


RESEARCH

Open Access



Accent modification for speech recognition of non-native speakers using neural style transfer

Kacper Radzikowski^{1,2*†} , Le Wang^{1†}, Osamu Yoshie^{1†} and Robert Nowak^{2†}

Abstract

Nowadays automatic speech recognition (ASR) systems can achieve higher and higher accuracy rates depending on the methodology applied and datasets used. The rate decreases significantly when the ASR system is being used with a non-native speaker of the language to be recognized. The main reason for this is specific pronunciation and accent features related to the mother tongue of that speaker, which influence the pronunciation. At the same time, an extremely limited volume of labeled non-native speech datasets makes it difficult to train, from the ground up, sufficiently accurate ASR systems for non-native speakers.

In this research, we address the problem and its influence on the accuracy of ASR systems, using the style transfer methodology. We designed a pipeline for modifying the speech of a non-native speaker so that it more closely resembles the native speech. This paper covers experiments for accent modification using different setups and different approaches, including neural style transfer and autoencoder. The experiments were conducted on English language pronounced by Japanese speakers (*UME-ERJ* dataset). The results show that there is a significant relative improvement in terms of the speech recognition accuracy. Our methodology reduces the necessity of training new algorithms for non-native speech (thus overcoming the obstacle related to the data scarcity) and can be used as a wrapper for any existing ASR system. The modification can be performed in real time, before a sample is passed into the speech recognition system itself.

Keywords: Speech recognition, Style transfer, Autoencoder, Non-native speaker, Machine learning, Neural network, Deep learning, Artificial intelligence

1 Introduction

Automatic speech recognition is a function that has been the subject of extensive research for decades. Enabling the communication between a human and a machine has been one of the most difficult problems to tackle and one of the most intensively studied topics.

Recently developed speech recognition tools can recognize speech with an almost human-like accuracy, depending on the dataset and benchmark test used [1]. Such performance can be achieved only when the system is used for recognizing the speech of native speakers (i.e., the native speakers of the language represented by the dataset used to train the ASR system). In the case of non-native speakers of the language of the ASR system, the accuracy of even the most advanced speech recognition systems cannot reach human-like or even high levels [2, 3]. The main reason for this drop is the presence of patterns related to the speaker's mother tongue which can influence the pronunciation of the second language. This makes their language biased to some extent which causes

*Correspondence: radzikowskikacper@gmail.com

†Kacper Radzikowski, Le Wang, Osamu Yoshie, and Robert Nowak contributed equally to this work.

¹Waseda University, Graduate School of Information, Production and Systems, Kitakyushu, Japan

²Warsaw University of Technology, Institute of Computer Science, Warsaw, Poland

the speech recognition system's accuracy to decrease in such cases [4–7]. The current pace of development in the global economy, education, and mobility of the workforce creates the need to properly recognize the speech of non-native speakers who nowadays represent the vast majority of users.

Traditional approaches for training speech recognition classifiers usually tend to employ supervised learning techniques [8–15]. While perfectly fitted for cases of recognizing speech of most popular languages worldwide, supervised learning methodologies will not produce classifiers of a decent quality for non-native speakers. The main reason is the lack of labeled datasets of non-native speech which would be large enough to be used as a training set in a supervised learning algorithm. We have dealt with the problem of data scarcity regarding the non-native speech in our previous research [16–19]. Our idea at the time was to use unlabeled datasets (e.g., Japanese people who speak English and English corpus) in a setup called dual supervised learning.

This time we plan to tackle the problem of non-native accents using the style transfer methodology [20, 21] adapted for the case of speech. The application of style transfer in the audio domain is not new. In [22], the authors investigated how to transfer the style of a reference audio signal to a target audio content. They proposed a flexible framework for the task, which uses a sound texture model to extract statistics characterizing the reference audio style, followed by an optimization-based audio texture synthesis to modify the target content. In contrast to mainstream optimization-based visual transfer methods, the process proposed by the authors is initialized by the target content instead of random noise and the optimized loss is only about texture, not structure.

In [23], the authors presented a new machine learning technique for generating music and audio signals. The focus of their work was to develop new techniques parallel to what has been proposed for artistic style transfer for images by others. They presented two cases of modifying an audio signal to generate new sounds. A feature of their method is that a single architecture can generate these different audio-style-transfer types using the same set of parameters which otherwise require complex hand-tuned diverse signal processing pipelines.

To tackle the problem of non-native speech recognition, we plan to apply and adjust style transfer to the domain of speech and sound in order to create an algorithm for real-time pronunciation and accent modification. Having done that, it would enable the possibility of creating a wrapper over already existing and trained ASR systems. Such an approach could allow the modification of a non-native speaker's voice in real time, so that the ASR system used at the time can recognize the speech with a higher degree of accuracy.

2 Methods

Within this article, we present an approach for handling the problem related to a specific, non-native accent. We created a method that modifies the accent of a non-native speaker so that it resembles the accent of a native speaker to a higher extent. The purpose of this method is to increase the accuracy of ASR systems which had already been developed and trained using a native speech dataset, without the necessity to train new ASR models adapted for a specific non-native accent.

