoder-decoder architectures. The best model is selected to perform text-generation using a lower seed-text as shown in [24].

3 Methodology

This section presents the tools, methodologies, and development approaches used for corpus creation, text-generation module training, humanoid robotic system design, and the system integration to allow students to interact with it.

3.1 Corpus Creation

The dataset consisted on different English sentences divided into three categories: basic, intermediate, and advanced. A human expert IELTS evaluator assisted in the creation of sentences with different levels of English proficiency, considering variation in grammatical range and lexical resources according to each level.

The corpus is structured in sentences, divided by punctuation signs that are further cleaned and omitted to individual process words in the text-generation model. It contains 4,785 sentences and 150,000 words.

3.2 Text Generation Module

Most advanced models for text-generation make use of deep learning models, including LSTM networks and transformer architectures [8]. Different DNN models were trained using the dataset described in the previous section to develop the text-generation component. The researched models were: Simple LSTM model, BERT fine-tuned model, Encoder-Decoder LSTM model, Bidirectional LSTM model.

To process the text, the input sentences were tokenized and passed through the input layer of each model, then to an embedding layer, and subsequently fed to the RNN substructure that processes the tokens. Finally a softmax layer is used to predict the probability of the next word. The general architecture of the networks are depicted in the figure 1

Each model was implemented using the Keras framework and trained using the same dataset split with 80% for training and 20% for testing. Also, at training time, a development set proportion of 10% was used for Keras to compute validation loss and accuracy.

After experimenting with the mentioned models, the model with the best performance accuracy is selected and fine-tuned to perform the text-generation.

3.3 Robotic System Design

The methodology used to design the humanoid robotic system consisted of three main phases: requirement definition, specification, and design. In the requirement definition phase, an analysis of the functionality requirements of the robot

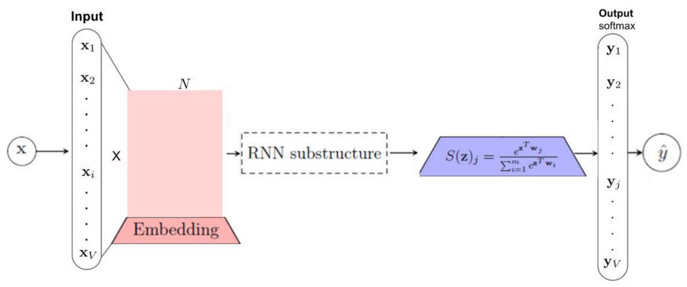

Input
x_1
x_2
x_i
x_V
x
X
N
Embedding
RNN substructure
$S(\mathbf{z})_j = \frac{e^{\mathbf{z}^T \mathbf{w}_j}}{\sum_{i=1}^{m} e^{\mathbf{z}^T \mathbf{w}_i}}$
Output
softmax
y_1
y_2
y_j
y_V
$\hat{y}$

Fig. 1. General architecture of the text-generation network.

was made, and the functional structures were defined. Then, through the specification stage, the robot and general guidelines for the project were carried on. In the design stage, specifications and guidelines were measured quantitatively, including the kinematics analysis and the definition of mechanical structures.

To favor student engagement with the robot, it was decided to use an anthropomorphic system bearing kinematics considerations. Regardless, the presented robotic system does not attempt to include mechanical components; the mechanical design was made to adopt mechanical actuators further to let the system move and increase interaction with users. The parameters that represent kinematics configuration in general terms were based on Denavitt Hardenberg [11] motion equations.

After the design stage was done, the system was drawn using the 3D drawing software fusion360. The manufacturing stage consists on printing and assembling a 3D sketch of the entire robotic system with the appropriate parameters obtained from the previous analysis.

3.4 System implementation

The implementation includes an embedded system that captures the user's speech and uses Google's Text to Speech (TTS) web service to get the transcription of the user utterance. The embedded system sends the transcription to a web service implemented in Flask to consume the best text-generation model found in the experimentation. The implemented service uses the TTS transcription as a seed to predict the following text using a fixed number of 5 words. After the model predicts the text, the Flask server sends the predicted text to the embedded system using a webhook. The embedded system uses Google's Speech to Text (STT) service to generate an audio file with the predicted text and play it using a speaker. The system is attached to the robot's body, and the user initiates the interaction. Alternatively, a Graphic User Interface (GUI) was implemented using the Gradio library [1], which can consume the service using a tablet incorporated into the robot. The GUI was intended to include users with speech or hearing disabilities. The communication architecture is depicted in the figure 2.

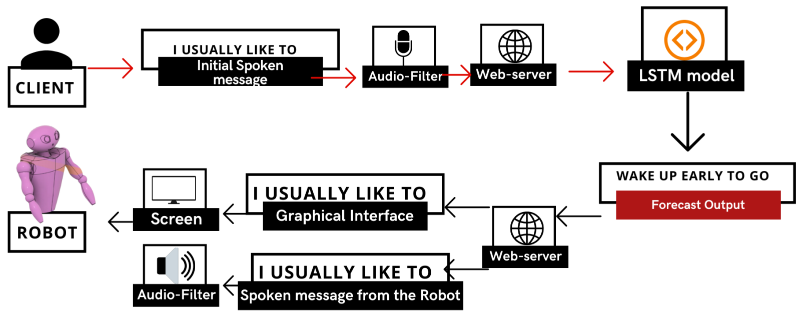


Fig. 2. System communication architecture.

4 Experimentation

This section shows the methods used for the text-generation module training, the manufacturing process of the robotic system, and the experimental process to measure system's effectivity to assist self-learning process for English students. The implemented mechanisms are illustrated, as described in section 3 of the document.

4.1 Corpus Data

The corpus consisted of sentences with 3 different English levels: elemental (IELTS accuracy level 1-2.5), pre-intermediate (IELTS accuracy level 3-4.5), and upper intermediate (IETLS accuracy level 6+). Each set contained different sequence-to-sequence compound-complex sentences. This was recommended by the IELTS evaluator to optimize three specific levels of English to tackle fluency levels in different scenarios. The corpus included 171,461 tokens, 150,356 words, and 4,785 sentences.

4.2 Text Generation Module

The different models were trained using the corpus described in section 4.1 divided into random partitions for training, validation, and test. Four different models were trained: Simple LSTM model, BERT fine-tuned model, Encoder-Decoder LSTM model, and a Bidirectional LSTM model. Each model was trained for 20 epochs, and the validation metrics were reported using the validation set. Different models were iterated using dropout regularization (*dropout*) with different probability parameters. Once the best model was obtained in the validation set, it was evaluated in the test data to report the metrics presented in section 5.1. The models were implemented using Tensorflow 2.0 and Keras on a Debian GNU/Linux 10 (buster) x86_64 operating system, supplied with an 11 GB Nvidia GTX 1080 TI GPU.

After the first experiments were conducted the best performance model found was the Bidirectional LSTM measured in terms of accuracy and validation. Once the best model was found further experimentation was done using a grid search strategy to find the best hyper-parameters of the model resulting in the following topology: LSTM layer (100 units), Dropout Layer (0.6 drop rate), LSTM layer (100 units), Dense layer (100 units, ReLU activation), Dense layer (125 units, softmax activation).

The best parameters found were the following: Embedding vocabulary-size: 70, dropout layer: 0.6, activation function: softmax, trainable parameters: 180,275, loss function: categorical cross entropy, batch size: 150.

4.3 Robotic System Manufacturing

The whole manufacturing design was approached under engineering methods to allow time-optimization and cost reductions to be considered. The process involves the following stages: Material Printing (Through a 3-D printing machine, segments from the material were printed to further treating and assembly), Material purification (Through chemical components, the segments of materials are purified through a specific epoxy designed to purify the material extracting impurities while adding brightness, Assembly of materials. (Through engineering glue, segments are assembled properly).

Each of the previous stages was divided in three segments: head-manufacturing segment, arm-manufacturing segment, body-manufacturing segment respecting each of the previously presented stages. Final configurations of the robot using the tablet and embedded system are presented in figure 3

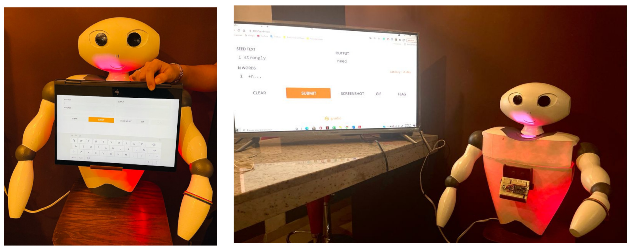


Fig. 3. Robot configurations using embedded system and tablet.

4.4 System Evaluation

To evaluate the system's effectiveness to help learners, they were evaluated using an IELTS rubric before interacting with the system. After that, the learners were exposed to interact with the system for 5 days and a new evaluation using the same rubric was made to asses the performance of the students. The evaluation was conducted with three subjects, one for each English level in the corpus.

5 Results

This section shows the results obtained from the experimentation described in section 4. The improvement of the subjects is analyzed from 250 recorded minutes of training with the system by each subject, including quantitative and qualitative evaluation from IELTS instructors. The system's performance was measured to determine the progress of the subjects.

5.1 LSTM Text-forecast Model with Encoded-decoded Attention Mechanism

Four different models were considered and evaluated to obtain the one with the best performance. Table 5.1 shows the accuracy obtained with the four different models when evaluated with the test dataset.

Model Type	Accuracy
Simple LSTM	80%
BERT fine tuned	80%
Encoder-Decoder LSTM	89%
Bidirectional LSTM	95%

Table 1. Model accuracy results.

The most suitable model that provided results to be used on experimental subjects was the Bidirectional LSTM model. Figure 4 shows the training accuracy and loss for the 20 epochs of training of the Bidirectional LSTM model.

