Cost effective, Multilingual, Privacy-driven voice-enabled Services

www.compriseh2020.eu

Call: H2020-ICT-2018-2020
Topic: ICT-29-2018
Type of action: RIA
Grant agreement N°: 825081

WP N°2: Privacy-driven voice interaction
Deliverable N°2.3: Final transformation library and privacy guarantees
Lead partner: INRIA
Version N°: 1.0
Date: 28/02/2021

Document information	
Deliverable N° and title	**D2.3– Final transformation library and privacy guarantees**
Version N°	**1.0**
Lead beneficiary	**INRIA**
Author(s)	**David Adelani (USAAR), Ali Davody (USAAR), Thomas Kleinbauer (USAAR), Mossad Helali (USAAR), Mohamed Maouche (INRIA), Brij Srivastava (INRIA), Marc Tommasi (INRIA), Nathalie Vauquier (INRIA)**
Reviewers	**Raivis Skadiņš (TILDE) and Álvaro Moretón (ROOTER)**
Submission date	**28/02/2021**
Due date	**28/02/2021**
Type[1]	**OTHER**
Dissemination level[2]	**PU**

Document history			
Date	**Version**	**Author(s)**	**Comments**
08/02/2021	0.1	David Adelani, Ali Davody, Thomas Kleinbauer, Mossad Helali, Mohamed Maouche, Brij Srivastava, Marc Tommasi, Nathalie Vauquier	Draft deliverable
23/02/2021	0.2	David Adelani, Ali Davody, Thomas Kleinbauer, Mossad Helali, Mohamed Maouche, Brij Srivastava, Marc Tommasi, Nathalie Vauquier	Revision based on the reviewers' comments
28/02/2021	1.0	Akira Campbell & Emmanuel Vincent	Final version reviewed by the Project Manager and the Coordinator

[1] **R**: Report, **DEC:** Websites, patent filling, videos; **DEM:** Demonstrator, pilot, prototype; **ORDP:** Open Research Data Pilot; **ETHICS:** Ethics requirement. **OTHER:** Software Tools

[2] **PU:** Public; **CO:** Confidential, only for members of the consortium (including the Commission Services)

Document summary

This deliverable closes the work on speech and text anonymisation in COMPRISE. It is devoted to the design, implementation, and evaluation of transformations aiming to delete the citizen's[3] identity and words carrying private information, and to the training of user-independent Speech-to-Text and Natural Language Processing models from the resulting anonymised data. The proposed privacy-driven speech and text transformations are integrated in the COMPRISE Client Library, which is part of the COMPRISE Software Development Kit (SDK), and are meant to run on the user's device or the user's instance of the COMPRISE Personal Server. We recall the choices that govern software development and present the final architecture of our solution and the developed software. We then describe the evaluation methodology for the proposed anonymisation methods and discuss the results. We also provide recent results on improving privacy during model learning.

[3] In this report we will use the terms "user" and "citizen" to refer to the person speaking to the dialogue system. From the GDPR point of view, the citizen is the "data subject".

Contents

1 Introduction

Modern applications now allow the user's voice to be the main means of interaction with computers or smart objects. Technologies in voice interaction rely on machine learning models trained on user's data. This raises serious privacy concerns because speech is considered as biometric data, and carries a lot of private and sensitive information. One of the objectives of COMPRISE is to design a framework that enables the training and the use of a voice interaction tool in a private-by-design manner. In COMPRISE, the voice interaction chain consists of two branches: the operating branch and the training branch (see Figure 1). Privacy in the operating branch is ensured by running all computations on the user's device or on a trusted instance of the COMPRISE Personal Server, and sending out only the information needed to deliver the required service provided by the app. Work Package 2 focuses on ensuring privacy in the training branch. To do so, we have introduced two complementary tools: the COMPRISE Voice Transformer and the COMPRISE Text Transformer. These tools aim to anonymise the user's speech and text data before it is sent to the COMPRISE Cloud Platform to (re)train user-independent Speech-to-Text (STT), Spoken Language Understanding (SLU) and Dialogue Management models. They have been integrated in the COMPRISE Client Library and are also meant to run on the user's device or a trusted instance of the COMPRISE Personal Server.