Our idea is to modify the accent of speech using the representation of a sound wave in a graphical domain, i.e., a spectrogram.

The general flow of our approach is depicted in Fig. 1.

At the beginning, we transform a sound wave file into a spectrogram (the process indicated as A on the diagram). Secondly, accent modification is performed. Within the second step, we decided to check two ways of modifying accent with spectrograms and they are described in detail in Sections 2.1 and 2.2.

Finally, the sound wave, in modified form, is fed into the ASR system in order to recognize the speech into text. In our research, we experimented with two kinds of speech recognition process. As shown in the figure, one way is to revert the modified spectrogram back to the sound wave (the process indicated as B on the diagram) in a form of WAV file and then feed it to a previously agreed ASR system. The second way indicated as C is to feed the spectrogram directly to another ASR system (adapted for recognizing the speech from spectrograms) created within this research (Section 2.3).

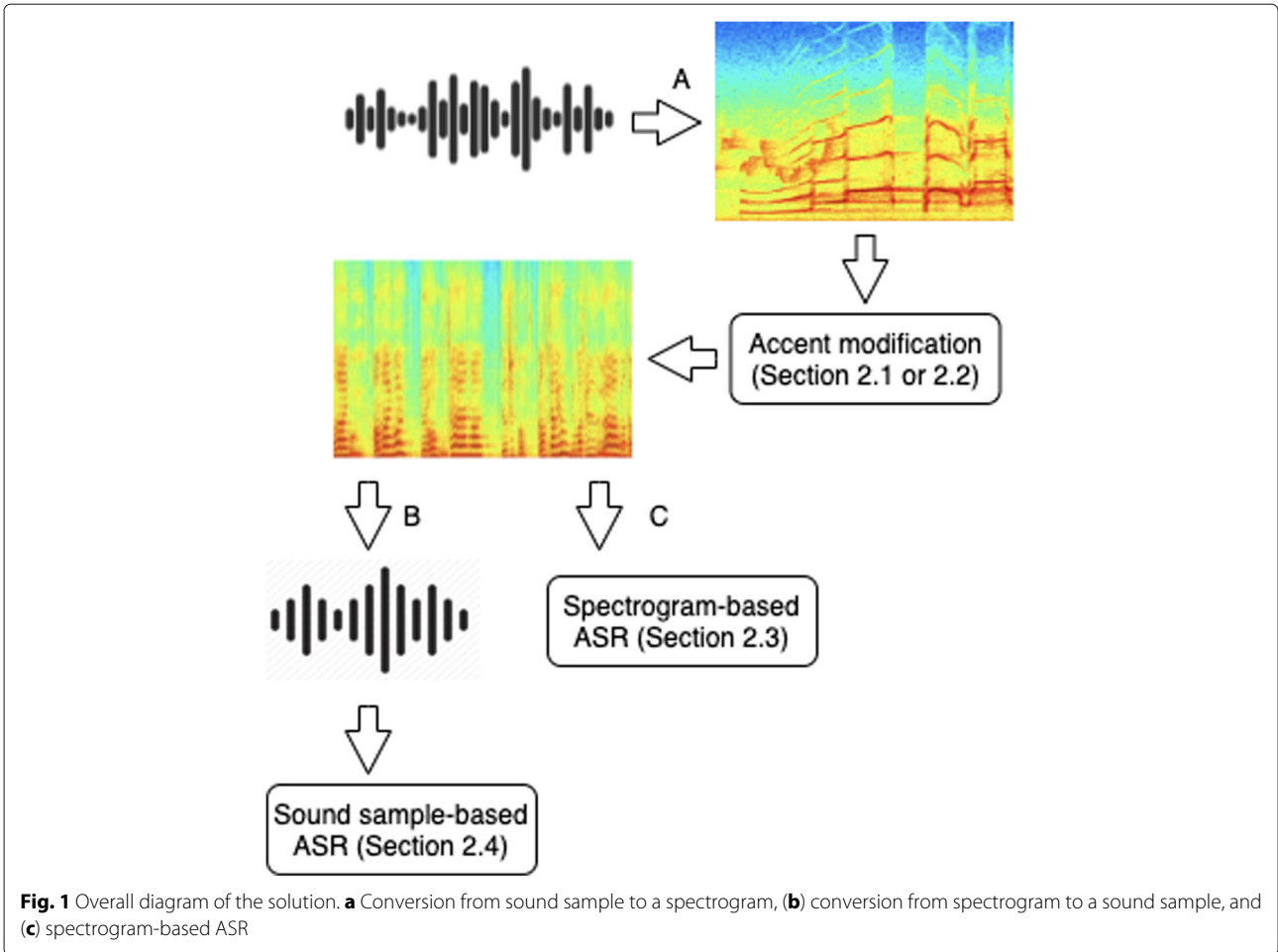
The accent modification algorithms are trained in a way that they learn how to modify the spectrogram representing non-native speech to one resembling the same utterance by a native speaker. The correctness of the modified speech is determined by the accuracy of the speech recognition system trained on a dataset containing native speaker samples, allowing us to evaluate the quality of accent modification. The criterion which decides whether or not the style-modifying algorithm can perform well is a reduction in the metrics related to the error yielded during the inference using the ASR networks, which were trained on native speaker samples.

