# LPC based voice anonimization

**Arkadiy Aliev** [1] **Vsevolod Ivanov** [1] **Yuriy Ivanov** [1] **Mikhail Mishustin** [1] **Andrey Safronov** [1]

## Abstract

Anonymization has the goal of manipulating speech signals in order to degrade the reliability of automatic approaches to speaker recognition, while preserving other aspects of speech, such as those relating to intelligibility and naturalness. It's important to note that voice anonymization involves altering not only the speaker's voice but also linguistic content, extralinguistic traits, and background sounds that might reveal the speaker's identity. At the moment, there are many methods that cope with this task and use different approaches for this. Within the framework of this project, we will consider one of the latest methods of speech anonymization (Tomashenko et al., 2022), and also compare it with other existing models for the quality of anonymization and preservation of the meaning of speech.

**Github repo:** https://github.com/ArkadiyAliev/LPC-based-voice-anonymisation
**Presentation file:** https://github.com/ArkadiyAliev/LPC-based-voice-anonymisation/blob/main/LPC-5-Gosling-gang.pdf

## 1. Introduction

Voice-activated interfaces have become increasingly popular due to their convenience and simplicity. The proliferation of applications utilizing speech data for user authentication raises concerns about the security and privacy of this information.

Speaker anonymization, a key aspect of privacy protection in speech data, involves removing speaker-specific biometric information while preserving the linguistic content of the speech.

Some of existing approaches to privacy protection in speech processing systems have primarily focused on extracting privacy-preserving features or applying encryption techniques to secure speech data. However, these methods often lack transparency, making it challenging for users to verify the effectiveness of privacy safeguards. This lack of user assurance may deter individuals from sharing their data for model improvement purposes.

Other approaches, like (Tomashenko et al., 2022) proposes leveraging voice conversion techniques for speaker anonymization, enabling the transformation of a speaker's identity while maintaining the original speech content. By adopting voice conversion methods, the study aims to enhance privacy protection in speech processing systems without compromising recognition accuracy.

## 2. Related work

Text-to-Speech (TTS) and Speech-to-Text (STT) technologies are key in digital communication, bridging human language and computer processing. TTS converts text into spoken words for applications like virtual assistants and accessibility tools, while STT, or automatic speech recognition (ASR), turns spoken language into text for voice commands and transcription. Alongside, speech anonymization is vital for protecting speaker identity in audio data by modifying voice characteristics to hinder speaker identification, maintaining privacy without losing speech clarity. These technologies significantly enhance user interfaces, ensure privacy, and improve digital content accessibility.

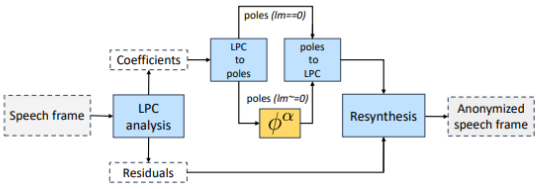
In the study (Patino et al., 2021) propose a new, simple method for voice anonymization. This method doesn't require lots of data to work and very lightweight to compute. It uses the McAdams coefficient to change the sound characteristics of speech in a way that prevents voice recognition systems from identifying the speaker. Their tests, conducted with VoicePrivacy 2020 standards, show that this method is effective at making speakers anonymous while keeping the speech clear and natural sounding, even when attackers have some knowledge. This approach stands out because it's efficient and straightforward, providing a fresh solution for protecting speaker privacy.

In contrast to other methods, method in (Tomashenko et al., 2022), shown in Figure 1, does not require any training data and is based upon simple signal processing techniques. It

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Mikhail Mishustin <Mikhail.Mishustin@skoltech.ru>.

*Final Projects of the Machine Learning 2023 Course*, Skoltech, Moscow, Russian Federation, 2023.

employs the McAdams coefficient to achieve anonymisation by shifting the pole positions derived from linear predictive coding (LPC) analysis of speech signals.