This report summarises the work done along this line since the beginning of COMPRISE, and introduces recent advances achieved in terms of speech transformation (pitch transformation in x-vector based voice conversion) and text transformation (support for multilingualism, and defences against the inference of certain speaker traits from text). It also presents the privacy guarantees stemming from the latest evaluation results, and a method to improve privacy during model learning.

The structure of the report is as follows. The choices that govern software development and the final design of the architecture are reported in Section 2. We present in Section 3 the software we deliver. In Section 4, we describe the evaluation methodology and the corresponding results, as well as recent results on improving privacy during user-independent model learning. A summary of the results appears in Section 5.

2 Final design

The global architecture representing the main tasks and the flow of information between them is depicted in Figure 1. In the learning branch, the aim is to produce privacy-preserving *transformed* (a.k.a. *neutral* or *anonymised*) speech and text data that will be used for the (re)training of STT, SLU, and Dialogue Management components.

The citizen's speech input is first processed by an STT module to obtain the corresponding text. The text is processed by the *Text Transformer* module that replaces sensitive words. The speech signal is also processed by the *Voice Transformer* module into a transformed voice, and then by the *Secure Voice Builder* module to obtain a secured speech signal where the sensitive words identified by the Text Transformer have been removed. These software modules are detailed below.

`VoiceTransformer:` this module involves two steps: `fit` and `transform`. The first one, `fit`, computes internal parameters to fit the transformation to the user's voice. Some transformations must be instantiated with specific features of the user's voice and this system is no exception. It needs a few speech utterances as input that are used only once. Once instantiated, `transform` performs the transformation of the user's speech to the target (neutral) voice.