2.1 Accent modification using autoencoder

In this approach, we came up with an autoencoder based on a convolutional neural network (CNN) [24]. Our idea is to employ such a network for the purpose of changing the pronunciation style (Fig. 2).

The autoencoder was written using the Keras library [25], and its detailed architecture is described in Table 1.

During the training phase, the autoencoder is fed spectrograms of samples of non-native speakers, whereas the autoencoder's output is compared against the spectrograms



of exactly the same utterances pronounced by native speakers of the particular language. Then back-propagation causes the autoencoder to learn the conversion of the same words and sentences from the speech containing a non-native accent to the one with the modified accent.

During the inference phase, the input spectrogram created in the first step of our pipeline is fed into the

autoencoder as input data. The output of the autoencoder is a spectrogram which is slightly converted according to the CNN layers' weights learned after the training.

2.2 Accent modification using style transfer-based approach

Another approach we decided to experiment with employs a style transfer methodology adapted for the

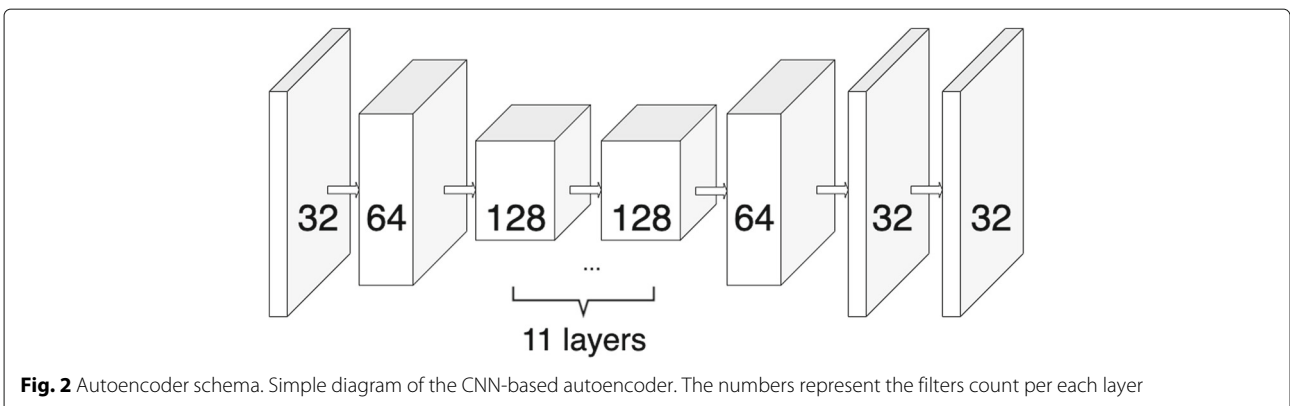


Table 1 Detailed architecture of the autoencoder

Layer	Output shape	Parameters
Conv2D (F=32, K=3)	B, X, T, 32	417344
Conv2D (F=64, K=3)	B, X, T, 64	18496
Conv2D (F=128, K=3)	B, X, T, 128	73856
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2D (F=128, K=3)	B, X, T, 128	147584
Conv2DTr (F=64, K=3)	B, X, T, 64	73792
Conv2DTr (F=32, K=3)	B, X, T, 32	18464
Conv2D (F=32, K=3)	B, X, T, 32	82976
	Total params:	2,160,768
	Trainable params:	2,160,768
	Non-trainable params:	0

Conv2D 2-dimensional convolutional layer, Conv2DTr 2-dimensional convolutional transpose layer, F number of filters, K kernel size, B batch dimension, X dimension related to spectrogram's frequency, T dimension related to spectrogram's time steps

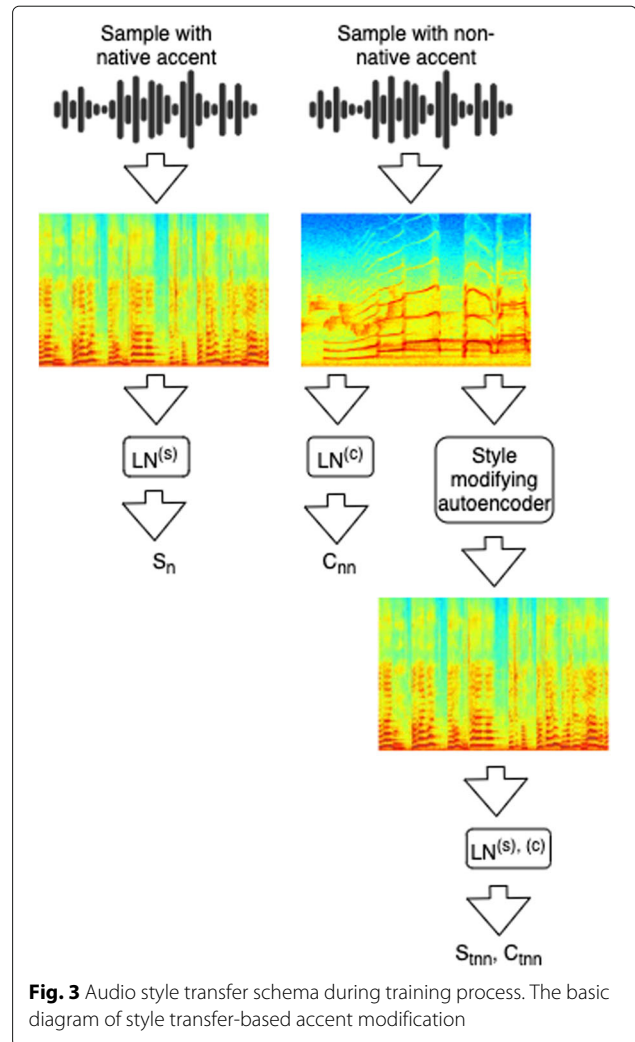
domain of speech and sound. Specifically we decided to create a method that resembled the style transfer feedforward algorithm from the graphical domain.