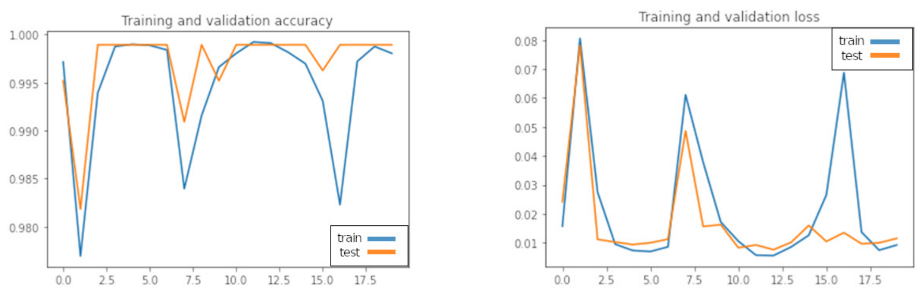


Fig. 4. Accuracy and loss validation for Bidirectional LSTM model.

5.2 Fluency Improvement in Subjects

This section presents the outcome for the fluency analysis in each of the three experimental subjects after 250 minutes of interaction (50 minutes per day for five consecutive days) with the robotic system.

The grammatical range and accuracy and marked by using a determined number of grammatical structures (6 types) in a percentage rate of accuracy and error-mistake (1-100%). The assigned instructors included the number of grammatical sentence usage in terms of accuracy percentage.

After elementary training, an increase in grammatical range and accuracy, lexical resources, and fluency is observed, while pronunciation and language-idiomatic terminology does not show improvement. From the pre-intermediate level training, a sustained increase overall dimensions was observed, except for pronunciation. The upper-intermediate level attempted to evaluate fully understanding of complex ideas generated from the advanced corpus previously trained. The idea is to oversee a different set of more compound-complex sentences generated by the robotic system. The results before and after the training are showed in figure 5.

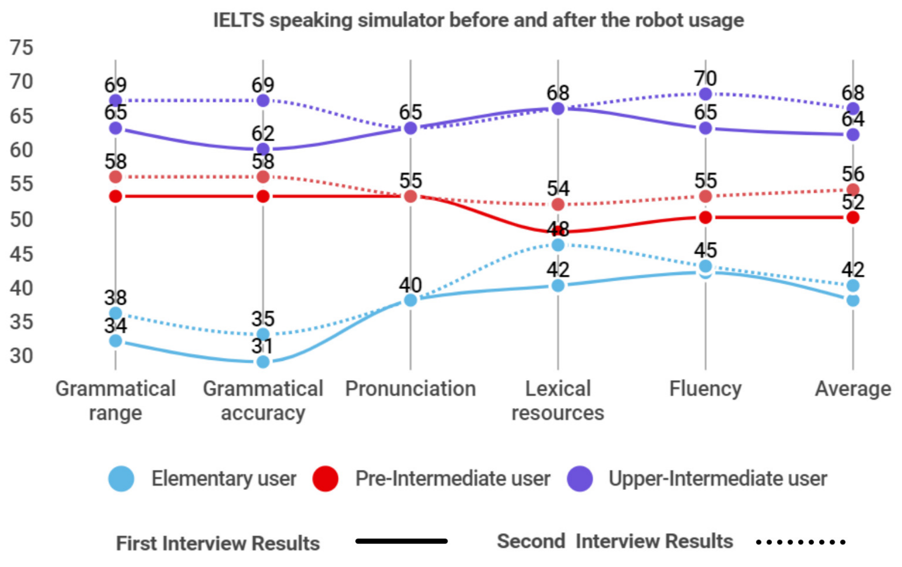
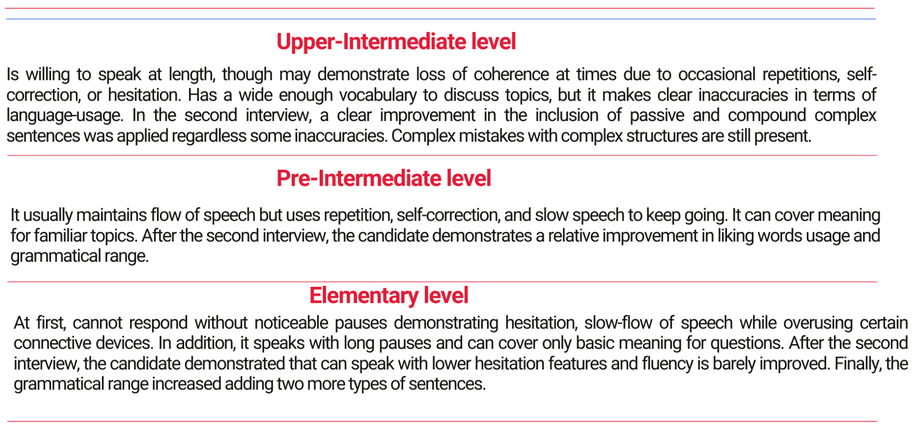


Fig. 5. IELTS metrics comparison before and after training.

5.3 Qualitative Results

The qualitative data obtained in this section was collected from IELTS instructors who evaluated and listened a set of questions from one specific context of coherence for each subject, to determine a mark in grammatical range and accuracy based on IELTS rubric. Finally, instructors who listened the same ideas in the second interview attached written feedback shown in the figure 6.



Upper-Intermediate level

Is willing to speak at length, though may demonstrate loss of coherence at times due to occasional repetitions, self-correction, or hesitation. Has a wide enough vocabulary to discuss topics, but it makes clear inaccuracies in terms of language-usage. In the second interview, a clear improvement in the inclusion of passive and compound complex sentences was applied regardless some inaccuracies. Complex mistakes with complex structures are still present.

Pre-Intermediate level

It usually maintains flow of speech but uses repetition, self-correction, and slow speech to keep going. It can cover meaning for familiar topics. After the second interview, the candidate demonstrates a relative improvement in liking words usage and grammatical range.

Elementary level

At first, cannot respond without noticeable pauses demonstrating hesitation, slow-flow of speech while overusing certain connective devices. In addition, it speaks with long pauses and can cover only basic meaning for questions. After the second interview, the candidate demonstrated that can speak with lower hesitation features and fluency is barely improved. Finally, the grammatical range increased adding two more types of sentences.

Fig. 6. Qualitative feedback from IELTS instructor after training.

The results express that the instructors perceived noticeable enhancement in the English abilities of the subjects after the interaction with the robot.

6 Conclusions and Future Work

This work presented the design, development, and manufacturing of a humanoid robotic system to assist English language students in a self-learning process. The robotic system was developed using a three-phase methodology (requirement analysis, specification, and design) which yields good results since the system is articulated and ready to add further interaction using actuators.

Various models were tested to implement the text-generation module; a particularly interesting observation is related to the relatively poor results (80% accuracy) obtained when using a fine-tuned BERT model. This occurs due to the relatively small amount of data used to perform the fine-tuning; in this regard, the bidirectional LSTM model performs better, achieving a 95% of accuracy in the test set.

The bidirectional LSTM text-generation model was useful to predict text using a seed given by the user; nevertheless, noticeable irregular fluctuations were reported on the validation accuracy and loss chart, which can be produced from irregularities in the English levels used within the corpus.

The experimentation was carried on with three English students of elementary, pre-intermediate, and upper-intermediate English levels, and their progress was measured according to the IELTS rubric. After 250 hours of training, comparative results demonstrated an average improvement of 4% in their grammatical range, 4% in grammatical accuracy, and 3.33 % in their fluency. No difference was observed in their pronunciation abilities.

Quantitative and qualitative data obtained from the experimentation depicted a positive result on how a robotic system can provide aid while tackling a specific ability from a foreign language. In this case, the main improvements were reported in terms of fluency and grammatical range skills. Qualitative results show a favorable opinion both from IELTS instructors and students. In general, they perceived the system as a beneficial tool for the progress of the students.

The experimental results were limited by time constraints and the reduced number of subjects, so further research is needed to generalize the observed results.

The future work regarding this project includes: robust experimentation using more subjects and more structured training sessions, revision of other learning techniques and the overall effect on the English language improvement, experiment with variations on the composition of the corpus to measure its impact in the learning process. Also, interesting research can be conducted regarding pronunciation improvement using a more controlled spoken interaction with the users and the effect of dynamic movement adding actuators to the robot and measuring the impact in the self-learning process.

References

1. Abid, A., Abdalla, A., Abid, A., Khan, D., Alfozan, A., Zou, J.: Gradio: Hassle-free sharing and testing of ml models in the wild. arXiv preprint arXiv:1906.02569 (2019)
2. Akkila, A., Almasri, A., Ahmed, A., Al-Masri, N., Abu, Y., Mahmoud, A., Zaqout, I., Abu-Naser, S.: Survey of intelligent tutoring systems up to the end of 2017. International Journal of Academic Information Systems Research vol. 3, 36–49
3. Anwar, S., Bascou, N., Menekse, M.: A systematic review of studies on educational robotics. Journal of Pre-College Engineering Education Research (J-PEER) 9(2) (2019)
4. Breuel, T.M.: Benchmarking of LSTM networks. CoRR abs/1508.02774 (2015), http://arxiv.org/abs/1508.02774
5. Cho, K., Merrienboer, B.v., Gulcehre, C., Bougares, F., Schwenk, H., Bengio, Y.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In: EMNLP (2014), https://hal.archives-ouvertes.fr/hal-01433235
6. Dale, R.: Gpt-3: What's it good for? Natural Language Engineering 27(1), 113–118 (2021)
7. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the NAACL-HLT Volume 1. pp. 4171–4186. Association for Computational Linguistics (2019), https://doi.org/10.18653/v1/n19-1423
8. Ezen-Can, A.: A comparison of lstm and bert for small corpus. ArXiv abs/2009.05451 (2020), http://arxiv.org/abs/2009.05451