Moving to approaches using neural networks, we will start with the paper (Yoo et al., 2020), where authors explore an innovative approach leveraging Generative Adversarial Networks (GANs) composed of Variational Autoencoders (VAEs) (Yoo et al., 2020). Authors created a complex method to anonymize speaker identity without undermining the linguistic content integral for speech recognition systems. Central to their approach is the modification of user embeddings within the encoder section of the VAE. By replacing an actual user's embedding with the average of multiple users' embeddings, the technique effectively obscures the speaker's identity. This methodological choice is underscored by the significant reduction in speaker identification accuracy to between 0.1 to 9.2%, evidencing successful anonymization. Concurrently, the approach preserves speech recognition accuracy, maintaining it within the range of 78.2 to 81.3%.

Recent advancements in voice anonymization and speech synthesis technologies have led to the development of innovative methods aimed at enhancing privacy and realism in generated speech. One such method, discussed in the work of (Muckenhirn et al., 2018), introduces a novel approach for speaker verification through direct modeling of raw speech signals using Convolutional Neural Networks (CNNs). This technique demonstrates competitive performance against state-of-the-art systems (at the time of writing) by emphasizing low-frequency information and the fundamental frequency for speaker discrimination (Muckenhirn et al., 2018).

Another relevant paper is (Jia et al., 2019), which presents integration of Tacotron 2 and WaveNet models presents a more modern approach for text-to-speech synthesis. This paper introduces a novel approach in text-to-speech (TTS) synthesis by leveraging transfer learning from speaker verification to enable multispeaker TTS synthesis. The method uses a speaker encoder model to extract speaker embeddings from speech samples, which are then fed into a Tacotron-based TTS model to generate speech in the voice of the target speaker, even if that speaker was not seen during training. This enables high-quality, speaker-agnostic TTS capabilities, significantly expanding the scope of application by creating more versatile and personalized speech generation applications compared to other approaches.

### 2.1. Comparative Analysis

- **Complexity**: The GANs with VAEs and the integration of Tacotron 2 and WaveNet represent the more complex end of the spectrum, leveraging deep learning for nuanced control over speech characteristics. The McAdams Coefficient-based approach stands out for its simplicity and ease of implementation.
- **Data Dependency**: The signal processing techniques require minimal data, making them accessible but potentially less robust. In contrast, neural network-based approaches, especially those involving GANs and Tacotron 2 with WaveNet, rely heavily on extensive datasets for training.
- **Computational Requirements**: Approaches utilizing deep learning (GANs with VAEs, CNNs, Tacotron 2, and WaveNet) necessitate significant computational resources, whereas the McAdams Coefficient-based method is noted for its computational efficiency.
- **Anonymization Effectiveness**: While all methods aim to effectively anonymize speech, neural network approaches offer more sophisticated ways to modify speech characteristics, potentially providing stronger anonymization capabilities at the cost of increased complexity and data requirements. Although it will be shown later in our work that it is not possible to unambiguously conclude that the

## 3. Algorithms and Models

### 3.1. LPC

Let $x_t$ be the amplitude of our signal at a given instant $t$. According to the source-filter model, it's generated by a source signal $e$ going through a resonant filter $h$.

$$x_t = (h * e)_t$$

The $*$ denotes the convolution operator. The model further assumes that the current signal also depends on the past $p$ samples, that is $x_{t-1}, \ldots, x_{t-p}$, and that the source is constant, so effectively:

$$x_t = \sum_{k=1}^{p} a_k x_{t-k} + e_t$$

We then have $n-1$ equations (one for each sample, except the first), and we have to determine the $p$ coefficients $a = [a_1, \ldots, a_p]^T$ and :

$$
\begin{aligned}
x_1 a_1 &\quad &+e_2 &= x_2 \\
x_2 a_1 + x_1 a_2 &\quad &+e_3 &= x_3 \\
&\vdots \\
x_p a_1 + x_{p-1} a_2 + \ldots + x_1 a_p &\quad &+e_{p+1} &= x_{p+1} \\
&\vdots \\
x_{n-1} a_1 + x_{p-1} a_2 + \ldots + x_1 a_{n-1} &\quad &+e_n &= x_n
\end{aligned}
$$

The approach taken in is to ignore the errors and solve for **a**, but trying to minimize the error. The error is then $e$. More precisely, we define the matrix $X$ where the $i$-th row is:

$$X_i = [x_i, x_{i-1}, \ldots, x_{i-p+1}]$$

We assume that $x_i = 0$ if $i \leq 0$ . We define as the column vector: $[x_1, \ldots, x_n]^T$, and then we solve the linear system: $Xa = b$ for $a$, minimizing the square of the error.