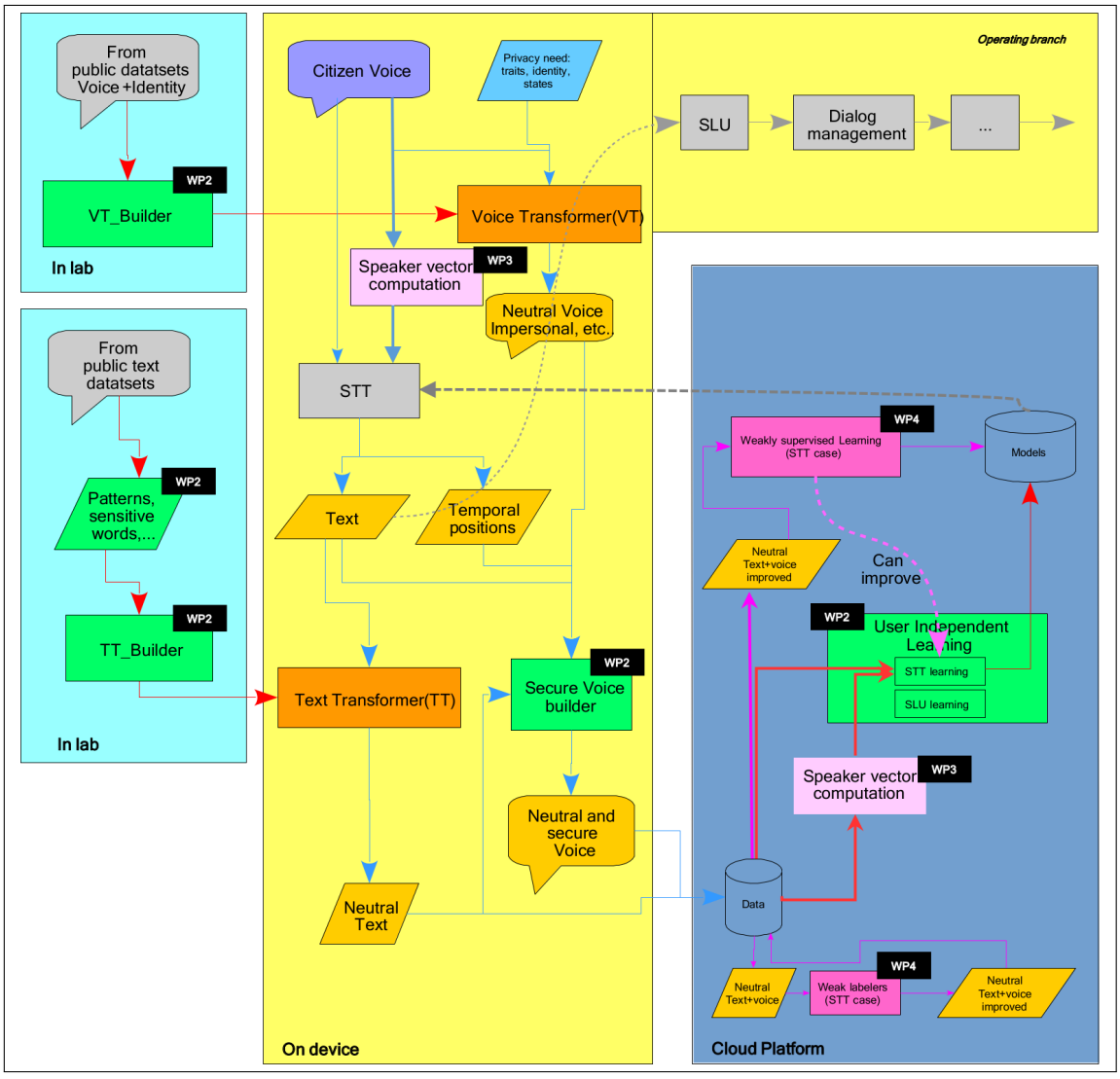


Figure 1: Overview of the global architecture of the learning branch, including the interactions with other WPs.

`TextTransformer`: this module involves two steps: first, it identifies those words in the input text that are considered as sensitive by means of Named Entity Recognition (NER), and then it replaces them by placeholders or other words to obtain transformed (neutral) text.

`SecureVoice_Builder`: this module also involves two steps: `align-s2t` and `mask_words_in_speech`. First, `align-s2t` takes the transformed (neutral) speech and the original (untransformed) text as inputs and it outputs the temporal positions of the words in the speech signal. Then, `mask_words_in_speech` takes the transformed speech and text and the estimated temporal positions as inputs and it outputs a secured speech signal where sensitive words have been removed. The `mask_words_in_speech` script gives fine control over the words that are to be masked in the speech signal. Hence, `SecureVoice_Builder` also includes an optional helper script `mask_words_in_text` which produces transformed (neutral) text with the same type of words and named entities that are masked in the speech signal. This feature is meant for detailed evaluation purposes.

Two specific tools are responsible for creating the Voice and Text Transformers. They are represented in the light blue frames on the left of the diagram.

`VT_Builder`: this tool takes as input a dataset containing speech signals and speaker identities and produces a Voice Transformer.

`TT_Builder`: this tool is responsible for selecting the appropriate transformation strategy to use inside the Text Transformer and for continuously improving the underlying models. Currently, `TextTransformer` uses fixed, external models only, pushing the need for a `TT_Builder` to a later stage.

It is important to note that, to maximise privacy and utility at the same time, the anonymised data which are sent to the COMPRISE Cloud Platform depend on the downstream task. When a given user input is to be used for training the acoustic model of an STT system, the data sent to the COMPRISE Cloud Platform consists of secured speech together with the corresponding text (from which the detected sensitive words have been *removed*). For greater privacy, the speech input and the corresponding text may even be cut into shorter slices without loss of utility (see Section 4.2.6). When the user input is to be used for training the language model of an STT system or a model for SLU, Dialogue Management, or any other text-only task, only transformed text (in which the detected sensitive words have been *replaced*) is sent to the COMPRISE Cloud Platform instead. That text cannot be cut into shorter slices. Quite the opposite, in the case when the SLU or Dialogue Management engine relies on past user inputs within a given dialogue, all those transformed text inputs must even be uploaded together. Therefore, for each dialogue with a given user, the COMPRISE Client Library will choose at random which training dataset (STT acoustic model, STT language model, SLU, Dialogue Management, etc.) it will contribute to.

This is because storing secured speech in a situation when only transformed text is required would result in a privacy leakage, since the secured speech signal makes it possible to identify which sensitive words in the transformed text have been missed by the sensitive word detector and/or have not been replaced. Conversely, storing transformed text (and the corresponding long speech utterances) in a situation when only short speech slices are required would also result in a privacy leakage.