9. Grubbs, M.: Robotics intrigue middle school students and build stem skills. Technology and Engineering Teacher 72(6), 12–16 (2013)
10. Han, J., Kim, D.: r-learning services for elementary school students with a teaching assistant robot. In: 2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI). pp. 255–256 (2009)
11. Hayat, A.A., Chittawadigi, R.G., Udai, A.D., Saha, S.K.: Identification of denavit-hartenberg parameters of an industrial robot. In: Proceedings of Conference on Advances In Robotics. p. 1–6. AIR '13, Association for Computing Machinery, New York, NY, USA (2013)
12. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF Models for Sequence Tagging (2015), Retrieved from http://arxiv.org/abs/1508.01991
13. Joselson, N., Hallén, R.: Emotion classification with natural language processing (comparing bert and bi-directional lstm models for use with twitter conversations) (2019), student Paper
14. Khurana, D., Koli, A., Khatter, K., Singh, S.: Natural language processing: State of the art, current trends and challenges. CoRR abs/1708.05148 (2017), http://arxiv.org/abs/1708.05148
15. Lai Poh, E.T., Albert, C., Pei-Wen, T., I-Ming, C., Song Huat, Y.: A review on the use of robots in education and young children. Journal of Educational Technology Society 19(2), 148–163 (2016), http://www.jstor.org/stable/jeductechsoci.19.2.148
16. Melchior, A., Cohen, F., Cutter, T., Leavitt, T.: More than robots: An evaluation of the first robotics competition participant and institutional impacts. In: Brandeis University Center for Youth and Communities Heller School for Social Policy and Management. (2005)
17. Menekse, M., Higashi, R., Schunn, C.D., Baehr, E.: The role of robotics teams collaboration quality on team performance in a robotics tournament. Journal of Engineering Education 106(4), 564–584 (2017)
18. Merity, S., Keskar, N.S., Socher, R.: Regularizing and optimizing LSTM language models. 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings (2018)
19. Muzaffar, S., Afshari, A.: Short-Term Load Forecasts Using LSTM Networks. Energy Procedia 158, 2922–2927 (2019), https://www.sciencedirect.com/science/article/pii/S1876610219310008
20. Newton, D.P., Newton, L.D.: Humanoid robots as teachers and a proposed code of practice. Frontiers in Education 4, 125 (2019), https://www.frontiersin.org/article/10.3389/feduc.2019.00125
21. Peters, F.: Master thesis: Design and implementation of a chatbot in the context of customer support (2018), http://hdl.handle.net/2268.2/4625
22. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. CoRR abs/1910.10683 (2019), http://arxiv.org/abs/1910.10683
23. Rosvall, E.: Comparison of sequence classification techniques with bert for named entity recognition. Electrical Engineering and computer science faculty (2019)
24. Santhanam, S.: Context based text-generation using LSTM networks. CoRR abs/2005.00048 (2020), https://arxiv.org/abs/2005.00048
25. Topal, M.O., Bas, A., van Heerden, I.: Exploring transformers in natural language generation: Gpt, bert, and xlnet. CoRR abs/2102.08036 (2021), https://arxiv.org/abs/2102.08036
26. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. CoRR abs/1706.03762 (2017), http://arxiv.org/abs/1706.03762

ISSN 1870-4069

Hyperparameter Optimization for Transfer Learning in Gastrointestinal Image Classification Using an Evolutionary Algorithm: Proof of Concept

Rogelio García-Aguirre, Luis Torres-Treviño, Alberto Gonzáles-González

Universidad Autónoma de Nuevo León,
Facultad de Ingeniería Mecánica y Eléctrica,
San Nicolás de los Garza, N.L.,
Mexico

rogelio.garciag@uanl.edu.mx

Abstract. Endoscopy is the foundation for the diagnosis and treatment of several gastrointestinal ailments. However, it is an operator-dependent procedure. The quality of assessment of endoscopy images relies on the physician's experience, ability, and conditions. A system capable of automatically evaluate endoscopy images and classify different gastrointestinal findings within them is an alternative to enhance the diagnosis quality. Convolutional neural networks (CNN) show great promise to this end. Research on this topic focuses on the generation of new complex architectures. Several publications have stated their concerns about the capability of these state-of-the-art schemes to be deployed in a clinical setting. Mainly due to the uncertainty to employ them in real-time because of the computing power these algorithms need. In this study, we performed hyperparameter optimization during the transfer learning and fine-tuning process using off-the-shelf CNN (more likely to operate in real-time) for the image classification task. To this purpose, we use an evolutionary algorithm. We provide preliminary results to this method, proving that this approach may reach classification performance competitive with the novel deep learning structures while maintaining low complexity in the architecture.

Keywords: Hyperparameter optimization, evolutionary algorithm, medical images, gastrointestinal tract, deep learning.

1 Introduction

In recent years, research on automatic medical image classification has gained significant importance. The implementation of Artificial Intelligence (AI) in medicine has been successful in image-intensive specialties, such as radiology, pathology, ophthalmology, and cardiology [22]. Several publications have reported the current state and expectations of such tools in the area of gastroenterology, in particular for endoscopy [1,3,6,8,23]. Endoscopy is the foundation for the diagnosis and treatment of diseases of the gastrointestinal (GI) tract. This procedure is operator-dependent, which generates substantial interobserver variation in the detection and assessment of GI findings [19]. So, the detection of GI lesions essentially relies on the expertise of the physician [28].

An automated system capable of classifying different GI findings would help to reduce the variation in endoscopists' performance [19]. The development of computer-aided diagnosis (CAD) systems with this purpose is currently an open challenge since the feasibility, effectiveness, and safety of CAD for upper gastrointestinal endoscopy in clinical practice remain unknown [23]. There are different approaches for classifying endoscopic images. Deep learning (DL) techniques usually outperform strategies that use hand-crafted features [23,28]. Consequently, convolutional neural networks (CNN) are the most employed method nowadays.

The majority of research focuses on proposing new architectures or combining existing frameworks to enhance classification/detection performance. However, the systems must allow operating in real-time to achieve the final goal, which is assistance in real-time during endoscopy [1,8,16,23]. Numerous state-of-the-art architectures run too slow to be implemented in a clinical setting [19]. Therefore, optimizing the performance of off-the-shelf architectures (that may allow real-time operation) is a possible solution to this.

We utilize less complex deep architectures to classify endoscopic images of the KVASIR dataset [24]. We implement transfer learning in different off-the-shelf CNN. Formerly, we aim to optimize hyperparameters to improve the classification performance of the algorithms. Borgli et al. [5] presented a similar scheme, yet, they carried out Bayesian optimization. In contrast, we propose to use an evolutionary algorithm for this purpose.

2 Related Work

Several studies are using AI to analyze endoscopic images. A great deal of these focuses on a specific GI finding, such as polyp detection and segmentation (e.g., [26]), gastric cancer detection and diagnosis (e.g., [20]), diagnosis and detection of Helicobacter Pylori infection (e.g., [30]), among others. The publication in 2017 of the KVASIR dataset [24], consisting of 8000 images of different GI findings in images of upper endoscopy, made possible the development of a new generation of algorithms for endoscopic image classification. These studies aim to achieve a general classification of the different GI findings that can appear during endoscopy instead of concentrating on a particular suffering or symptom.

2.1 Dataset

Machine learning (ML) and DL schemes need datasets to be developed, validated, tested, and compared. With this in mind, Pogolerov et al. [24] created and published the KVASIR dataset. The original version of this dataset consists of 8000 images from inside the GI tract. This dataset contains anatomical landmarks, pathological findings, procedures, and normal findings. In concrete, the images belong to one of the following classes: z-line, pylorus, cecum, esophagitis, polyps, ulcerative colitis, dyed lifted polyps, dyed resection margin, normal colon mucosa, and stool. Each class has 1000 images of it. Hence, the dataset is balanced.

2.2 Evaluation Metrics

There exist standard evaluation metrics used to assess classification algorithms. We take the performance measures from [16]. Since it is the most complete and recent paper describing and comparing the performance of different methods for automatic endoscopic image classification. The evaluation metrics are as follows: recall (REC), specificity (SPEC), accuracy (ACC), precision (PREC), Matthews correlation coefficient (MCC), and F_1 value (F1):

$$REC = \frac{TP}{TP + FN}, \tag{1}$$

$$SPEC = \frac{TN}{TN + FP}, \tag{2}$$

$$PREC = \frac{TP}{TP + FP}, \tag{3}$$

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}, \tag{4}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FN)(TN + FP)(TP + FP)(TN + FN)}}, \tag{5}$$

$$F1 = 2 \times \frac{PREC \times REC}{PREC + REC}. \tag{6}$$

In the above, *TP*, *TN*, *FP* and, *FN* stand for *true positive*, *true negative*, *false positive* and, *false negative*, respectively.

2.3 Classification algorithms

To compare and measure the performance of the proposed method, in this investigation, we only consider studies that use one version of the KVASIR dataset for training, validation, and testing. Table 1 shows the studies that use some version of the KVASIR dataset and have the best classification performance.

During the last years, different paradigms for the classification of endoscopic images came to be tested. CNN architectures are the current most effective approach. Regarding classification performance, hybrid architectures have gained notice. Chang et al. [7] denoted the importance of applying an adequate data augmentation technique. In [7], they developed an algorithm to automatically

Table 1. Studies with the best evaluation metrics for endoscopic image classification using some version of the KVASIR dataset. Metrics as presented in [16].

Autor	Architecture	MCC	ACC	F1
Chang et al., 2019 [7]	ResNet34 [10] + SE-ReNext [14] + Attention-inception-v3 [27]	0.9520	0.9946	0.9569
Harzing et al., 2019 [9]	MobileNetV2 [25]	0.9490	0.9936	0.9105
Luo et al., 2019 [21]	10 CNN + LightGBM [17]	0.9480	0.9941	0.9533
Hoang et al., 2019 [13]	ResNet-101 [10] + Faster R-CNN	0.9406	0.9933	0.9464
Hoang et al., 2018 [12]	ResNet-101 [10] + Faster R-CNN	0.9398	0.9932	0.9342
Thambawita et al., 2018 [29]	ResNet-152 [10] + DenseNet-161 [15] + MP	0.9397	0.9932	0.9297
Hicks et al., 2018 [11]	DenseNet-169 [15]	0.9325	0.9924	0.9236

select the data augmentation technique based on the F1 value of a rapid training in 20 groups of randomly selected test samples. For the classification of endoscopic images, they developed a CNN architecture consisting of a reduced version of residual neural network (ResNet34 [10] combined with SE-ReNext [14] and Attention -inception-v3 [27]. The addition of the attention blocks aimed for these to learn the differences between classes. This study introduced multi-epoch fusion, which consists of using the average of the weights of the last 5 training epochs to improve the model generalization and avoid parametric overfitting.