To briefly explain the problem of the graphical style transfer, we try to modify an image in a way that its style resembles the style of another, a so-called style image. At the same time, the content of the image ideally should not be modified.

The general flow of the accent modification using style transfer is depicted in Fig. 3.

In order to utilize such a setup, we first train a network (here, called a loss network) separately, beforehand, which will be used as a speech recognizer in the style transfer approach. Its role is to separate speech spectrograms into multiple layers using a convolutional network. It will be used for extracting content (related to the utterance) and style (related to the accent and pronunciation) from the images (spectrograms). The loss network is depicted on the diagram as LN . In order to properly extract style and content from the input spectrograms, the loss network must be trained using data from such a domain. The data utilized in the training process is described in Section 3.1

As the loss network model for automatic speech recognition tasks, we combined properties of convolutional recurrent layers, where the former layers become, in fact,

**Fig. 3** Audio style transfer schema during training process. The basic diagram of style transfer-based accent modification

employed as feature extractors. Convolutional neural networks have been proven to give outstanding results when applied to images, here spectrograms. They are able to detect and learn local features which are later passed on to recurrent layers. The architecture of the neural network is depicted in Table 2. It accepts an image as the input and outputs a sequence of letters.

The main step of this approach is training the autoencoder for style modification, which performs the essence of the idea. Its architecture is the same as the one of the autoencoder used in the previous approach for accent modification and is described in detail in Table 1.

During one training step, the spectrogram of a sample with a native accent is fed into the loss network, which extracts style matrix S_n from certain layers. It is depicted in the diagram as $LN^{(s)}$. Next, the spectrogram of a sample containing a non-native accent is pushed through the same loss network which results in extraction of content matrix C_{nn} . The process is depicted in the diagram

Table 2 Detailed architecture of the CNN-RNN model used as the loss network in style transfer approach

Layer	Output shape	Parameters
InputLayer	B, T, 161	0
Conv1D (F=220, K=3)	B, T, 220	389840
Conv1D (F=220, K=3)	B, T, 220	389840
Maxpool (P=2)	B, T, 220	880
Conv1D (F=150, K=3)	B, T, 150	265800
Conv1D (F=150, K=3)	B, T, 150	265800
Maxpool (P=2)	B, T, 150	600
Conv1D (F=100, K=3)	B, T, 100	177200
Conv1D (F=100, K=3)	B, T, 100	177200
Maxpool (P=2)	B, T, 100	400
Conv1D (F=80, K=3)	B, T, 80	141760
Conv1D (F=80, K=3)	B, T, 80	141760
Maxpool (P=2)	B, T, 80	320
Conv1D (F=80, K=3)	B, T, 80	141760
Conv1D (F=80, K=3)	B, T, 80	141760
Maxpool (P=2)	B, T, 80	320
Conv1D (F=80, K=3)	B, T, 80	141760
Conv1D (F=80, K=3)	B, T, 80	141760
Maxpool (P=2)	B, T, 80	320
Conv1D (F=80, K=3)	B, T, 80	141760
Conv1D (F=80, K=3)	B, T, 80	141760
Bidirectional (U=200)	B, T, 400	505200
BatchNormalization	B, T, 400	1600
TimeDistributed	B, T, 29	11629
Dropout	B, T, 29	0
TimeDistributed	B, T, 29	870
SoftmaxActivation	B, T, 29	0
	Total params:	3,038,059
	Trainable params:	3,038,059
	Non-trainable params:	0

Conv1D 1-dimensional convolutional layer, *Maxpool* max pooling layer, *Bidirectional* wrapper with RNN, *BatchNormalization* batch normalizing layer, *TimeDistributed* layer for every temporal slice of the input, *Dropout* dropout layer, *SoftmaxActivation* softmax activation function layer, *P* pool size, *U* number of hidden units in the RNN

as $\mathbf{LN}^{(c)}$. The sample is also fed into the style modifying autoencoder which outputs a modified spectrogram that is fed into the loss network to extract matrices representing style and content of the transformed sample (\mathbf{Stnn} , \mathbf{Ctnn} respectively). It is symbolized in the figure as $\mathbf{LN}^{(s)}$, (c).