3 Implementation and distribution

In Deliverable D2.1 "Baseline speech and text transformation and model learning library" (Submitted to the European Commission on August 30, 2019 – Public), we released initial versions of the COMPRISE Voice Transformer, the COMPRISE Text Transformer, and the Secure Voice Builder. The initial version of the Voice Transformer was based on two voice conversion techniques: VoiceMask [Qia+17] and Vocal Tract Length Normalisation (VTLN) [SN03]. The Text Transformer identifies named entities and replaces them by other words or by placeholders. Scripts were also delivered to train and evaluate these transformations.

In Deliverable D2.2 "Improved transformation library and initial privacy guarantees" (Submitted to the European Commission on April 30, 2020 – Public), we released an improved version of the COMPRISE Voice Transformer that relies on the x-vector based voice transformation method we contributed to developing as a baseline for the Voice Privacy Challenge. We also provided Docker containers with the execution environments for the Voice Transformer (and builder) and the Text Transformer.

In the current deliverable, we release the final versions of the COMPRISE Voice Transformer, the COMPRISE Text Transformer, and the Secure Voice Builder. The x-vector based voice conversion method in the Voice Transformer has been further improved by means of pitch transformation (see Section 4.2.4), the Text Transformer has been made multilingual, and the Voice and Text Transformers are now fully packaged as RESTful services in Docker images. The source code is available in open source under the GPLv3 licence, using Python as the main language, on the COMPRISE Gitlab repository.[4] The Text Transformer uses the Flair library.[5] For the Voice Transformer, we rely on the Kaldi toolkit,[6] the flac library,[7] and the CURRENNT toolkit[8] of the National Institute of Informatics (NII), Japan. We illustrate below the ease of use of this software by providing some use-case examples of the dockerised RESTful services.

3.1 Voice Transformer

Due to its relying on Kaldi and the CURRENNT toolkit, the Voice Transformer requires a CUDA-capable graphic card. When using Docker, the NVIDIA Container Toolkit[9] is also required. If both these requirements are met, the simplest way to run the Voice Transformer is to run the RESTful service with Docker:

```
docker run -d --gpus all -p 5000:5000 \
       registry.gitlab.inria.fr/comprise/voice_transformation
```

The Voice Transformer service is then running on port 5000 and exposes the following endpoint: `http://localhost:5000/vpc`

- Method: POST
- Input: audio file, parameters (JSON)

[4] https://gitlab.inria.fr/comprise/
[5] https://github.com/flairNLP/flair
[6] http://kaldi-asr.org/
[7] https://xiph.org/flac/
[8] https://github.com/nii-yamagishilab
[9] https://github.com/NVIDIA/nvidia-docker

- Output: audio file

Here is an example of the way to use it in Python:

```
import requests
import json

# replace localhost with the proper hostname
API_URL = 'http://localhost:5000'

# Read the content from an audio file
input_file = 'samples/vctk_p225_003.wav'
with open(input_file, mode='rb') as fp:
    content = fp.read()

# Transformation parameters
params = {'wgender': 'f',
          'cross_gender': 'same',
          'distance': 'plda',
          'proximity': 'random',
          'sample-frequency': 48000}

# Call the service
response = requests.post('{}/vpc'.format(API_URL),
                         data=content,
                         params=json.dumps(params))

# Save the result of the transformation in a new file
result_file = 'transformed.wav'
with open(result_file, mode='wb') as fp:
    fp.write(response.content)
```

The parameters of the transformation (described in Section 4.2.3) are:

- `wgender` (required): gender of the original speaker whose voice is to be transformed,
- `cross-gender`: gender of the target speaker (`same` as the original speaker, `other`, or `random`),
- `anon-pool`: path to the pool of speakers from which the target is to be selected,
- `distance`: distance metric between x-vectors (`plda` or `cosine`),
- `proximity`: strategy for choosing the target speaker in the pool (`dense`, `farthest` or `random`).

The Transformer expects audio files sampled at 16 kHz by default, but other sampling frequencies can be accepted using the `sample-frequency` parameter.