Chang et al. [7] designed their architecture to carry out multi-label classification. So, they realized a threshold selection of belongings to each label. They tested different threshold combinations for each label and selected the one that had the best performance. This work was the best evaluated for classification task in the Biomedia ACM MM Grand Challenge 2019 [16], where they reached an MCC of 0.9520.

Harzig et al. [9], used 2 CNNs for image classification, and although they used data augmentation techniques, their results were affected by the imbalance in the training samples [9]. In this study, the authors focused on making a fast classification, and not only accurate. Consequently, they used smaller CNNs that can run even on mobile devices. With MobileNetV2 [25] they achieved an MCC of 0.95974 in the KVASIR database [9] with an inference time that suggests that this algorithm could be implemented in real-time.

An interesting idea to combine different CNNs in a single model is that proposed by Luo et al. [21], they individually tested some state-of-the-art CNNs and selected the 10 with the best classification results for the KVASIR database. Subsequently, they trained 10 sub-models for each of the selected CNNs using cross-validation with the training data. Then, they used the output as a vector of probabilities of membership to each class of the trained submodels as a set of characteristics to train ML systems for classification. Their best MCC was

0.948035, and they obtained it with a LightGBM [17] classifier.

Hoang et al. [13] proposed and applied a data augmentation technique, which consists of cropping the region of interest for classification and adding this region of the image to others in the same database. These authors implemented a residual neural network in conjunction with a Faster R-CNN. The goal of using these 2 neural networks working together is that ResNet CNN carries out the classification work and, the detection network serves to reiterate the class. With this methodology, the authors achieved an MCC of up to 0.9406 for classification in the Biomedia ACM MM Grand Challenge 2019 test database [16,**?**].

In [29], Thambawita et al. studied different pre-trained models and combinations of these. They concluded that the combination of ResNet-152 and DenseNet-161 to extract image features, with a multi-layer perceptron for the classification lead to the best performance. With this approach, they got an MCC of 0.9397 in the 2018 Medico Classification task.

Hicks et al. [11] conjectured that pre-training the models with a medical dataset could enhance the models' performance. However, they discovered that vast and diverse datasets were better to pre-train, even if they were not similar to the final dataset. These authors reached an MCC of 0.9325 in the 2018 Medico Classification task using DenseNet-169.

2.4 Hyperparameter Optimization

In the previous subsection, we presented several studies concerning endoscopic image classification. In concrete, all of these works use a version of the KVASIR dataset. It is important to denote that every one of the presented studies used transfer learning to adapt the employed model to the target domain. During transfer learning, there are some hyperparameters to tune. These are capable of enhancing or worsen the model's performance. Despite the importance of the hyperparameter, these are usually manually tuned.

The only precedent that currently exists in the literature regarding the automatic tuning of hyperparameters to optimize the classification performance of endoscopic images is the research of Borgli et al. [5]. In this study, they used a Bayesian optimization approach achieving an improvement of up to 10% in terms of accuracy with other works that used the same CNNs for classification in the KVASIR database by adjusting the hyperparameters manually.

Borgli et al. [5] considered 4 hyperparameters: The pre-trained model, the gradient descent optimizing function, the learning rate and, the delimiting layer. The pre-trained model refers to the kind of architecture that is used to classify the images. They used KERAS to train the models, so the architectures and gradient descent optimizing function that they used during the optimization process were the ones available in this API. The architectures prospects were: Xception, VGG16, VGG19, ResNet50, InceptionV3, InceptionRes- NetV2, DenseNet121, DenseNet169, and DenseNet201. The gradient descent optimizing function prospects were SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, and Nadam. The learning rate was set in a continuous value between 1 and 10^{-4}. Finally, the delimiting layer refers to the number of layers that are trained in the model. This value was set between 0 and the number of layers in the selected model.

3 Methods and Implementation

3.1 Evolutionary Algorithm

Evolutive algorithms use the paradigm of evolution proposed by Darwin, in which the fundamental law is the principle of variation and selection. This principle of changing each generation (through reproduction) is the main component of the evolutionary strategies [4]. *Evolutionary algorithms are based on the collective learning process within a population of individuals, each of which represents a search point in the space of potential solutions to a given problem.* Back, 1993 [2].

In evolutionary algorithms, several individuals explore the solution space of the environment at random points. Then, the best-evaluated individuals pass their genes (information) to the next generation. Evaluation is the procedure of assessing how well the solutions fit the established goals. The genes of the selected individuals are preserved and mixed in new individuals with recombination mechanisms. Also, it is a good practice to consider a mutation factor during this procedure. The mutation is the introduction of random information, which introduces innovation into the population [2]. Fig. 1 shows a general overview of an evolutionary algorithm.

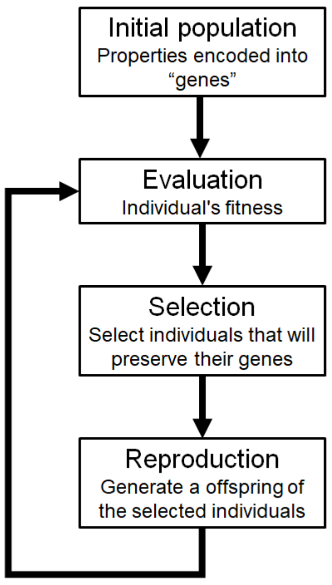

Initial population
Properties encoded into "genes"
Evaluation
Individual's fitness
Selection
Select individuals that will preserve their genes
Reproduction
Generate a offspring of the selected individuals

Fig. 1. General evolutionary algorithm.

We considered the whole population for the selection process. We use tournament selection, in which the algorithm takes n individuals randomly from the population, compares them, and selects the individual with the best evaluation. The algorithm repeats this process until it reaches the desired total number of selected individuals. All the individuals have the same chance to participate in the tournament. Nevertheless, the individuals with the highest evaluations are more likely to win the match and preserve their genes.

There are different mechanisms for the generation of new individuals. Overall, this procedure consists of the information (genes) combination of the selected individuals and a mutation factor to introduce novel information into the population. There exist different combination mechanisms, such as recombination (generate a new individual mixing up the parents' genes), raw combination (generate new genes by blending the parents' genes), etc. The combination mechanisms could be static or dynamic. The same happens with the mutation factor. Fig. 2 shows an example of the different combination techniques and the mutation procedure.

During this work, we use a static combination method without recombination. We use the arithmetic mean of the parents' genes as the combination method. We considered a mutation factor as an independent variable for every gen for every individual. Table 2 shows a pseudocode representation of the evolutionary algorithms that we use during the experiments presented in this study.

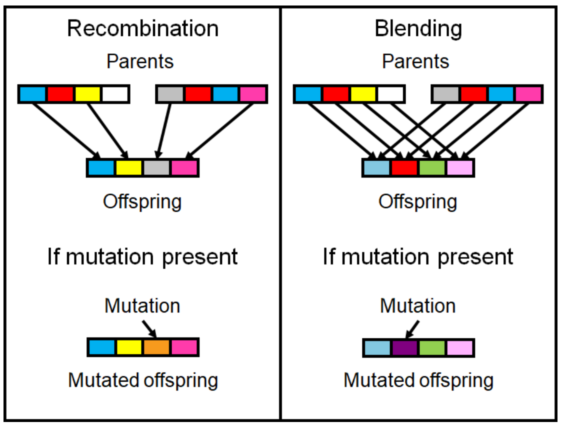


Fig. 2. Generation mechanisms of evolutive algorithms.

3.2 Data

The KVASIR dataset [24] has 8000 labeled images, 1000 for each class. We split these images into three subsets for training, validation, and testing, each with 4000, 2400, and 1600 images, respectively.

Table 2. Pseudocode representation of the evolutive algorithm.

```
1) Random generation of initial population of size $M$.
2) While condition of conclusion is not satisfied.
3)    Evaluation of the population.
4)    Selection of the $N$ individuals for the crossover.
5)    Generation of a new population of size $M$ with the crossover of the $N$
      selected individuals.
6) End.
```

We used data augmentation techniques. In every training iteration, the training images went through a transformation step, where they are randomly rotated at an angle between $1°$ and $355°$. Also, a horizontal flip, vertical flip, and brightness adjustment are applied to every image, with a 50% probability for every transformation.

3.3 Experiments Settings

The experiments were carried out using the following hardware specifications: AMD Ryzen 5 3400G CPU, one NVIDIA GeForce GTX 1660 Ti GPU, 16 GB RAM, and 476 GB system memory. All the algorithms were implemented in Python 3.8.5, using the environment Spyder 4.1.5. Pytorch 1.7.1 was used to obtain the pretrained CNNs architectures and gradient descent optimization for training, which was Adam algorithm [18].

We included four hyperparameters in the optimization algorithm during the transfer learning: kind of CNN architecture, learning rate, delimiting layer, and training epochs. The gene representing the kind of CNN architecture takes discrete values, one for each architecture available, which were: AlexNet, ResNet-18, ResNet-34, Resnet50, SqueezeNet-1.1, DenseNet-121, DenseNet-169, MobileNet-v2, ShuffleNet-v2-x0.5, and ResNext-50-32x4d. The other three genes take continuous values. For the learning rate, we established bounds between 10^{-4} and 10^{-6}. The bounds of the delimiting layer depended on the selected architecture. The lower bound represents that the last half of the layers are fine-tuned, and the upper bound represents that the training only affects the classification layers. For the training epochs gen, we set the bounds between 5 and 15.

The maximization of the validation accuracy was the optimization target for the evolutionary algorithm. During the generation process, the genes had a probability of crossover (with arithmetic mean) of 50%, except the CNN architecture, which automatically inherited the gene of the best-evaluated parent. The learning rate, delimiting layer, and training epochs genes had a mutation probability of 15%, and the CNN architecture a probability of 25%. The population during the optimization consisted of 20 individuals. The evolutionary algorithm had 5 generations in total since the experiment goal was to prove that this kind of optimization is effective for this task.

4 Results

The best-evaluated individual of the initial population (0-generation) had a validation accuracy of 0.8995, and the best-evaluated individual of the fifth generation has a validation accuracy of 0.9821. Table 3 shows the characteristics of the best-evaluated individual of each generation in detail.

Table 3. Best evaluated individuals per generation.