After having received \mathbf{Sn} , \mathbf{Cnn} , \mathbf{Stnn} , \mathbf{Ctnn} , we can formulate the content and style losses. Content loss is calculated as:

$$L_c = \sum_l \sum_{i,j} \left(\alpha \mathbf{Cnn}_{i,j}^l - \alpha \mathbf{Ctnn}_{i,j}^l \right)^2 \quad (1)$$

where l is the set of convolutional layers representing the content of the sound wave.

Style loss is calculated as:

$$L_s = \sum_l \sum_{i,j} \left(\beta \mathbf{Gn}_{i,j}^l - \beta \mathbf{Gtnn}_{i,j}^l \right)^2 \quad (2)$$

where:

\mathbf{Gn}^l —the Gram matrix of l th layer of \mathbf{Sn} received from the loss network

\mathbf{Gtnn}^l —the Gram matrix of the l th layer of \mathbf{Stnn}

Gram matrix is the result of the multiplication of the matrix by its transpose.

Therefore, the final loss function is represented as:

$$L = L_s + L_c \quad (3)$$

After having formulated our loss function, we backpropagate the error to train the style modifying autoencoder network for the task of accent modification. At this step, the weights of the loss network are already frozen and do not take part in the training process.

Such sequences are executed repeatedly with samples drawn from native speech datasets and non-native ones, respectively. It is worth mentioning that in cases of style transfer, as opposed to the autoencoder approach, it is not necessary for both spectrograms (with native and non-native accents) to represent the same content. As mentioned in the experimental part of this article, we performed several runs of training the autoencoder in order to find the best subsets of convolutions to represent the style and content layers.

During the inference phase, we use only the trained autoencoder that modifies the accent of a new sample.

2.3 Speech recognition using spectrograms

At the end of our pipeline, the speech recognition process is performed. One of the two approaches we experimented with is using an ASR system trained on spectrograms. We decided to create a model for the speech recognition using spectrograms converted from WAV files. The architecture of the network playing the role of the ASR system is depicted in Table 3. We used a combination of convolutional and recurrent neural networks (CNN-RNN) in order to train a new speech recognition system. Similar to the loss network mentioned earlier, this network also accepts images and outputs a sequence of letters.

We trained the network using a popular and publicly available dataset *LibriSpeech*. The details, together with the metrics and the results of the training process, are shown in Section 3.

2.4 Speech recognition using sound sample-based ASR

2.4.1 Cloud-based ASR

Another way of speech recognition that we decided to check is an online ASR service. In our research, we

Table 3 Detailed architecture of the CNN-RNN model used as the ASR module

Layer	Output shape	Parameters
InputLayer	B, T, 161	0
Conv1D (F=250, K=3)	B, T, 250	443000
Conv1D (F=250, K=3)	B, T, 250	443000
Maxpool (P=2)	B, T, 250	940
Conv1D (F=150, K=3)	B, T, 150	265800
Conv1D (F=150, K=3)	B, T, 150	265800
Maxpool (P=2)	B, T, 150	600
Conv1D (F=100, K=3)	B, T, 100	177200
Conv1D (F=100, K=3)	B, T, 100	177200
Maxpool (P=2)	B, T, 100	400
Conv1D (F=80, K=3)	B, T, 80	141760
Conv1D (F=80, K=3)	B, T, 80	141760
Bidirectional (U=200)	B, T, 400	505200
BatchNormalization	B, T, 400	1600
TimeDistributed	B, T, 29	11629
Dropout	B, T, 29	0
TimeDistributed	B, T, 29	870
SoftmaxActivation	B, T, 29	0
	Total params:	2,576,759
	Trainable params:	2,576,759
	Non-trainable params:	0

decided upon Google Cloud Speech-to-Text and used the results of recognized text to calculate accuracy metrics.

2.4.2 TDNN architecture-based ASR

Another network—Time Delay Neural Network (TDNN)—was used as an evaluation tool in our methodology.

3 Results

3.1 Datasets used

One of the two datasets utilized within this research is a set of around 75,000 samples called *English Speech Database Read by Japanese Students (UME-ERJ)* containing Japanese, as well as Americans, pronouncing English sentences.

1. Sentences for learning phonemic pronunciation:

- 460 phonetically balanced sentences
- 32 sentences including phoneme sequences difficult for Japanese to pronounce correctly
- 100 sentences designed for test set
- 302 minimal-pair words
- 300 phonemically balanced words

2. Sentences for learning prosody of speech:

- 94 sentences with various intonation patterns
- 120 sentences with various accent and rhythm patterns
- 109 words with various accent patterns

The same dataset was used in our previous work [16]. The dataset was employed for training both the autoencoder in Section 2.1 and the style transfer network in Section 2.2, as it contains sentences and words pronounced by both native and non-native speakers. The training dataset contains around 18,662 pairs of spectrograms representing the exact same utterances from native and non-native speakers. This amount of the recordings represents around 26 h of speech. The remaining test and validation subsets did not overlap with the training subset.

Another dataset used in the research is the *LibriSpeech* dataset. It was used to train both the spectrogram-based ASR module (after converting samples to spectrograms) used as the last part of our pipeline (Section 2.3) and the TDNN-based network (Section 2.4.2). Another application of the dataset is training the loss network for the style transfer approach in one of the accent modification variants (Section 2.2). Also, we used the dataset to train the TDNN-based ASR system, as another network evaluating the performance of our pipeline.

