Multi-Task Variational Auto-Encoder for Source-Filter Decomposition of Speech
Project Report

Anne Baril, Remi Bourgerie, Kobe Moerman & Zakaria Nassreddine

Superviser: Jonas Beskow

July 7, 2022

Contents

Abstract

State-of-the-art end-to-end speech synthesis systems provide output in the form of spectrograms that do not allow manipulation of either one of the underlying source and filter spectra in total isolation of the other. This hinders the prospects of more flexible and sophisticated post-processing of the output to manipulate features that are characteristic of one of the two latent spectra. In this work, we experiment with a novel approach for source-filter decomposition that uses Variational Auto-Encoders (VAE) as building blocks in a more complex architecture that takes an input spectrogram and decomposes it into two latent spectra such that they sum up to the input. To incentivise our model to learn the two representations that are of interest to us, we incorporate multi-task learning mechanisms that push it to, on top of reconstructing the input, also predict explicit parameters that are specific to the source and filter spectra. While our current results aren't satisfactory enough to approve our architecture as a reliable system for source-filter decomposition of speech, our preliminary findings suggest that such an approach is worthy of further investigation and might yield better out-turns with more exhaustive hyper-parameter fine-tuning and more advanced data pre-processing.

1 Introduction

According to the Source-Filter model, speech production can be deconstructed into a two-phase process that involves the generation of a sound source, that is a waveform from the glottis, which is then filtered by the resonant properties defining the transfer function of the vocal tract. This is modeled as an element-wise multiplication of the two spectra in the spectral domain, so the source $X(k)$ and the filter $Y(k)$. Equivalently we have the summation in the log-spectral domain.

$$Y(k) = X(k)H(k) \tag{1}$$

$$log(Y(k)) = log(X(k)) + log(H(k)) \tag{2}$$

Thus, under this model, a given spectrogram can be seen as a compound of the latent source and filter spectrograms. The aim of this project is then to build an auto-encoder that will separate these two components into distinct spectrograms. Typically, current established end-to-end systems output spectrograms that are the compact convoluted biproduct of the two spectra of interest. However, some tasks would benefit from adjustable components during post-processing for a more flexible speech synthesis; that is singularly modify the pitch through the source filter, or experimenting with substituting various speech filters with one another.

The aim of this report is first to introduce methods to isolate both the source and filter spectra all while experimenting the usability of such models. The results are then discussed apropos the outcome along with the scientific gain from similar methods.

2 Related work

There exists a wide range of research in machine learning with respect to speech technology. Some aim at decomposing voice into source and glottal flow for speech analysis purposes, based on filtering techniques [4]. Others use auto-encoders to classify audio scenes [1].

Auto-encoders represent standard techniques for dimensionality reduction [6]. However, their decoding process is often limited by the discrete nature of the latent space which makes reconstruction difficult for large scale data sets. Variational Auto-Encoders (VAE) help cope with this problem by introducing a latent space based on a probability distribution [3]. Auto-encoders are often used in relation with specific applications which include additional conditions for low-dimension representations (for instance formant and fundamental frequencies in the case of speech analysis). To address this purpose, mutitask learning can be used as a technique for managing learning based on multiple function optimization [5].

3 Method

3.1 Data Framework

This study uses the `CMU_ARTIC` dataset. It was put together at the Language Technologies Institute at CMU as a phonetically balanced English speaking database for unit selection speech synthesis. A total of 1150 wav-files were recorded in a sound proof booth at 32 `KHz` of which both male and female speakers were taken into account.

3.1.1 Corpus

The sentences used in each utterance originate from the *Gutenberg Project*. The CMU then refined the dataset to clear utterances, that is which are easily read by Native English speakers. Some examples are as follow:

> `'For the twentieth time that evening the two men shook hands'`
> `'I'm playing a single hand in what looks like a losing game'`
> `'It was my reports from the north which chiefly induced people to buy'`

Even though the CMU dataset is thorough, its quality can impact the result of material depending on the sought after purpose. The dataset was developed in order to "be good for reading short stories, bad for reading poems, and adequate for dialog systems". As this project has an experimental nature — determining whether the source and filter spectra are dissociable — having recordings issuing from studio conditions is beneficial. However, in order to have a more practical use case, some adjustments would need to be put into place with respect to the diversity and quality of the dataset.