At a lower level, the Voice Transformer is implemented as a Kaldi recipe and can be used as is, for the use cases not fulfilled by the RESTful API (for example, building the anonymisation pool, or batch processing): the entry points are then a set of scripts, one for each functionality. We also

provide another Docker image with the required execution environment (including Kaldi) to run these scripts. The instructions to use them are given in the documentation in the Gitlab repository.

Note that voice technologies require high computational power. Our solution relies on a Kaldi recipe that runs on machines equipped with a GPU. Loading the trained models and performing initialisation is time-consuming. Therefore, we recommend to apply the Voice Transformer on a batch of utterances. For instance on a Tesla P100-PCIE-12GB GPU card the processing time for one LibriSpeech utterance (with an average duration of 5 s) is 17 s, but it is only 4 s longer for a second utterance and 2 min 26 s for a batch of 54 utterances.

3.2 Text Transformer

The Text Transformer is implemented in Python and requires `numpy` and the `flair` library. When using Docker, the NVIDIA Container Toolkit is also required. If these requirements are met, the simplest way to run the Text Transformer is to run the RESTful service with Docker:

```
docker run -d -p 5000:5000 registry.gitlab.inria.fr/comprise/text_transformer
```

The Text Transformer service is then running on port 5000 and exposes the following endpoint: `http://localhost:5000/vpc`

- Method: POST
- Input: original text (string), parameters (JSON)
- Output: transformed text (string)

Here is an example of the way to use it in Python:

```
import requests
import json

text_to_transform= 'I live in Berlin'
# replace localhost with the proper hostname
ENDPOINT_URL = 'http://localhost:5000/transform'

# Transformation parameters
params = {'r': 'WORD'}

# Call the service
response = requests.post(ENDPOINT_URL,
                         data=text_to_transform,
                         params=json.dumps(params))

# Get the result
print(response.text)  #  'I live in Sweden'
```

The parameters of the transformation are:

- `m`: name of the Named Entity Recognition model from the `flair` library or path to a model trained with the builder to be used for detecting sensitive words,
- `r`: replacement strategy (`REDACT`, `WORD` or `FULL`).

3.3 Secure Voice Builder

The Secure Voice Builder removes sensitive words from the speech signal. It comprises two steps, Alignment and Masking, and works for any language.

- The Alignment tool (`align-s2t`) aligns a batch of speech files and the corresponding text transcriptions. It outputs a set of alignment timestamps at the word level and at the phone level. In order to perform the alignment, it relies on an STT model in Kaldi format. More specifically, it is compatible with any GMM-HMM or nnet3 model obtained via the unified STT training recipe in the COMPRISE Cloud Platform[10].
- The Word Masking Tool masks sensitive words in both the speech files and the corresponding text transcriptions. It takes a batch of speech files, the corresponding text transcriptions and alignment timestamps and the list of words to be masked as inputs, and it produces privacy-preserving transformed (neutral) speech and text.

3.4 European Language Grid

The European Language Grid (ELG) is a catalogue and a cloud platform that offers access to a multitude of assets related to language technology (LT), including services and data resources. Our technical choices of using RESTful services and Docker make our tools compatible with the requirements of the ELG. Therefore, we have recently submitted the Text Transformer and the Voice Transformer to the ELG. The Text Transformer will be available directly on the ELG grid for testing and demonstration. The Voice Transformer will be available for download only, since it requires a virtual machine equipped with a GPU — a service that is currently not available on the ELG grid.

4 Evaluation

4.1 Privacy and risks in voice-enabled systems

In this work package, we have proposed different analyses of voice-enabled systems, the definition and the role of privacy, its impact from a legal point of view, and the associated risks. A COMPRISE partner, ROOT, has published several reports on the topic. One of the first results was the categorisation of personal data, described in Deliverable D2.1. Our article [MJ20] provides an overview of how personal data recorded by voice-enabled systems can be identified through the categorisation of personal information and the analysis of context. The study reveals that the context of use at the moment a user interacts with the system is of major importance to disambiguate and identify personal data. The paper depicts many situations where *contextualisation* is of major importance with respect to personal data. This work enlightens the benefits brought by the choices made in defining the architecture of COMPRISE.