Gene-ration	Learning rate	Architecture	Layers pretrained	Epochs Epochs	Validation Acc	Validation MCC	Validation F1
0	8.888^{-4}	ShuffleNet-v2-x0.5	0.9341	11	0.9718	0.8709	0.8870
1	1.892^{-4}	ResNext-50-32x4d	0.6615	12	0.9786	0.9023	0.9145
2	1.892^{-4}	ResNet-18	0.9587	12	0.9769	0.8942	0.9075
3	2.209^{-4}	ResNext-50-32x4d	0.6950	11	0.9783	0.9006	0.9130
4	3.231^{-4}	MobileNet-v2	0.8870	11	0.9804	0.9103	0.9215
5	2.306^{-5}	ResNet-50	0.9101	11	0.9821	0.9183	0.9285

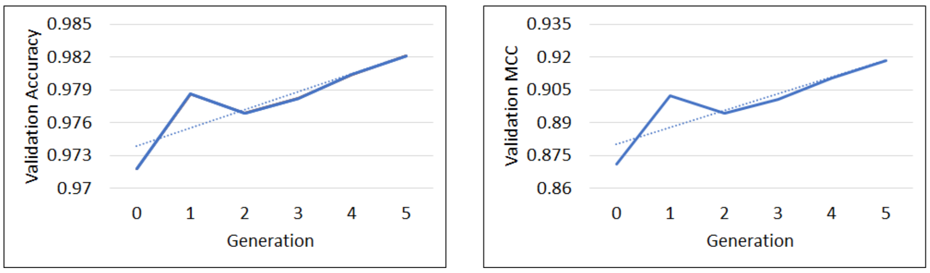


Fig. 3. Validation accuracy (left) and MCC (right) of the best-evaluated individuals per generation.

5 Conclusions

The experiment results imply that an evolutionary strategy can improve the accuracy of an endoscopy image classification algorithm. The results presented are evidence that this kind of optimization paradigm can lead to classification performance comparable to that of the best-evaluated architectures since the validation accuracy, MCC, and F1 value reached during the experiment are similar to those presented in table 1. Nevertheless, the objective of the experiment was to demonstrate that optimization of hyperparameter using a genetic algorithm is capable of improving the classification performance of off-the-shelf architectures.

The results presented are enough to prove it, considering that we got a rising of 0.0474% in the validation MCC in 5 optimization steps using only 20 individuals and basic generation mechanisms.

An inconvenience of using evolutionary strategies is that this kind of optimization algorithms usually needs several individuals. In computationally expensive tasks (such as this), it can take too long to reach the optimal solution. A possible solution to this issue is using surrogate models.

During the experiment, we only considered four genes (CNN architecture, learning rate, delimiting layer, and training epochs), and we set the bounds of the searching space based on the literature. In the future, we can extend the number of genes by including other hyperparameters, and we can refine the bounds of the searching space by carrying out a characterization of the solution space of the optimization problem.

Also, in this work, we set as optimization target the maximization of the validation accuracy. In future research, we can implement multi-objective optimization. For example, we can set as optimization target the maximization of other evaluation metrics, such as the MCC value, along with the minimization of the training time.

References

1. Alagappan, M., Brown, J., Mori, Y., Berzin, T.: Artificial intelligence in gastrointestinal endoscopy: The future is almost here. World Journal of Gastrointestinal Endoscopy 10, 239–249 (10 2018)
2. Back, T., Schwefel, H.P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation 1, 1–23 (03 1993)
3. Berzin, T., Parasa, S., Wallace, M., Gross, S., Repici, A., Sharma, P.: Position statement on priorities for artificial intelligence in gi endoscopy: a report by the asge task force. Gastrointestinal Endoscopy 92 (06 2020)
4. Beyer, H.G.: The theory of evolution strategies (01 2001)
5. Borgli, R.J., Kvale Stensland, H., Riegler, M.A., Halvorsen, P.: Automatic hyperparameter optimization for transfer learning on medical image datasets using bayesian optimization. In: 2019 13th International Symposium on Medical Information and Communication Technology (ISMICT). pp. 1–6 (2019)
6. Chahal, D., Byrne, M.: A primer on artificial intelligence and its application to endoscopy. Gastrointestinal Endoscopy 92 (05 2020)
7. Chang, Y., Huang, Z., Chen, W., Shen, Q.: Gastrointestinal tract diseases detection with deep attention neural network. In: Proceedings of the 27th ACM International Conference on Multimedia. MM '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3343031.3356061
8. Gross, S., Sharma, P., Pante, A.: Artificial intelligence in endoscopy. Gastrointestinal Endoscopy 91 (12 2019)
9. Harzig, P., Einfalt, M., Lienhart, R.: Automatic disease detection and report generation for gastrointestinal tract examination. In: Proceedings of the 27th ACM International Conference on Multimedia. MM '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3343031.3356066
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)

11. Hicks, S., Smedsrud, P., Halvorsen, P., Riegler, M.: Deep learning based disease detection using domain specific transfer learning (11 2018)
12. Hoang, T.H., Nguyen, H.D., Nguyen, T.A., Nguyen, V.T., Tran, M.T.: An application of residual network and faster - rcnn for medico: Multimedia task at mediaeval 2018. In: CEUR (ed.) MEDIAEVAL 2018, MediaEval Benchmarking Initiative for Multimedia Evaluation Workshop, 29-31 October 2018, Sophia-Antipolis, France. Sophia-Antipolis (2018), cEUR
13. Hoang, T.H., Nguyen, H.D., Nguyen, V.A., Nguyen, T.A., Nguyen, V.T., Tran, M.T.: Enhancing endoscopic image classification with symptom localization and data augmentation. In: Proceedings of the 27th ACM International Conference on Multimedia. MM '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3343031.3356073
14. Hu, J., Shen, L., Albanie, S., Sun, G., Wu, E.: Squeeze-and-excitation networks (2019)
15. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks (2018)
16. Jha, D., Ali, S., Hicks, S., Thambawita, V., Borgli, H., Smedsrud, P.H., de Lange, T., Pogorelov, K., Wang, X., Harzig, P., Tran, M.T., Meng, W., Hoang, T.H., Dias, D., Ko, T.H., Agrawal, T., Ostroukhova, O., Khan, Z., Atif Tahir, M., Liu, Y., Chang, Y., Kirkerød, M., Johansen, D., Lux, M., Johansen, H.D., Riegler, M.A., Halvorsen, P.: A comprehensive analysis of classification methods in gastrointestinal endoscopy imaging. Medical Image Analysis 70, 102007 (2021), https://www.sciencedirect.com/science/article/pii/S1361841521000530
17. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
19. de Lange, T., Halvorsen, P., Riegler, M.: Methodology to develop machine learning algorithms to improve performance in gastrointestinal endoscopy. World Journal of Gastroenterology 24 (12 2018)
20. Luo, H., Xu, G., Li, C., He, L., Luo, L., Wang, Z., Jing, B., Deng, Y., Jin, Y., Li, Y., Tan, W., He, C., Seeruttun, S., Wu, Q., Huang, J., Huang, D.w., Chen, B., Lin, S.b., Xu, R.h.: Real-time artificial intelligence for detection of upper gastrointestinal cancer by endoscopy: a multicentre, case-control, diagnostic study. The Lancet Oncology 20 (10 2019)
21. Luo, Z., Wang, X., Xu, Z., Li, X., Li, J.: Adaptive ensemble: Solution to the biomedia acm mm grandchallenge 2019. In: Proceedings of the 27th ACM International Conference on Multimedia. MM '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3343031.3356078
22. Maddox, T., Rumsfeld, J., Payne, P.: Questions for artificial intelligence in health care. JAMA 321 (12 2018)
23. Mori, Y., Kudo, s.e., Mohmed, H., Misawa, M., Ogata, N., Itoh, H., Oda, M., Mori, K.: Artificial intelligence and upper gastrointestinal endoscopy: Current status and future perspective. Digestive Endoscopy 31 (12 2018)
24. Pogorelov, K., Randel, K.R., Griwodz, C., Eskeland, S.L., de Lange, T., Johansen, D., Spampinato, C., Dang-Nguyen, D.T., Lux, M., Schmidt, P.T., Riegler, M.,

Halvorsen, P.: Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection. In: Proceedings of the 8th ACM on Multimedia Systems Conference. pp. 164–169. MMSys'17, ACM, New York, NY, USA (2017), http://doi.acm.org/10.1145/3083187.3083212

25. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
26. Shin, Y., Balasingham, I.: Automatic polyp frame screening using patch based combined feature and dictionary learning. Computerized Medical Imaging and Graphics 69, 33–42 (08 2018)
27. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision (2015)
28. Thambawita, V., Jha, D., Hammer, H.L., Johansen, H.D., Johansen, D., Halvorsen, P., Riegler, M.A.: An extensive study on cross-dataset bias and evaluation metrics interpretation for machine learning applied to gastrointestinal tract abnormality classification. ACM Trans. Comput. Healthcare 1(3) (6 2020), https://doi.org/10.1145/3386295
29. Thambawita, V., Jha, D., Riegler, M., Halvorsen, P., Hammer, H.L., Johansen, H.D., Johansen, D.: The medico-task 2018: Disease detection in the gastrointestinal tract using global features and deep learning (2018)
30. Yasuda, T., Hiroyasu, T., Hiwa, S., Okada, Y., Hayashi, S., Nakahata, Y., Yasuda, Y., Omatsu, T., Obora, A., Kojima, T., Ichikawa, H., Yagi, N.: Potential of automatic diagnosis system with linked color imaging for diagnosis of Helicobacter pylori infection. Digestive Endoscopy 32(3), 373–381 (2020)

ISSN 1870-4069

DNN Speaker Tracking with Embeddings

Carlos Rodrigo Castillo-Sanchez[1], Leibny Paola Garcia-Perera[2], Anabel Martin-Gonzalez[1]

[1] Universidad Autónoma de Yucatán, Computational Learning and Imaging Research, Mexico

[2] Johns Hopkins University, Center for Language and Speech Processing, USA

carloscastillomvc@gmail.com, amarting@correo.uady.mx, leibny@gmail.com

Abstract. Speaker tracking is the task of finding hypothesized speakers in a multi-speaker conversation. In this paper, we propose a novel way to perform online speaker tracking based on neural networks. We designed an architecture that mimics the probabilistic linear discriminant analysis (PLDA) algorithm and outputs the most likely regions uttered by a predefined target speaker. This output can be used for downstream tasks such as diarization or tracking, as analyzed in this paper. For sake of generalization, we used two standard public datasets that were carefully modified to create two-speaker subsets with additional overlapping speech and non-target speakers. Relative improvements of 40% and 20% in minDCF for CALLHOME and DIHARD II single-channel show promising performance.