3.1.2 Transformation

The dataset originally comes in the format of waveforms. However, it is extremely difficult to use raw audio signals as input to a machine learning model. In order to simplify such task, it is sensible to visualise the process. Indeed, sounds can be characterised as varying pressure waves with respect to time. Some pre-defined library, such as `librosa`, will help us convert an audio signal into a visual representation.

As the audio recordings vary in length, it is important to split the array representation into specific segments. This way the data will have equal shapes. For this project we set the split threshold to one second, where any remaining segment smaller than the threshold is ignored. The values are then normalised in order to comply with the model training process. This step however heavily impacts the reconstitution to waveform files, and as a result the possible experimental evaluation.

3.2 Model Architecture

Up to this point the data has been transformed into tensors with acceptable training and testing splits. It is now pertinent to expand on the methods used to build a model that will handle the discussed input.

3.2.1 Tools

Machine learning is a very complex field of study that requires many years of practice. With the help of machine learning frameworks, such as TensorFlow, the process of acquiring data, training models, and obtaining predictions becomes far more accessible.

TensorFlow is an open source library for numerical computation used by neural networks. It uses Python for building a convenient front-end API, and executes its applications in C++. As such it allows developers to create architectures that describe how the data is modified, without having to implement these algorithms themselves.

Recently, TensorFlow has adopted Keras as the high-level API. The main reason to use Keras arises from its ease of learning and model building. The API was "designed for human beings, not machines", and "follows best practices for reducing cognitive load". There already exist a wide range of predefined modules – such as neural layers, cost functions, optimizers, activation functions, etc – in order to create unique models.

3.2.2 Autoencoders

An autoencoder is a neural network which aims at reproducing its given input. It is comprised by a hidden layers, h, that defines the result of transformations applied to the original image. Generally, the network is characterised by two parts: an encoder function $h = f(x)$ that modifies the input to a compact bottleneck, and a decoder function $r = g(h)$ that reconstructs this latent space [2]. This basic network is depicted in Figure 1. They are designed to only carry out approximate reproductions of an input that relates to the training data. This way, the model targets specific aspects of the input and, as a result, learns useful features of the data.

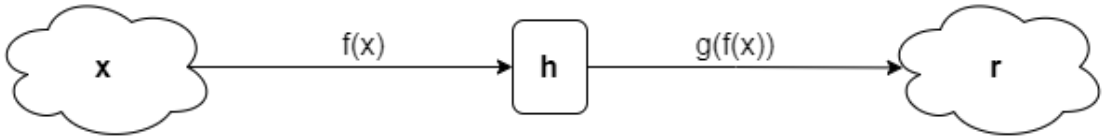


Figure 1: General structure of an autoencoder

CNNs are similar to ordinary neural networks with the difference that the weighted neurons are selected by a sliding convolution filter. Each neuron receives an input from which it learns local patterns by applying a series of non-linear transformations. This architecture maps a small convolution filter on-top of an image input, leading to a reduction of parameters in the network. Two key characteristics arise from this point. First, the patterns they discovered through the training are translation invariant, contrary to a densely connected network. Second, they can learn increasingly complex and abstract patterns through a spatial hierarchy.

Convolutional Autoencoders (CAE) are a type of CNN with the main difference in their learning procedure being unsupervised, meaning that the filters extract features with the sole purpose of reconstructing the input. Likewise, the parameters required to produce a representative activation map stay constant, regardless of the input size. For this reason they prove to be proficient with high-dimensional data. This gentle data extraction is accomplished through a sequence of steps, including the convolution layer, and the reLu layer.

Similarly to CAE, a variational autoencoder (VAE) consists from both an encoder and a decoder which is trained in order to minimise the error between the input and the reconstructed output. However, to include regularisation to the respective latent space, the

encoding to the bottle-neck differs slightly. It not longer maps the input to a single point but instead over a distribution of values.

3.2.3 Architecture

As mentioned previously, the project aims at dissociating the source and filter spectra. In order to accomplish this task, the model will consist of two separate VAEs that influence one another to achieve the optimal reconstruction; call these V_{f_0} and V_{frt}. The main difference between them is the regularisation aspect, one will analyse the f_0 parameter from the source whereas the other will analyse the formants from the filters. The log-spectral domains from the decoder of V_{f_0} and V_{frt} are then combined in order to also minimise reconstruction loss of the original input.