In a recent report [MJC21], we have studied anonymisation and re-identification risks. We have recalled the concepts of anonymisation and pseudonymisation given in Recital 26 of the GDPR. Different interpretations of this text are made by different European Supervisory Authorities. To conform the most restrictive aspects in Europe we can follow the French CNIL that considers that anonymisation is achieved when *identification is practically impossible*. Less restrictive is, for

[10] `https://gitlab.inria.fr/comprise/comprise-stt-training-recipe`

example, the Irish Data Protection Authority (DPA), that considers enough to demonstrate that the re-identification is highly unlikely given the specific circumstances. Taking this difference into consideration, the report proposes a risk-based approach that determines whether data qualifies as personal or non-personal after applying anonymisation techniques to a dataset. This way, it is possible to evaluate the risk of re-identification to have a better view of the impact of our technologies. Therefore, we list the aspects to be considered as a re-identification risk. In the report we also list the motivations of an attacker in order to better evaluate the risk. Interestingly, we have identified specific and additional sources of information an attacker may have in the case of voice-enabled technologies. As a consequence, we give general recommendations for data governance in the different use cases associated with voice-enabled technologies in order to reduce re-identification risk. In particular a set of actions to ensure unlinkability is proposed. In the report, the choices made in COMPRISE are also analysed. As a conclusion, we were able to elaborate a complete basis including a reliable risk evaluation of our solution, which will be useful to obtain accreditation from the different national data protection authorities.

4.2 Speech transformation

4.2.1 Attacker model and privacy metrics

The evaluation of the strengths and weaknesses of the proposed anonymisation techniques requires the definition of an attacker model. We introduced an original, rigorous evaluation methodology that takes the attacker knowledge into account and relies on strong and adequate metrics for privacy.

The attacker model considered in COMPRISE in depicted in Figure 2. Given a public dataset of anonymised speech utterances contributed by several speakers using COMPRISE apps, an attacker records/finds a sample of speech of a speaker and attempts to guess which utterances in the anonymised dataset have been spoken by that speaker, potentially leveraging some knowledge about the anonymisation algorithm. A good anonymisation scheme must prevent such linkage attacks from being successful, while preserving the perceived speech naturalness and intelligibility and/or the performance of downstream tasks such as training an STT system. The figure shows the three actors, namely the speaker (the user of the COMPRISE app), the data user (for instance, the operator of the COMPRISE Cloud Platform who is trying to learn new a STT model), and the attacker, together with the corresponding actions.

One originality of this research was to consider different levels of knowledge for the attacker. As opposed to past studies, we consider different linkage attacks depending on the attacker's knowledge of the anonymisation method. At one end of the spectrum, an *Ignorant attacker* is unaware of the speech transformation being applied. This is a form of privacy by obfuscation which is not acceptable for serious privacy requirements. At the other end an *Informed attacker* can leverage complete knowledge of the transformation algorithm and the chosen parameter values. This represents the situation when the citizen's device has been fully compromised and we fall into a security, but not privacy, problem. Informed and ignorant attackers give us lower and upper privacy bounds for insightful comparisons. In the middle of the spectrum lie various attackers who know the voice transformation algorithm but not its parameter values. This matches the COMPRISE setting because the source code of the Voice Transformer is open and thus publicly available. We refine this intermediate level of knowledge by considering *Semi-Informed* and *Lazy-Informed attackers* that require different computational power and external resources for their attack.

A second originality in our evaluation was to consider, compare and analyse the results given by three metrics:

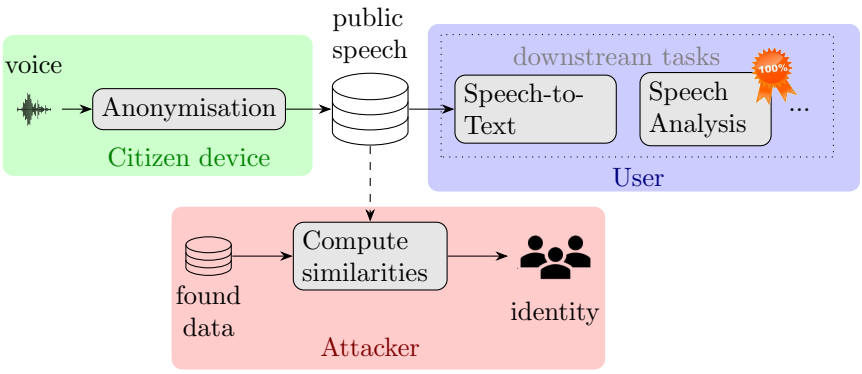


Figure 2: COMPRISE attacker model. Speech is anonymised on the citizen's device or Personal Server and sent to the COMPRISE Cloud Platform to build a public dataset, which is used for downstream tasks such as STT model training. An attacker tries to retrieve the identities of speakers in the public dataset by computing similarities with some speech samples collected by other means.