One way of constructing the matrix $X$ is to generate a vector $[x_n, \ldots, x_1, 0, \ldots, 0]$ and then assign the last entries to the first row ($[x_1, 0, \ldots, 0]$), then shift the window by one and assign to the second row ($[x_2, x_1, 0, \ldots, 0]$), and so on.

We can then use np.linalg.lstsq() to solve and find $a$.

**Algorithm 1** Solve_LPC

**Input:** $x, p$
$b := [x_1, \ldots, x_n]$
$X := make\_matrix(x, p)$
$a := solve\_linear\_equation(X, b)$
$e := b - Xa$
$g := Var(e)$
**return** $[a, g]$

The vector $e$ is assumed to be samples from a white noise source. This can be modeled by a normal distribution with zero mean and the same variance $g = \sigma^2$, so this is the only parameter we need to store in our model.

We can now define the LPC algorithm, by first splitting the original signal into chunks then solving the model for each chunk:

**Algorithm 2** LPC_encode

**Input:** $x$, $p$, $w$
$B \in \mathbb{R}^{nb \times nw} := create\_blocks(x, w)$
$A \in \mathbb{R}^{p \times nb} := [0, \ldots, 0]$
$G \in \mathbb{R}^{1 \times nb} := [0, \ldots, 0]$
**for** $i = 0$ **to** $nb$ **do**
  $[a, g] = Solve\_LPC(B[i, :], p)$
  $A[:, i] = a$
  $G[:, i] = g$
**end for**
**return** $[A, G]$

Decoding a LPC model consists in simulating a Source-Filter model: we first generate a source signal (white noise) and then apply a filter corresponding to the coefficients. For the white noise, we get samples from a normal distribution. The function randn() implements a normal distribution with mean 0 and variance 1. To get a variance of $g$, we need to multiply by $\sqrt{g}$, since

$$\mathcal{N}(\mu, \sigma^2) = \mathcal{N}(0, 1) \cdot \sigma + \nu$$

Now that we can decode the signal from a given single LPC model, we can generate all OLA blocks ($\hat{B}$) and add them back up to obtain the full signal ($\hat{x}$)

**Algorithm 3** LPC_decode

**Input:** $A \in \mathbb{R}^{p \times \text{-}}$, $G \in \mathbb{R}^{ne, n}$, $w$
$\hat{B} \in \mathbb{R}^{n, nw} = \mathbf{0}$
**for** $i = 0$ **to** $n$ **do**
  $\hat{B}[i, :] = run\_source\_filter(A[:, i], G[:, i], mw)$
**end for**
$\hat{x} = add\_blocks(\hat{B})$
**return** $\hat{x}$

### 3.2. McAdams coefficient LPC based voice anonymisation

The main idea of this method is to use LPC for encoding, transform obtained LPC coefficients and use them for decoding.