The idea to constrain a VAE with regards to a specific feature is to impact the learning experience of the model with a fully connected network that itself learns a representation of the feature (Fig. 2). The model architecture is two-fold; First we have a simple VAE which attempts to reconstruct the input spectrogram through an `MSE` and `KL` loss function. To push this one in dissociating a feature, we include a fully connected network to the latent space. This layer is reduced to the feature's shape and compared to the ground value through an `MSE`. By doing so, the model is forced to learn a latent space that is analogous to either the source or filter spectra.

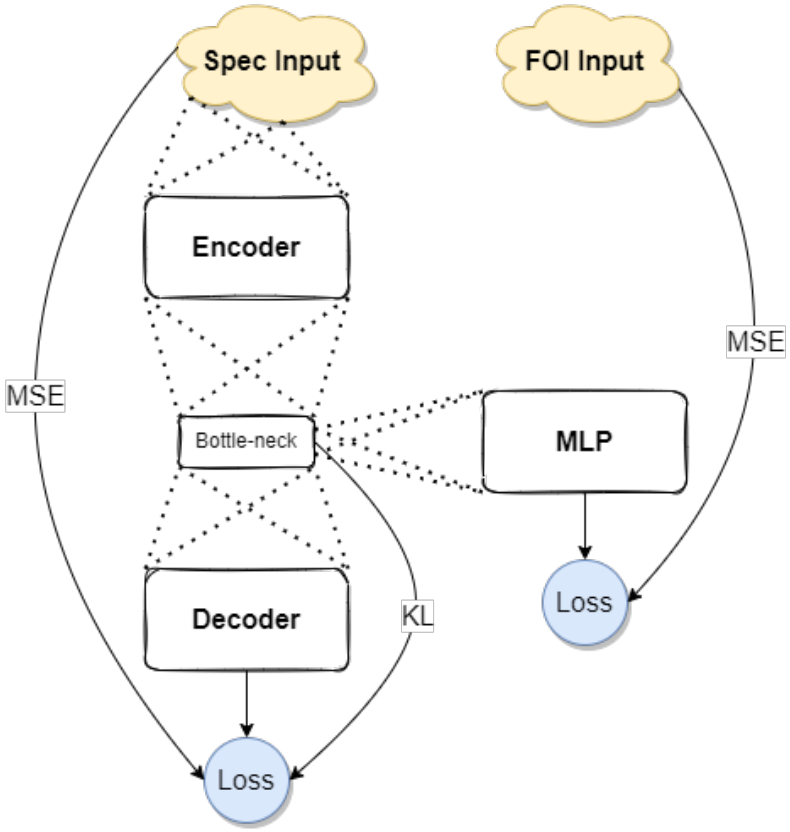


Figure 2: Variational Autoencoder where the latent space is pushed towards a representation analogous to the feature.

The following step is to create a model that encompasses V_{f_0} and V_{frt}. Both sub-models will carry on their task in parallel as to have their respective outputs combined to a log representation (Fig. 3). By doing so, this project hopes that V_{f_0} learns a spectrogram representation exclusive to the source filter and V_{frt} to the filter spectra.

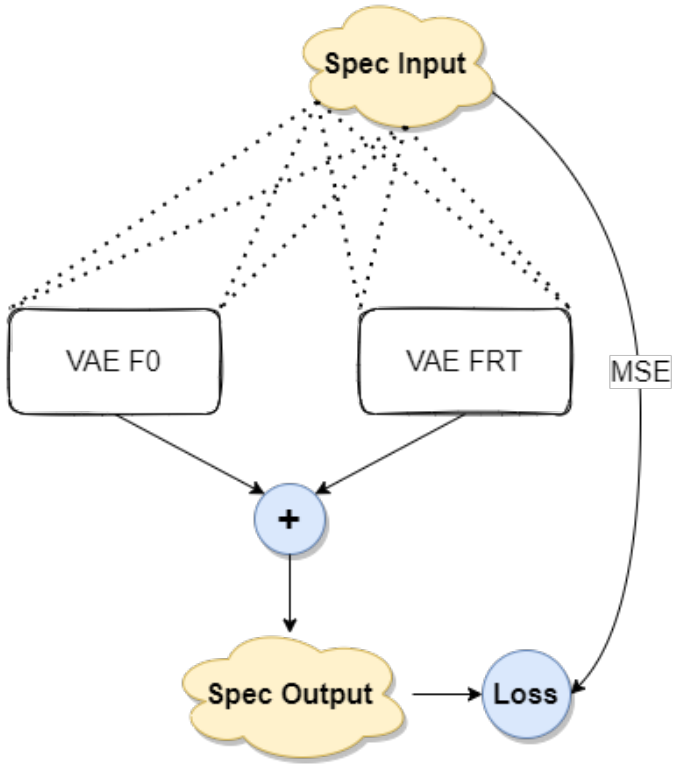


Figure 3: Parallel Model that combines the reconstruction of two distinct Autoencoders