Keywords: Speaker tracking, speaker diarization, speaker verification, x-vector, i-vector.

1 Introduction

Speaker tracking can be considered as the process of identifying all regions uttered by a hypothesized speaker in a multi-speaker recording [1]. Similarly to speaker diarization, which answers the question *"who spoke when?"*, speaker tracking searches for those regions, but assigns speaker identities to them. Finding where a given speaker is intervening in a conversation is an essential preprocessing step for many multi-speaker applications, where speech data from previous enrollments may be available, such as virtual assistants, meetings and broadcast news transcription and indexing [8].

As shown in [6], diarization and tracking are two methods closely related. Although tracking would benefit from the diarization, in this research, we explored the possibility of including a neural network as a robust classifier that

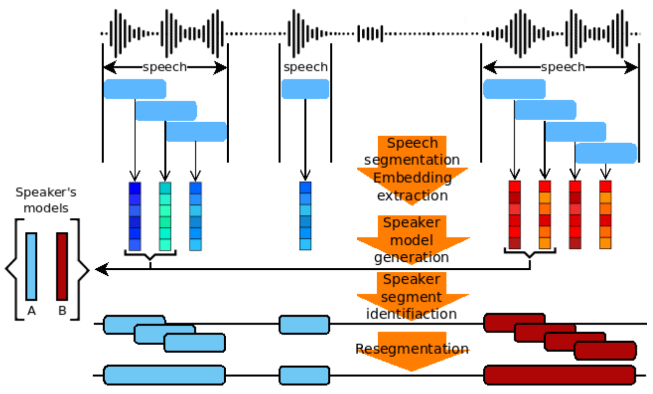


Fig. 1. Pipeline of the proposed speaker tracking system.

can operate similarly to the probabilistic linear discriminant analysis (PLDA), with the goal of naturally providing results for diarization and tracking.

Since there are just a few studies on speaker tracking [1,8,24], we use diarization as the main background and inspiration of this work. Most of the standard speaker diarization systems focus on offline clustering as it uses all the contextual information to label the speech regions. Examples of such algorithms include agglomerative hierarchical clustering (AHC) [10,13], k-means [14,2] and spectral clustering [11,15]. These clustering methods cannot be used in real-time applications since they require complete speech data upfront. Latency-sensitive applications must have speaker labels generated as soon as speech segments are available to the system. We reviewed diarization approaches that are effective in an online setup. In [27] an embedding-based speaker diarization system is presented, it uses *d-vectors* [26] with an LSTM-based scoring function in combination with spectral clustering to successfully perform offline diarization; however, the diarization error rate almost doubles in its online modality. Another online diarization approach is introduced in [7], they propose a deep neural network (DNN) embedding suitable for online processing referred to as speaker-corrupted embedding. The diarization algorithm uses cosine similarity to compare the speaker models and the segments embeddings to make the labeling decisions. A promising approach for diarization is the use of acoustic features of a speaker to target the system's detection to their speech. In [12], an initial estimation of target's speaker features (i-vectors) is performed with clustering-based diarization, providing excellent performance in CHiME-6. Although, this is an offline approach, it could be extended to an online setup.

In this paper, we propose an online speaker tracking pipeline by replacing the unsupervised offline clustering module from the standard diarization system with an online tracking method that uses a DNN as a robust embedding classifier. The main idea is to mimic the PLDA, scoring the similarity of each hypothesized speaker at every segment of a recording. As shown in Fig. 1, our speaker tracking system shares many of its components with the standard diarization pipeline (segmentation, embedding extraction, clustering, and resegmentation) [4,30,18], with the main difference being the removal of the clustering algorithm.

The experimental results on CALLHOME and DIHARD II single-channel [16] reveal that our method achieves competitive results in comparison to the PLDA baseline, while improving the verification performance in EER and minDCF[3].

2 Methodology

In this section, we introduce our speaker tracking framework; Fig. 1 illustrates the overall steps of our tracking pipeline.

2.1 Speech Segmentation and Embedding Extraction

The first module in our pipeline is inspired by the standard diarization system. It uses a Voice Activity Detector (VAD) to determine the speech parts in the input audio signal, excluding the non-speech regions from subsequent processing. A sliding window further divides these regions into a set of smaller, overlapping speech segments, which are the units of audio that can be attributed to a speaker, establishing the temporal resolution of the speaker tracking results. We decided to use an oracle VAD as a segmentation mechanism to focus our efforts on checking whether our proposed architecture can track speakers accurately.

Embedding extraction The next step in the pipeline is to extract an embedding from each segment; such embeddings will be used in two tasks: develop the hypothesized speaker's models and label the segments. Our system was tested following the i-vector- and x-vector-based approaches [17,20]. The i-vector, introduced by Dehak *et al.* [3], is a speaker representation that provides a way to reduce large-dimensional input speech data to a small-dimensional feature vector that retains most of the relevant channel and speaker information. The x-vector, introduced by Snyder *et al.* [23,20] is an embedding extracted from a deep neural network trained to discriminate between speakers, mapping variable-length speech segments to a fixed-length feature vector. Nowadays, the x-vector approach provides state-of-the-art performance in many speaker recognition fields, such as speaker verification and speaker diarization [19,22,28,16].

2.2 Speaker Model Generation

After extracting the segment embeddings, a speaker model is generated for each hypothesized speaker. In our experimental setup, we compute each speaker model by averaging its first embeddings from ground truth labels. The number of embeddings used in this process depends on a tunable time window that will be

[3] Code available at: https://github.com/CarlosRCS9/kaldi/tree/paper-dnn-tracking/egs/dnn_tracking/v1

analyzed later in this research. We define the variable *model time* as the window width used to generate the speakers' models.

With this approach, the system operates in an online fashion in which, with a few labeled samples of the target speakers, it can find their appearances along the complete audio. In a real-life scenario, we expect to have speech data from the target speakers from previous enrollments or a method to record a speaker model, such as a calibration procedure.

2.3 Speaker Segment Identification

The resulting segment embeddings and the speakers' models are then passed through a speaker identification/verification stage. The speaker-tracking DNN, the key component of our pipeline, performs this task.

According to the run-time latency, the speaker identification module follows an **online** tracking strategy. It produces a speaker label immediately after a segment is available without the knowledge of future segments, making it easier for the system to deal with large amounts of audio data since the clustering stage is no longer used.

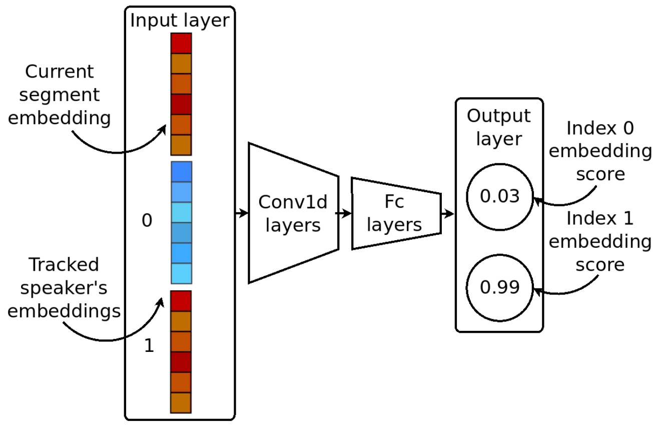


Fig. 2. Network input and output layer for the segment identification process.

Features Fig. 2 illustrates the structure of the network's input and output layers during the segment labeling process. For a given utterance, the input and output sequences of the network (X', Y') are defined as follows:

– The speech segmentation and embedding extraction module provides a sequence of embeddings $X = (x_1, x_2, \ldots, x_T)$, where each $x_t \in \mathbb{R}^b$ has a 1:1 correspondence to the T segments obtained from the input utterance, and b is the dimension of every embedding.

- The speaker model generation module provides the sequence $M = (m_1, m_2, \ldots, m_S)$ where $m_s \in \mathbb{R}^b$, such that each entry of the sequence is a model of one of the S tracked speakers.
- The input sequence of our network is defined as the concatenation of M to each element of X. $X' = \{x_t^\frown M | x_t \in X\}$.
- The sequence $Y = (y_1, y_2, \ldots, y_T)$ is given by the speaker labels of the T segments.
- The output sequence is given by $Y' = \{\Phi(y_t) | y_t \in Y\}$ where $\Phi(y_t) = \{P(m_s | x_t, y_t) | m_s \in M\}$. At training time, Y is given by the ground-truth labels. At inference, Y is computed by the estimated labels.

Architecture Table 1 summarizes the final DNN architecture used in this work. The first three convolutional layers of the network provide a comparison stream for each of the S speakers models and the current audio segment. The similarity measure between the segment embedding and the input speaker models is hence computed using the contextual information of all the speaker models. Note that our architecture intends to track up to S speakers simultaneously. To track less than S speakers, it is required to add zero-padding in the input layer at the location where a speaker model would be.

The last fully-connected feed-forward layers use the S comparison streams to score the similarity of the target speaker model and the incoming segment, with the last layer having a sigmoid activation function instead of softmax. Such activation function allows the network to provide zero scores in all of its outputs when a segment does not belong to any of the tracked speakers, as shown in Fig. 3.

Table 1. Speaker-tracking DNN architecture.