The summary of utilized datasets is shown in Table 4.

3.2 Experiments and metrics

The autoencoder introduced in Section 2.1 as well as the loss network (Section 2.2) and the spectrogram-based ASR system (Section 2.3) were trained using the Connectionist temporal classification (CTC, [26]) function.

In our research, we designed separate experiments for several processes in our pipeline. Namely, we performed experiments and evaluated the results for:

1. Relative improvement in the speech recognition accuracy in case of autoencoder-based accent modification, including both approaches for ASR in the final stage
2. Relative improvement in the speech recognition accuracy in cases of audio style transfer-based accent modification, including both approaches for ASR. In this approach, we performed several runs of training

Table 4 Summary and splits of the utilized datasets

Trained network	Utilized dataset
Autoencoder	UMEERJ (18,662 samples subset)
Style transfer network	UMEERJ (split (0.8/0.1/0.1))
Loss network	LibriSpeech (train-clean-360)
CNN-RNN-based ASR	LibriSpeech (train-clean-360)
TDNN-based ASR	LibriSpeech (train-clean-360)

the style modifying autoencoder to check the best combination of subsets of style and content layers in the loss network

3.3 Metrics

We employed two different evaluation processes depending on the experiment type.

As a quality metric for the speech recognition processes (loss network, ASR module, and the cloud-based service), we chose three different metric types. First is the standard Word Error Rate (WER) and the second one is Character Error Rate (CER), which is expressed as:

$$CER = \frac{i + s + d}{n} \quad (4)$$

where:

- i —number of insertions
- s —number of substitutions
- d —number of deletions
- n —total number of characters

Another metric type introduced is phoneme similarity [27]. It is expressed as Mean Similarity Score (MSS) in the results section of our work.

As for the evaluation of the accent modification itself, we decided to present a relative decrease in CER yielded by the ASR module from the last part of our pipeline (Google Cloud Speech-to-Text and the spectrogram-based ASR trained using *LibriSpeech*).

3.4 Results

Each respective result below represents an average over ten runs of each experiment with a particular setup.

3.4.1 The results without accent modification

The ASR module (Section 2.3) was trained using spectrograms converted from *LibriSpeech train-clean-360* subset. It was evaluated using the *test-clean* dataset and achieved 15.7% CER and 19.7% WER. This model, evaluated with a 10% test subset of the spectrograms of *UME-ERJ* dataset, achieved only 46.3% CER and 56.8% WER.

The averaged result of speech recognition using spectrograms yielded by our loss network is 11.2% CER and 14.9% WER using the *LibriSpeech test-clean* dataset.

The 10% test subset of the WAV samples of the *UME-ERJ* dataset was also used to evaluate the performance of the Google Cloud Speech-to-Text API, and the result we obtained was 39.8% of CER.

The *LibriSpeech test-clean* dataset was also used for evaluating the TDNN-based ASR network we trained, and the result achieved in our test was 10% CER and 12.5% WER.

3.4.2 Impact of the autoencoder-based accent modification

After activating the autoencoder-based accent modification in our pipeline, the same test subset of the *UME-ERJ* dataset gave a result of 36.1% CER (evaluation by the spectrogram-based ASR model trained only on the *LibriSpeech* training set). Therefore, it yielded a 22% $\left(\frac{46.3\% - 36.1\%}{46.3\%}\right)$ relative improvement in terms of CER.

In the case of the Google Cloud API, we first fed the data to the autoencoder and then converted the modified spectrograms back to the sound wave format. At the end, we sent it to the cloud service and recorded the recognized text. We used the 10% subset of samples from *UME-ERJ* and after the process obtained a result of 27.3% CER, which translates to 31.4% of the relative improvement.

3.4.3 Impact of the accent modification based on the style transfer approach

For each combination of subsets tested for style and content layers, we checked the CER value on the 10% subset of spectrograms from the *UME-ERJ* dataset by feeding it into the trained autoencoder that modifies the style (Section 2.2) and feeding the respective result into the ASR (Section 2.3). The best setup gave a result of 31.7% CER. Therefore, it yielded a 32% relative improvement in terms of CER.

In the case of the cloud service evaluation, we followed the analogical process as in Section 3.4.2. The best combination of style and content layers achieved the result of 23.9% CER which means a relative improvement of 40%.

All experimental results with the best CER are presented in Tables 5 and 6. The results for experiments conducted on the style transfer approach with different style and content layers are presented in Tables 8 and 9. Tables 5, 6, 7, 8, and 9 represent the results obtained when testing the setup with both 10% subset of the *UME-ERJ* dataset and specifically prepared 100-sentences test subset of *UME-ERJ*.