3.3 Evaluation

3.3.1 Loss Function

The key concept behind the success of an autoencoder is the loss function. This one incites the model towards an optimal reconstruction, that is the output would ideally be identical to the input. Accordingly, the loss function is simply the mean squared error between the input and output.

$$loss_{\text{ae}} = \|x - g(f(x))\|^2 = \|x - \hat{x}\|^2 \tag{3}$$

To have a useful learning practice, the model also attempts to decrease the dimensionality of the latent space. This principle leads to a conflict, as reducing to a narrow latent space often leads to a significant loss of information. A balance between these conditions is therefore crucial.

In addition to this reconstruction term, the VAE also expects a regularisation term on the latent layer which encourages the distribution of the latent space to a standard normal distribution. This is the Kulback-Leibler (KL) divergence between the latent distribution and a standard Gaussian. The loss function then becomes the sum of these two terms.

$$loss_{\text{vae}} = \|x - \hat{x}\|^2 + KL[N(\mu, \sigma), N(0, 1)] \tag{4}$$

Since both the F_0 and formant parameters need to be taken into consideration with their respective models, the loss equation needs to be updated. By doing so we expect the V_{f_0} and V_{frt} models to train a reconstruction where the latent space is representative of the desired feature.

$$loss_{\text{custom}} = \alpha\|x - \hat{x}\|^2 + \beta\|\gamma - \hat{\gamma}\|^2 + KL \tag{5}$$

where γ is the feature ground truth extracted from the input data, $\hat{\gamma}$ is the expected value from the fully connected network and α, β are regularizing weights. Thus, in addition to the reconstruction process, the latent space is also further modified to the feature representation in order to impact the learning experience of the respective VAE.

The final loss function in equation (5) is then further influenced by the global loss function of the parallelised model. As mentioned previously, this one takes the log addition of both model outputs to then compare the result to the log representation of the input data. Through back-propagation, this loss function impacts the weight update of both V_{f_0} and V_{frt}.

3.3.2 Experimental Results

An audio file is comprised of many factors which have varying responsibility in the clarity of the resulting speech. Therefore, when reconstituting the audio file from the source and filter spectra, it can be difficult to determine the accuracy of the achieved latent representation. Indeed, it is simple to compare the spectrogram between the input and output, but this one is not guaranteed to be objectively coherent. Further, it is not clear whether interchanging both the F_0 and formant spectra in an audio sample leads to relevant results.

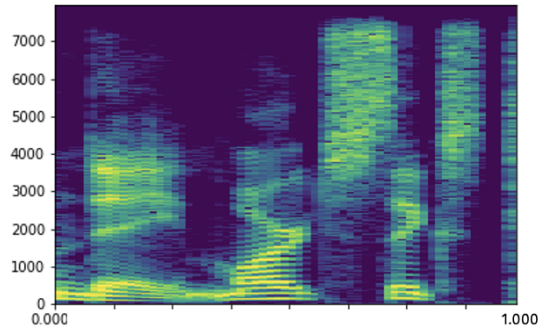
The evaluation of research following this project is essentially two-fold. First, the resulting spectra from models V_1 and V_2 are summed according to equation (2). The spectrogram can then be translated to an audio file. As such, an evaluation can be carried out in a subjective manner since the reconstruction accuracy can then be judged. Then, it can also be of interest to collect differing samples and interchange the source and filter spectra. Similarly to the aforementioned, the spectra are summed and converted to an audio file. It is difficult for a computer program to evaluate similar schemes, therefore a subjective take on the results is necessary. This step does not have any benefits to the performance of the model but will be beneficial from an innovative stand-point.

4 Results and evaluation

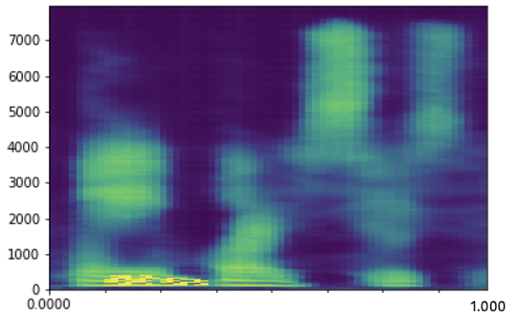
4.1 Initial results

After the training, the model returns the test set predictions from both V_{f_0} and V_{frt}. These are responsible for the source and filter features respectively, and when summed together represents the input spectrogram.