**Algorithm 4** McAdams coefficient LPC based voice anonymisation

**Input:** $x$, $p$, $w$, $\alpha$
$A =$ LPC_encode$(x, p, w)$
$\tilde{A}$ - matrix with the same shape as $A$
**for** $a \in A$ **do**
  $p :=$ polynomial with coefficients from vector $a$.
  $r_1, \ldots, r_k :=$ roots of $p$
  $\tilde{r}_i := |r_i| e^{i \arg(r_i)^\alpha}$
  $\tilde{p} :=$ polynomial with roots $\tilde{r}_1, \ldots, \tilde{r}_k$
  $\tilde{a}_0, \ldots, \tilde{a}_k$ - coefficients of $\tilde{p}$
  $\tilde{A}[i] := (Re(\tilde{a}_0), \ldots, Re(\tilde{a}_k))$
**end for**
Output = LPC_decode$(\tilde{A})$
**return** Output

### 3.3. NN-based methods

Initially, user embeddings were generated using the (Eren & The Coqui TTS Team, 2021) model. Subsequently, the text content was extracted from the original speech using the (Team, 2021) text-to-speech model. Following this, a process involving the (Eren & The Coqui TTS Team,

2021) speech-to-text models was employed. The resulting text, along with the substituted user embeddings, were fed into the model, with a detailed explanation provided on the embedding substitution process. Ultimately, the output of this methodology yielded anonymized speech data.

Now we will describe in more detail the technique of embedding substitution.

We have used several techniques in our work:

- the average of embedding of all users;
- Random noise from a normal distribution with meat is equal to 0, with a variance equal to the variance of the embeddings;
- Random noise from a normal distribution with mean is equal mean of user's embeddings, with a variance equal to the variance of the embeddings;
- embedding a random speaker;
- Weighted embedding of the average from all users and the original speaker with different coefficients, we selected coefficients 0.3, 0.5 and 0.7.

## 4. Experiments and Result

### 4.1. Datasets

We used the Common Voice dataset (Ardila et al., 2020). This dataset is a massively-multilingual collection of transcribed speech intended for speech technology research and development. Common Voice is designed for Automatic Speech Recognition purposes but can be useful in other domains (e.g. language identification).

### 4.2. Dataset split

When calculating the metrics, we choose the first 100 texts from CommonVoice dataset (Ardila et al., 2020).

### 4.3. Hyperparameters

We went through several $\alpha$ parameters in the McAdams algorithm, and got the following conclusion: when $\alpha$ is greater or less than $0.8$, white noise is obtained.

### 4.4. Metrics

Using the Word error rate (WER) metric, we compare the quality of text recognition obtained after anonymization.

1) WER (Content quality)

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

- $S$ is the number of substitutions
- $D$ is the number of deletions
- $I$ is the number of insertions
- $C$ is the number of correct words
- $N$ is the number of words in the reference ($N = S + D + C$)

The $CosineSimilarity$ metric compares the quality of anonymization.

2) Cosine Similarity (difference between audios' embeddings)

$$CosineSimilarity(u, v) = \frac{\langle u, v \rangle}{\|u\|\|v\|}.$$

The lower the $CosineSimilarity$, the better the anonymization.

### 4.5. Results

After receiving the anonymized speech, we do the following: Reconstruct text from these files using whisper-large-v3 (Radford et al., 2022).

And to compare speech in WER, we use convgru embedder (Payer et al., 2018).

*Table 1.* Comparison time of inference

| | TIME | MEMORY |
|---|---|---|
| LPC | 288MS | 144 MIB |
| NN-BASED | 1.53S | 2126 MIB (GPU) |

*Table 2.* Comparison of metrics (Anonymyzation performance and Text recognition performance)

| | COSINESIMLARITY | WER |
|---|---|---|
| LPC | 0.734 | **0.198** |
| NN: NO ANON. | 0.685 | 0.277 |
| NN: MEAN EMB. | 0.464 | 0.270 |
| NN: NOISE EMB. | 0.628 | 0.256 |
| NN: NORMAL EMB. | 0.359 | 0.272 |
| NN: RAND. SPEAKER EMB. | **0.345** | 0.279 |
| NN: LIN.COMB 0.3 | 0.647 | 0.271 |
| NN: LIN.COMB 0.5 | 0.616 | 0.265 |
| NN: LIN.COMB 0.7 | 0.547 | 0.266 |

- LPC works much faster than the NN-based method.
- LPC anonymization is the worst.

- When using random speaker embeddings, anonymization is better than in other approaches, but WER is at the same level with other NN-based methods.