In the tables, we use the following symbols:

Table 5 Results of the style modification process in cases of evaluation done using the trained CNN-RNN ASR model

	WM	A	ST
CER	46.3%	36.1%	31.7%
RI _{CER}	–	22%	32%
WER	56.8%	43.2%	34.9%
MSS	– 0.94	1.43	1.97
CER (100-sentences)	42.1%	33.2%	28.3%
RI _{CER} (100-sentences)	–	21%	24.7%
WER (100-sentences)	51.8%	39.1%	32.3%
MSS (100-sentences)	– 0.85	1.56	2.09

Table 6 Results of the style modification process in cases of evaluation done using Google cloud service

	WN	A	ST
CER	39.8%	27.3%	23.9%
RI_{CER}	–	31.4%	40%
CER (100-sentences)	34.3%	23.7%	23.5%
RI_{CER} (100-sentences)	–	30.9%	31.5%

WM —results of the experiment without the accent modification process

A —the experiment with the autoencoder-based modification process

ST —the experiment with the style transfer-based modification process

RI_{CER} —relative improvement for the CER metrics

4 Discussion

This article contains study of the audio style transfer methods used for improving accuracy of ASR which had been trained using native speech datasets and is used by non-native speakers.

We found that the style transfer methodology adapted to the speech domain yields better results than an autoencoder trained in a supervised way. We think that the reason behind it lies in the fact that we performed the training step repeatedly by sampling random samples to include, respectively, non-native and native accents, and transforming them into the spectrograms. The samples in each such pair do not have to represent the same. This observation may suggest another interesting idea that if this approach were to be extended it might be possible to create a more universal autoencoder that might convert the accent of non-native speakers from multiple into one (e.g., North American English) accent. This is going to be one of the steps for the future of our research.

Another observation is an extension of the fact that the accent modification process is not conditioned on any variables related to the speaker or speech environment. That causes the situation where during the experiment testing phase we could detect that the gender of the speaker in the sample after style transformation is different than in the one from before, while preserving the

Table 7 Results of the style modification process in cases of evaluation done using TDNN-based ASR network

	WN	A	ST
CER	38.1%	27.1%	26.1%
RI_{CER}	–	28.8%	31.4%
CER (100-sentences)	34.1%	23.1%	22.2%
RI_{CER} (100-sentences)	–	32.3%	34.8%

Table 8 Relative improvement depending on the content and style layers in cases of ASR model-based evaluation using 10% *UME-ERJ* subset

Style layers	Content layers	RI(CER)
1–10	6–12	32%
1–8	8–12	29.6%
1–10	10–12	30.1%
1–5	5–12	26.7%
1–4	4–12	15.6%

actual content of the pronounced word or sentence. However, we did not treat such cases as failed, as our primary goal was to increase the accuracy of the ASR system, which was eventually achieved. Nevertheless, it will be a next step for our team to address.

As another future step in our research, we would like to conduct more experiments, i.e., the evaluation of the style transfer approach described in Section 2.2 using more datasets. We are also planning to evaluate the process of a style transfer-based approach and autoencoder-based approach with longer sentences and samples.

We are planning to develop our idea for non-native speech recognition further and to constantly improve the quality of the designed methodology. Furthermore, additional experiments will be conducted, i.e., using multiple nationalities of non-native English speakers, as well as using different datasets including samples of languages other than English.

5 Conclusions

In this research, we explained the problem of non-native speech recognition and the reason why training ASR systems adapted for such speech may be problematic.

We described in detail the idea behind style transfer methodology and our adaptation to the speech and sound domain. We presented the method as a way to transform non-native pronunciation so that it resembles native speech to a higher extent, thus enabling the ASR system to perform better when being used by a non-native speaker.

Table 9 Relative improvement depending on the content and style layers in cases of Google cloud-based evaluation using 10% *UME-ERJ* subset

Style layers	Content layers	RI(CER)
1–10	6–12	40%
1–8	8–12	36.1%
1–10	10–12	38.3%
1–5	5–12	29.4%
1–4	4–12	21.1%

We performed initial experiments using the *UME-ERJ* dataset and tested several different pipelines for pronunciation modification. We evaluated each approach on a custom ASR trained to recognize speech from spectrograms, as well as on the publicly available Google Cloud Speech-to-Text. Our initial findings show that it is possible to augment the non-native speech samples in a way that they will be recognized with a higher accuracy by an ASR system.

We also pointed out several issues that appeared while we were training and evaluating our algorithms. This proves that there is definitely a lot of room for improvement in order to adapt the method to multiple speaker-dependent conditions and other non-native nationalities.

Abbreviations

ASR: Automatic speech recognition; WAV: Waveform Audio File; CNN: Convolutional neural network; RNN: Recurrent Neural Network; TDNN: Time Delay Neural Network; CTC: Connectionist temporal classification; CER: Character Error Rate; WER: Word Error Rate; MSS: Mean Similarity Score; API: Application Programming Interface

Acknowledgements

In our research, we are using a English Speech Database Read by Japanese Students (*UME-ERJ*), which was provided by Speech Resources Consortium at National Institute of Informatics (NII-SRC) in Tokyo.

Authors' contributions

KR designed the algorithms of style modifications from Sections 2.1 and 2.2. LW helped with data analyses and the code for transforming data between sound wave and spectrogram forms. RN and OY provided helpful advice and support during the time of designing the algorithms and experiments. All authors equally contributed to the research and this paper. The author(s) read and approved the final manuscript.

Funding

No founding sources were obtained for the purpose of this research.

Availability of data and materials

Please contact the author for data requests.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare no competing interests.