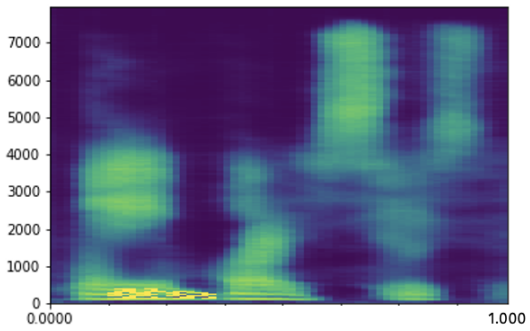
The first results were rather unsatisfactory. This enlightened us to the complexity and challenge in training such multi-task model; namely how to find a good balance between losses of varying magnitudes. In this case, the reconstruction loss for the source feature was more significant than the one for the filter feature. Consequently, the V_{frt} was not active in the learning experience and essentially reduced to a constant whereas the V_{f_0} was dominant (Fig. 4).



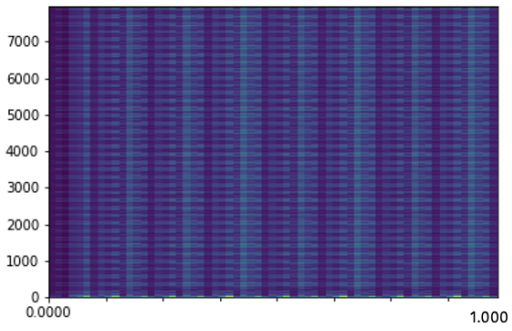
(a) Input Spectrogram



(b) Reconstructed Spectrogram



(c) Source Spectrogram



(d) Filter Spectrogram

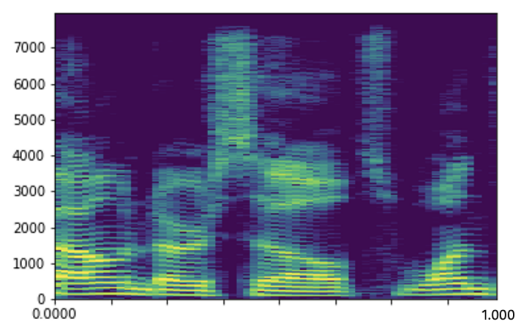
Figure 4: Multi-task autoencoder with varying sub-model loss magnitudes.

4.2 Progressive results

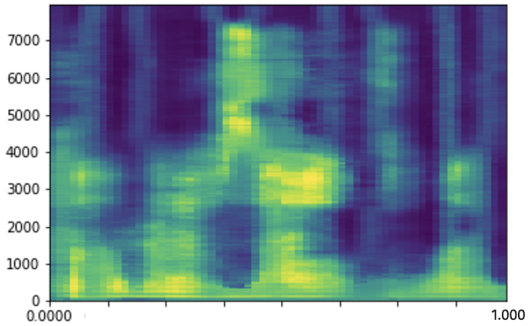
With this goal in mind, we introduced weights to the feature loss of both V_{f_0} and V_{frt} to draw them to similar magnitudes. By experimenting with varying values, the model achieved more concrete results (Fig. 5). Although the decomposition remains average, we can see that each VAE is learning differing representations. These are promising first results given that this study is exploratory of nature.

To evaluate the results of our work, we converted the spectrograms back to audio files and we made a qualitative assessment of our decomposition. A significant problem was

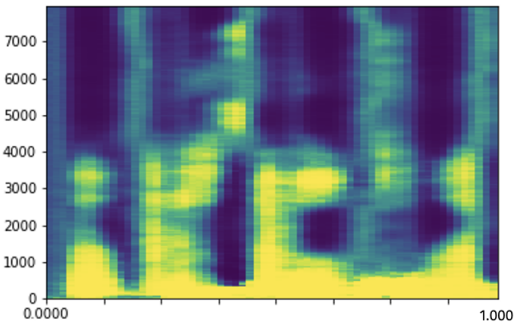
during the conversion from spectrogram to an audio file. Even for the input data, after this one has been normalised, would lead to a poor quality with a lot of noise which made it unintelligible. Thus, the reconstructed audio is also plagued by noise. Further evaluation also involved human study of the latent spectrograms.