## 5. Conclusion

In our study, we delve into the intricate balance between the quality and comprehensibility of anonymized speech. Our findings reveal that while both LPC and NN-based algorithms offer viable options for anonymization, each method presents distinct advantages. Notably, LPC demonstrates significantly faster processing speeds compared to NN-based techniques, albeit at the cost of lower anonymization quality. However, the LPC approach shines in efficiency, showcasing advantages in time and memory utilization during inference.

## References

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., and Weber, G. Common voice: A massively-multilingual speech corpus, 2020.

Eren, G. and The Coqui TTS Team. Coqui TTS, January 2021. URL https://github.com/coqui-ai/TTS.

Jia, Y., Zhang, Y., Weiss, R. J., Wang, Q., Shen, J., Ren, F., Chen, Z., Nguyen, P., Pang, R., Moreno, I. L., and Wu, Y. Transfer learning from speaker verification to multispeaker text-to-speech synthesis, 2019.

Muckenhirn, H., Magimai.-Doss, M., and Marcell, S. Towards directly modeling raw speech signal for speaker verification using cnns. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4884–4888, 2018. doi: 10.1109/ICASSP.2018.8462165.

Patino, J., Tomashenko, N., Todisco, M., Nautsch, A., and Evans, N. Speaker anonymisation using the mcadams coefficient, 2021.

Payer, C., Štern, D., Neff, T., Bischof, H., and Urschler, M. Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks, 2018.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust speech recognition via large-scale weak supervision, 2022. URL https://arxiv.org/abs/2212.04356.

Team, S. Silero models: pre-trained enterprise-grade stt / tts models and benchmarks. https://github.com/snakers4/silero-models, 2021.

Tomashenko, N., Wang, X., Miao, X., Nourtel, H., Champion, P., Todisco, M., Vincent, E., Evans, N., Yamagishi, J., and Bonastre, J.-F. The voiceprivacy 2022 challenge evaluation plan, 2022.

Yoo, I.-C., Lee, K., Leem, S., Oh, H., Ko, B., and Yook, D. Speaker anonymization for personal information protection using voice conversion techniques. *IEEE Access*, 8:198637–198645, 2020. doi: 10.1109/ACCESS.2020.3035416.

## A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

**Arkadiy Aliev (20% of work)**

- Reviewing literature on LPC-based methods for voice anonymization (2 papers)
- Implement LPC-based aprroach
- Verifying the algorithm's ability to anonymize voice (Wav2Vec embendings, cosine distance)

**Vsevolod Ivanov (20% of work)**

- Reviewing literature (2 paper)
- Implementing Speaker anonimization algorithm using the McAdams coefficient
- Quality compare using WER metric

**Yuriy Ivanov (20% of work)**

- Reviewing literature on NN-based methods for voice anonymization (2 papers)
- Creating report and presentation
- Help with quality comparison

**Mikhail Mishustin (20% of work)**

- Study what the architecture typical for ASR and TTS looks like
- Creating report and presentation

**Andrey Safronov (20% of work)**

- Implementation of NN-based voice anonymization method based
- Quality comparison
- Github repository preparation

## B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **General comment:** If the answer is **yes**, students must explicitly clarify to which extent (e.g. which percentage of your code did you write on your own?) and which code was used.

   **Students' comment:** None

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

4. A complete description of the data collection process, including sample size, is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   ☐ Yes.
   ☐ No.
   ☑ Not applicable.

   **Students' comment:** None

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

9. The exact number of evaluation runs is included.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   **Students' comment:** None

10. A description of how experiments have been conducted is included.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    **Students' comment:** None

11. A clear definition of the specific measure or statistics used to report results is included in the report.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    **Students' comment:** None

12. Clearly defined error bars are included in the report.

    ☐ Yes.
    ☐ No.
    ☑ Not applicable.

    **Students' comment:** None

13. A description of the computing infrastructure used is included in the report.

    - ☐ Yes.
    - ☐ No.
    - ☑ Not applicable.

    **Students' comment:** None