Layer type	**Filters**	**Kernel**	**Input $\times$ output**
Conv1d.ReLU	S^3	3	$b(S+1) \times (b-2)S^3$
Conv1d.ReLU	S^2	3	$(b-2)S^3 \times (b-4)S^2$
Conv1d.ReLU	S	3	$(b-4)S^2 \times (b-6)S$
Dense.ReLU			$(b-6)S \times 32S$
Dense.ReLU			$32S \times 16S$
Dense.Sigmoid			$16S \times S$

Training During training, all possible permutations of the elements of M are computed and appended to every input x_t with two main goals: reduce overfitting by forcing all output neurons to score the same speaker models, and augment the number of training samples. This procedure ensures the DNN scoring to be independent of the speaker model permutation order. Fig. 3 shows how the training data is furthermore augmented by adding zero-padding as a non-speaker model feature. This procedure simulates a verification task during training since the network has to decide whether the current segment embedding belongs to one of the available models or not.

At inference time, our system initializes with an array of hypothesized speaker models (with length less or equal to S). With each recording segment, the similarity of each hypothesized speaker is computed. This is done by appending the models' array to the segment embedding as the network's input, with the output neurons providing similarity scores for each speaker. In an identification setup, we label the segment with the highest score index. If the task requires verification, a certainty threshold is used to label the segments.

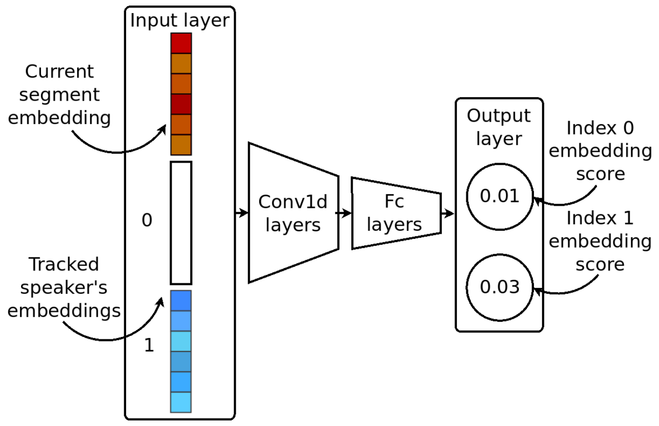


Fig. 3. Input layer with zero padding.

2.4 Probabilistic Linear Discriminant Analysis (PLDA)

The baseline system uses probabilistic linear discriminant analysis (PLDA) scoring as the similarity measure[4]. It has been proven to achieve state-of-the-art performance in many speaker recognition tasks. It provides a powerful distortion-resistant mechanism to distinguish between different speakers and robustness to same variability [9,31,16,29].

2.5 Post-processing

Due to the **online** nature of our pipeline, the post-processing step is applied as soon as a speaker label is inferred. This step refines the tracking results by performing two tasks: merging the contiguous segments that share the same label, and, utilizing a median-filtering-like process to adjust the previously inferred

[4] PLDA scoring computes the loglikelihood ratio between two embeddings

label (x_{t-1}). This process is performed with a window of the last three segments $W_t = (x_{t-2}, x_{t-1}, x_t)$, modifying the in-between speaker label if the surrounding labels are equal to each other, producing three contiguous segments with the same label.

3 Experiments

This section describes our experimental setup and results. We decided on a 1.0 s width and 0.5 s step sliding window at the speech segmentation step, discarding segments shorter than 0.5 s to ensure sufficient speaker information. Both i- and x-vectors were extracted using the Kaldi's CALLHOME diarization recipes[5] [18]. For CALLHOME x-vector experiments, a publicly available [20,21] model and PLDA backend were used.

3.1 Evaluation Metrics

The system performance was evaluated in terms of Equal Error Rate (EER) and minimum Detection Cost Function (minDCF) [25], as the key component of our tracking framework follows a speaker verification approach. Besides, we report Diarization Error Rate (DER) [5] since our framework shares characteristics with the standard diarization system.

3.2 Datasets

We tested our system on two standard public datasets: (1) CALLHOME, it contains 500 utterances distributed across six languages: Arabic, English, German, Japanese, Mandarin, and Spanish. Each utterance contains up to 7 speakers (2) DIHARD II single-channel development and evaluation subsets (LDC2019E31, LDC2019E32), focused on "hard" speaker diarization, contains 5-10 minute English utterances selected from 11 conversational domains, each including approximately 2 hours of audio. Since our approach is supervised, we performed a 2-fold cross-validation on each dataset using standard partitions: callhome1 and callhome2 from Kaldi's CALLHOME diarization recipe [18], and DIHARD II single channel's development and evaluation subsets. Then, the partitions' results are combined to report the averaged DER, EER and minDCF of each dataset.

To evaluate our proposed method in more difficult conditions, we increased the variability of the datasets in two steps. First, we increased the number of non-target speakers by adding to each recording speakers models from all the other recordings as new segments features. Such models were extracted with the same *model time* as the target's speakers. This set is used as the *speaker verification* conditions with its 0.17% target probability.

[5] https://github.com/kaldi-asr/kaldi/tree/master/egs/callhome diarization/v1 and /v2

The second modification to the datasets aims to give us a better hint of the system's performance in a real-life scenario, by increasing the number of overlapping speech instances, as both datasets have a low percentage of speaker overlap (CALLHOME ∼16%, DIHARD II single channel ∼9%). To increase the overlapping examples, we use the ground-truth labels to extract the non-overlapping audio segments of each speaker. Then, those segments are merged into a set of single-speaker utterances for each recording. After that, the single-speaker utterances are pairwise overlapped to create a new set of two-speaker-overlapping utterances. Finally, the new overlapping utterances are cut into segments (following Algorithm 1) and inserted into their original recordings at random locations with a uniform distribution.

Algorithm 1: Get the lengths to cut from an utterance

```
Result: A list of lengths to cut from an utterance
$T$ is the length of an utterance;
$L$ is an empty list;
while $T > 1.5$ do
    $l \leftarrow \sqrt{T}$;
    $T \leftarrow T - l$;
    append $l$ to $L$;
end
append $l$ to $L$;
```

The resulting datataset is used as the *speaker overlap* condition. It contains an additional ∼18% of speaker overlap in CALLHOME, and ∼30% in DIHARD II single channel. It is worth mentioning that the *speaker verification* condition is a subset of the *speaker overlap* one, so the target probability increases to 0.35% with the additional target examples.

3.3 Baseline

We compared the performance of our proposed system with a conventional offline diarization method: PLDA scoring with AHC, following the Kaldi's CALLHOME diarization recipe [18] with oracle number of speakers. The i- and x-vector PLDA backends were trained for each cross-validation fold with the recipe and used along all experiments.

Our primary baseline method follows the same procedure as our proposed system, but replaces the DNN-based identification module with a PLDA. The PLDA backends are the same as the ones used in the offline diarization baseline. We report the averaged results across the dataset partitions.

3.4 Results

The first set of experiments follows optimal conditions for speaker tracking: the input audio signal contains only speech from two tracked-speakers, and there is

Table 2. DER (%), EER (%) and minDCF (52% target probability) on two datasets given the optimal conditions.

Model time	**PLDA**			**DNN**		
	DER	**EER**	**minDCF**	**DER**	**EER**	**minDCF**
CALLHOME i-vector (Offline DER: 16.95)						
3.0 s	7.11	19.03	0.39	5.86	4.33	0.08
5.5 s	5.47	16.09	0.33	4.99	3.32	0.06
10.5 s	4.42	14.32	0.29	**4.30**	**2.84**	**0.05**
x-vector (Offline DER: 15.84)						
3.0 s	9.86	17.63	0.36	11.46	11.45	0.23
5.5 s	7.21	14.18	0.29	8.61	8.10	0.16
10.5 s	5.53	11.66	0.24	5.74	4.83	0.10
DIHARD II i-vector (Offline DER: 21.53)						
3.0 s	18.96	36.62	0.75	18.22	21.95	0.45
5.5 s	16.11	34.73	0.72	13.80	14.80	0.30
10.5 s	13.23	33.70	0.69	**11.36**	**12.45**	**0.26**
x-vector (Offline DER: 21.36)						
3.0 s	15.80	28.03	0.58	20.20	27.25	0.56
5.5 s	11.95	24.86	0.51	18.20	25.75	0.53
10.5 s	10.17	23.66	0.49	12.25	15.75	0.32

no overlapping speech. To have 2-speaker recordings, we applied a mask at the instances where a third speaker appeared in each recording.

We took this decision based on the fact that if we filtered out entire recordings with more than two speakers, we would have lost a large percentage of each dataset (60% CALLHOME and 47% DIHARD II single channel).

Table 2 show the results. All offline diarization results follow the same trend: x-vectors perform better than i-vectors, with the PLDA-based tracking having a clear advantage over it's offline counterpart. The reason behind this behavior is that the tracking pipeline receives the speakers' models beforehand.

An interesting phenomenon is that the PLDA-based tracking in CALLHOME shows better DER performance with i-vectors rather than x-vectors (also happens in Table 3). We believe that this is related to the generation of speakers models with embeddings trained with less data (as it does not happen in DIHARD II, whose x-vector extractor was trained with VoxCeleb data).

In most cases, the DNN-based tracking outperforms the PLDA baseline in the verification metrics (EER, minDCF). It is reasonable for several reasons: (1) The network's training promoted a binary-like similarity score. (2) Due to the speaker models permutations performed in training, the network had to perform more rejections. (3) The similarity score for each speaker is computed with all speakers' models available as contextual information.

For DER, the PLDA system has a clear advantage. Still, the DNN pipeline keeps close results despite its relatively simple architecture; we expect to overcome this by moving to a recurrent neural network (RNN).

The most interesting phenomenon in Tables 2 and 3 is that in all DNN results, the x-vectors have a clear disadvantage against i-vectors in all the provided metrics. We reviewed and discarded possible procedural and architectural mistakes. The same behavior was found in [12] with a similar DNN architecture. We agree that a possible reason for this behavior is the need for a complex DNN architecture to score an embedding derived from a much more complex architecture.

Table 3. DER (%), EER (%) and minDCF (17% target probability) given the speaker verification conditions.