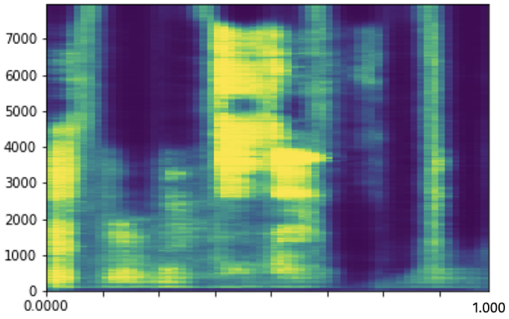
(a) Logarithmic Input Spectrogram



(b) Logarithmic Reconstructed Spectrogram



(c) Source spectrogram



(d) Filter spectrogram

Figure 5: Multi-task autoencoder with consistent sub-model loss magnitudes

5 Discussion

The process of generating input spectrograms has proven quite troublesome. The back and forth spectrogram/audio transformation turned out to have significant information loss which ultimately produced data of poor quality, thus hindering our network's capacity to reconstruct natural speech samples. Our current findings suggest that more exhaustive hyper-parameter fine-tuning should come in handy to improve both the reconstruction and the decomposition. Furthermore, the Griffin Lim algorithm that was used for audio conversion was a rather convenient option for us given the scope and the time limits of the project, with it being relatively simple to use and integrate in our data processing pipeline. But its ease of use comes at a performance cost, and we would suggest resorting to more robust and accurate Neural Vocoders for this task moving forward.

Ultimately, if we observe our reconstructed spectrogram 5b, the important regularity and blur among all images suggest that the network is not complex enough to recover most of the details and features in the learned images. The very structure of our network could be enhanced for instance by using a latent space greater than 64 units currently used, or stacking more convolutional layers.

Another alteration would be to cut the samples to fractions that are shorter than a full second so that each spectrogram maps to one fundamental frequency.

It is clear from figure 5 that our architecture is not able to decompose according to the source/filter model. Indeed, it does not capture the pitch contour. However, and this is pretty encouraging, as can be seen on this same figure, the two VAE do not learn the same features. Therefore, by considering more accurate hyper-parameters (increasing the latent dimensions, taking shortest audio recordings with an overlapping window for instance, to avoid having different fundamental frequencies over a segmented audio as on figure 6), one could expect better results.

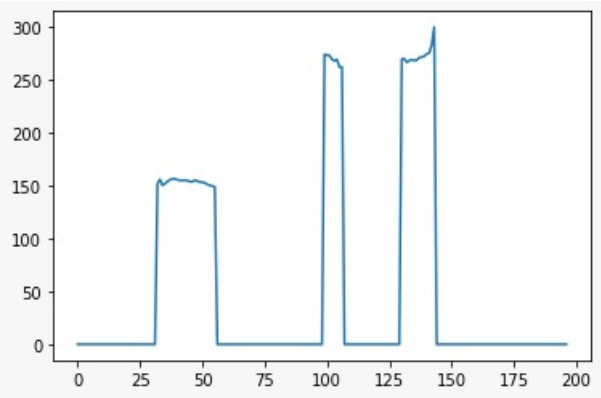


Figure 6: Evolution of the fundamental frequency w.r.t time

6 Conclusion

Our work shows that it is possible to train parallel VAEs for different tasks. However, the respective behaviour are different from the expected source-filter model, and as such, the model cannot be validated. Furthermore, the audio reconstruction performances are mitigated since some similarities of the spectrograms can be observed, but the recovered audios are not exploitable. Some optimizations of the results through parameters tweaking of both pre-processing of data and learning can be achieved for further improvements of the method.

References

[1] Shahin Amiriparian et al. *Sequence to sequence autoencoders for unsupervised representation learning from audio*. Universität Augsburg, 2017.

[2] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[3] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].

[4] Erfan Loweimi, Jon Barker, and Thomas Hain. "Source-filter separation of speech signal in the phase domain". In: *16th annual conference of the international speech communication association (interspeech 2015), VOLS 1-5*. ISCA. 2015, pp. 598–602.

[5] Ozan Sener and Vladlen Koltun. "Multi-task learning as multi-objective optimization". In: *Advances in neural information processing systems* 31 (2018).

[6] Wei Wang et al. "Generalized autoencoder: A neural network framework for dimensionality reduction". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2014, pp. 490–497.