Received: 18 November 2019 Accepted: 27 January 2021

Published online: 18 February 2021

References

- W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, A. Stolcke, The Microsoft 2017 Conversational Speech Recognition System (2017). <https://arxiv.org/abs/1708.06073>
- T. T. Ping, Automatic speech recognition for non-native speakers. PhD thesis, Université Joseph-Fourier - Grenoble (2008)
- A. Metallinou, J. Cheng, in *Fifteenth Annual Conference of the International Speech Communication Association*. Using deep neural networks to improve proficiency assessment for children English language learners, (2014)
- T. Drugman, T. Dutoit, in *INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*. Glottal closure and opening instant detection from speech signals (ISCA, 2009), pp. 2891–2894. http://www.isca-speech.org/archive/interspeech_2009/i09_2891.html
- K. Livescu, J. Glass, in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, vol. 3. Lexical modeling of non-native speech for automatic speech recognition, (2000), pp. 1683–1686. <https://doi.org/10.1109/ICASSP.2000.862074>
- T. Tan, L. Besacier, in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 4. Acoustic Model Interpolation for Non-Native Speech Recognition, (2007), pp. IV-1009–IV-1012. <https://doi.org/10.1109/ICASSP.2007.367243>
- L. M. Tomokiyo, Recognizing non-native speech: characterizing and adapting to non-native usage in LVCSR. PhD thesis, Carnegie Mellon University (2001)
- O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, D. Yu, Convolutional neural networks for speech recognition. *IEEE/ACM Trans. Audio Speech Lang. Proc.* **22**(10), 1533–1545 (2014)
- N. Dave, Feature extraction methods LPC, PLP and MFCC in speech recognition. *Int. J. Adv. Res. Eng. Technol.* **1**, 1–5 (2013)
- N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, P. Ouellet, Front-end factor analysis for speaker verification. *Trans. Audio Speech Lang. Proc.* **19**(4), 788–798 (2011)
- M. Li, Han K.J., Narayanan S., Automatic speaker age and gender recognition using acoustic and prosodic level information fusion. *Comput. Speech Lang.* **27**(1), 151–167 (2013). <https://doi.org/10.1016/j.csl.2012.01.008>
- A. Graves, A. Mohamed, G. Hinton, in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 38. Speech recognition with deep recurrent neural networks, (2013), pp. 6645–6649. <https://doi.org/10.1109/ICASSP.2013.6638947>
- D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. Engel, L. Fan, C. Fougner, T. Han, A. Hannun, B. Jun, P. LeGresley, L. Lin, S. Narang, A. Ng, S. Ozair, R. Prenger, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, Y. Wang, Z. Wang, C. Wang, B. Xiao, D. Yogatama, J. Zhan, Z. Zhu, Deep speech 2: end-to-end speech recognition in English and Mandarin (2015). <https://arxiv.org/abs/1512.02595>
- G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, B. Kingsbury, Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Process. Mag.* **29**(6), 82–97 (2012). <https://doi.org/10.1109/MSP.2012.2205597>
- K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*. **abs/1409.1556** (2015)
- K. Radzikowski, L. Wang, O. Yoshie, R. Nowak, Dual supervised learning for non-native speech recognition. *EURASIP J. Audio Speech Music Process.* **2019**(3), 1–10 (2019). <https://doi.org/10.1186/s13636-018-0146-4>. <https://rdcu.be/bgUxy>
- R. Kacper, W. Le, Y. Osamu, in *Proceedings of the Conference of Institute of Electrical Engineers of Japan, Electronics and Information Systems Division*. Non-native english speaker's speech correction, based on domain focused document, (2016)
- R. Kacper, W. Le, Y. Osamu, in *Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services (iiWAS '16)*. Non-native english speakers' speech correction, based on domain focused document (ACM, New York, 2016), pp. 276–281
- R. Kacper, W. Le, Y. Osamu, in *Proceedings of the conference of institute of electrical engineers of japan, electronics and information systems division*. Non-native speech recognition using characteristic speech features, with respect to nationality, (2017)
- L. A. Gatys, A. S. Ecker, M. Bethge, A Neural Algorithm of Artistic Style (2015). <https://arxiv.org/abs/1508.06576>
- J. Johnson, A. Alahi, L. Fei-Fei, Perceptual losses for real-time style transfer and super-resolution (2016). <https://arxiv.org/abs/1603.08155>
- E. Grinstein, N. Q. K. Duong, A. Ozerov, P. Pérez, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Audio Style Transfer (IEEE, 2018). <http://dx.doi.org/10.1109/icassp.2018.8461711>
- P. Verma, J. O. Smith, neural style transfer for audio spectrograms (2018). <https://arxiv.org/abs/1801.01589>
- K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
- F. Chollet, et al., Keras. GitHub (2015). <https://github.com/fchollet/keras>

26. A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks (Association for Computing Machinery, New York, 2006), pp. 369–376. <https://doi.org/10.1145/1143844.1143891>
27. B. Hixon, E. Schneider, S. Epstein, in *INTERSPEECH*. Phonemic Similarity Metrics to Compare Pronunciation Methods, (2011)

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)