Model time	**PLDA**			**DNN**		
	DER	**EER**	**minDCF**	**DER**	**EER**	**minDCF**
CALLHOME i-vector						
5.5 s	5.47	22.27	0.80	4.56	7.75	0.28
10.5 s	4.42	22.22	0.83	**4.43**	**5.89**	**0.21**
x-vector						
5.5 s	7.20	11.09	0.41	8.13	8.74	0.30
10.5 s	5.53	9.50	0.37	6.24	4.40	0.19
DIHARD II i-vector						
5.5 s	16.11	32.64	0.99	15.42	17.96	0.62
10.5 s	13.23	32.89	0.99	**11.44**	**14.67**	**0.54**
x-vector						
5.5 s	11.85	15.59	0.70	16.84	15.89	0.66
10.5 s	10.25	15.17	0.68	13.77	15.06	0.73

Finally, we evaluate our proposed system considering overlapped speech, as described in Section 3.2. In this set of experiments, the number of tracked speakers is fixed to 2, with the input audio signal containing non-overlapping and overlapping speech from them in addition to non-target speakers.

In order to select a segment as an overlap of the tracked speakers, it was necessary to train a DNN model able to work with three speaker models simultaneously ($S = 3$), the first two models representing each of the two speakers, and the third one, their overlap; as shown in Fig. 4. During test time, an embedding of the tracked speaker's overlapping speech was used as the third model, so when the network selected such embedding, we knew it was overlapping speech from the tracked speakers.

In 4, we report a loss in DER performance as overlapping speech dramatically increases the complexity of tracking. However, even with an additional model to score, we keep competitive performance in both EER and minDCF since DNN keeps its binary-like scoring while selecting overlapping speech.

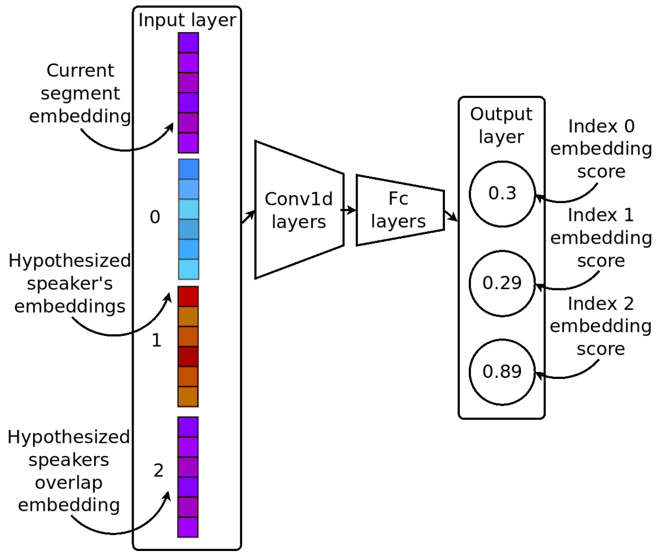


Fig. 4. Network input and output layers extended for overlap detection.

Table 4. DER (%), EER (%) and minDCF (35% target probability) for **i-vector**, given the speaker overlap conditions.

Model time	**CALLHOME**			**DIHARD II**		
	DER	**EER**	**minDCF**	**DER**	**EER**	**minDCF**
3.0 s	20.82	13.20	0.35	37.17	29.46	0.73
5.5 s	15.78	9.72	0.26	31.99	24.78	0.64
10.5 s	**12.52**	**7.50**	**0.20**	**28.78**	**21.32**	**0.55**

4 Conclusions

In this paper, we propose a novel embedding-based speaker-tracking DNN model focused on online tracking. We demonstrated our approach's efficiency through several experiments on two standard public datasets: CALLHOME and DIHARD II single channel. Results show better performance than the PLDA baseline in EER and minDCF in different experimental conditions.

For future research, we would like to extend our current DNN model to an **online** diarization and tracking system, where a recurrent neural network (RNN) will be responsible for selecting and updating the speaker models without having to resort to external sources. We expect such a system to provide not only the diarization results but also the set of speaker models that it will generate during an adaptive diarization process.

Acknowledgments. We would like to thank Diego Nigel Joaquin Campos Sobrino and Mario Alejandro Campos Soberanis for their helpful discussions.

References

1. Bonastre, J.F., Delacourt, P., Fredouille, C., Merlin, T., Wellekens, C.: A speaker tracking system based on speaker turn detection for NIST evaluation. Acoustics, Speech, and Signal Processing, IEEE International Conference on 2, II1177–II1180 (06 2000)
2. Dabbabi, K., Salah, H., Adnen, C.: Hybridization DE with k-means for speaker clustering in speaker diarization of broadcasts news. International Journal of Speech Technology 22 (09 2019)
3. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. IEEE Transactions on Audio, Speech, and Language Processing 19(4), 788–798 (May 2011)
4. Diez, M., Landini, F., Burget, L., Rohdin, J., Silnova, A., Žmolíková, K., Novotny, O., Veselý, K., Glembek, O., Plchot, O., Mošner, L., Matejka, P.: BUT system for DIHARD speech diarization challenge 2018. pp. 2798–2802 (09 2018)
5. Fiscus, J., Ajot, J., Michel, M., Garofolo, J.: The rich transcription 2006 spring meeting recognition evaluation. pp. 309–322 (01 2006)
6. Garcia, P., Villalba, J., Bredin, H., Du, J., Castan, D., Cristia, A., Bullock, L., Guo, L., Okabe, K., Nidadavolu, P.S., Kataria, S., Chen, S., Galmant, L., Lavechin, M., Sun, L., Gill, M.P., Ben-Yair, B., Abdoli, S., Wang, X., Bouaziz, W., Titeux, H., Dupoux, E., Lee, K.A., Dehak, N.: Speaker detection in the wild: Lessons learned from JSALT 2019 (2019)
7. Ghahabi, O., Fischer, V.: Speaker-Corrupted Embeddings for Online Speaker Diarization. In: Proc. Interspeech 2019. pp. 386–390 (2019)
8. Karim, D., Adnen, C., Salah, H.: A system for speaker detection and tracking in audio broadcast news. In: 2017 International Conference on Engineering MIS (ICEMIS). pp. 1–5 (2017)
9. Khosravani, A., Homayounpour, M.M.: A PLDA approach for language and text independent speaker recognition. Computer Speech & Language 45, 457 – 474 (2017), http://www.sciencedirect.com/science/article/pii/S0885230816302972
10. Landini, F., Wang, S., Diez, M., Burget, L., Matějka, P., Žmolíková, K., Mošner, L., Plchot, O., Novotný, O., Zeinali, H., Rohdin, J.: BUT system description for DIHARD speech diarization challenge 2019 (2019)
11. Lin, Q., Yin, R., Li, M., Bredin, H., Barras, C.: LSTM based similarity measurement with spectral clustering for speaker diarization. Interspeech 2019 (Sep 2019)
12. Medennikov, I., Korenevsky, M., Prisyach, T., Khokhlov, Y., Korenevskaya, M., Sorokin, I., Timofeeva, T., Mitrofanov, A., Andrusenko, A., Podluzhny, I., Laptev, A., Romanenko, A.: Target-speaker voice activity detection: a novel approach for multi-speaker diarization in a dinner party scenario (2020)
13. Novoselov, S., Gusev, A., Ivanov, A., Pekhovsky, T., Shulipa, A., Avdeeva, A., Gorlanov, A., Kozlov, A.: Speaker diarization with deep speaker embeddings for DIHARD challenge II. In: INTERSPEECH (2019)
14. Pal, M., Kumar, M., Peri, R., Narayanan, S.: A study of semi-supervised speaker diarization system using GAN mixture model (2019)
15. Park, T.J., Han, K.J., Kumar, M., Narayanan, S.: Auto-tuning spectral clustering for speaker diarization using normalized maximum eigengap. IEEE Signal Processing Letters 27, 381–385 (2020)

16. Ryant, N., Church, K., Cieri, C., Cristia, A., Du, J., Ganapathy, S., Liberman, M.: The second DIHARD diarization challenge: Dataset, task, and baselines (2019)
17. Sell, G., Garcia-Romero, D.: Speaker diarization with PLDA i-vector scoring and unsupervised calibration. 2014 IEEE Workshop on Spoken Language Technology, SLT 2014 - Proceedings pp. 413–417 (04 2015)
18. Sell, G., Snyder, D., McCree, A., Garcia-Romero, D., Villalba, J., Maciejewski, M., Manohar, V., Dehak, N., Povey, D., Watanabe, S., Khudanpur, S.: Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge. pp. 2808–2812 (09 2018)
19. Snyder, D., Garcia-Romero, D., Sell, G., McCree, A., Povey, D., Khudanpur, S.: Speaker recognition for multi-speaker conversations using x-vectors. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5796–5800 (May 2019)
20. Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S.: X-vectors: Robust DNN embeddings for speaker recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5329–5333 (April 2018)
21. Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S.: X-vectors: Robust dnn embeddings for speaker recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE (2018)
22. Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Povey, D., Khudanpur, S.: Spoken language recognition using x-vectors. pp. 105–111 (06 2018)
23. Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., Khudanpur, S.: Deep neural network-based speaker embeddings for end-to-end speaker verification. pp. 165–170 (12 2016)
24. Sonmez, K., Heck, L., Weintraub, M.: Speaker tracking and detection with multiple speakers (September 1999), https://www.microsoft.com/en-us/research/publication/speaker-tracking-and-detection-with-multiple-speakers/
25. Van Leeuwen, D., Brummer, N.: An introduction to application-independent evaluation of speaker recognition systems. vol. 4343, pp. 330–353 (01 2007)
26. Variani, E., Lei, X., McDermott, E., Moreno, I., Gonzalez-Dominguez, J.: Deep neural networks for small footprint text-dependent speaker verification. pp. 4052–4056 (05 2014)
27. Wang, Q., Downey, C., Wan, L., Mansfield, P.A., Moreno, I.L.: Speaker diarization with LSTM (2017)
28. Xu, L., Das, R.K., Yılmaz, E., Yang, J., Li, H.: Generative x-vectors for text-independent speaker verification (2018)
29. Zajíc, Z., Kunešová, M., Hruz, M., Vanek, J.: UWB-NTIS speaker diarization system for the DIHARD II 2019 challenge (05 2019)
30. Zhang, A., Wang, Q., Zhu, Z., Paisley, J., Wang, C.: Fully supervised speaker diarization (2018)
31. Zhao, C., Li, L., Wang, D., Pu, A.: Local training for PLDA in speaker verification. In: 2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA). pp. 156–160 (Oct 2016)