Equal Error Rate (EER): the EER is the classical metric used in speaker verification. It assumes a threshold-based decision on the score of a linkage function. The EER is the error rate corresponding to the threshold t^* for which the rates of false alarm and miss errors are equal.

Log-Likelihood-Ratio (LLR) cost function: $C_{\text{llr}}^{\text{min}}$ and $C_{\text{llr}}^{\text{min}}$ are application-independent speaker verification metrics. They pool across all possible costs for false alarm vs. miss errors, and all possible priors for mated vs. non-mated trials.

Linkability: this metric was proposed for biometric template protection systems. Denoting by H (resp., $\bar{H}$) the binary variable expressing whether two random utterances a and b are mated (resp., non-mated), a local linkability metric for a score s of the linkage function is defined as $p(H \mid s) - p(\bar{H} \mid s)$. The linkability metric, denoted by $D_{\leftrightarrow}^{\text{sys}}$, is a global version defined as the integral over scores of the local linkability, weighted by $p(s \mid H)$.

The EER and $C_{\text{llr}}^{\text{min}}$ assume that the attacker makes threshold-based decisions on the linkage score, while $D_{\leftrightarrow}^{\text{sys}}$ implicitly models a more powerful, non-threshold-based *oracle* attacker. The comparison on real speech data processed via 4 anonymisation techniques with different target selection strategies and with 9 attackers suggests that these metrics behave similarly. Yet, experiments on simulated data highlight fundamental differences. Specifically, the EER may yield a fixed value for situations involving different levels of privacy correctly captured by $C_{\text{llr}}^{\text{min}}$, and $C_{\text{llr}}^{\text{min}}$ becomes less informative than $D_{\leftrightarrow}^{\text{sys}}$ when the mated scores are lower or interleaved with non-mated scores.

4.2.2 Approaches implemented in Comprise

In COMPRISE, several privacy-driven voice transformation methods have been implemented, improved and compared.

VoiceMask: This voice transformation method described in [Qia+18; Qia+17] performs frequency

warping based on the composition of a quadratic function and a bilinear function using two different parameters.

VTLN-based conversion: This voice conversion method described in [SN03] represents each speaker by a set of spectra for k phonetic classes, and maps the original speech to the target speaker's voice by finding the transformation parameters that minimise the distance between target class spectra and transformed original class spectra.

The VPC method: COMPRISE members have been actively involved in the organisation of the Voice Privacy Challenge (VPC)[11] and have joined efforts with Avignon Université, EURECOM, and NII to design and implement an attacker scenario, evaluation metrics and two baseline methods for speech privacy. One of these baseline methods is inspired from the speaker anonymisation method proposed in [Fan+19], which performs voice conversion based on x-vectors [Sny+18], a fixed-length representation of speech signals that form the basis of state-of-the-art speaker verification systems. We have brought several improvements to this method, which are described in Sections 4.2.3 and 4.2.4 below.

All these methods are available in the COMPRISE Voice Transformer Gitlab repository. The main advantage of VoiceMask and VTLN transformations is their computational speed but they do not provide sufficient privacy guarantees for entire utterances. Therefore we propose to use the VPC x-vector based anonymisation method, as detailed below.

4.2.3 X-vector based anonymisation

In their original implementation of x-vector based anonymisation [Fan+19], Fang et al. proposed to select x-vectors within a pool of speakers at a fixed distance from the x-vector of the original speaker, and to combine them to obtain a target *pseudo-speaker* representation. This representation, along with the phonetic representation of the original utterance, is provided as input to a Neural Source-Filter (NSF) based speech synthesiser [WTY19] to produce anonymised speech.

Our improved x-vector based anonymisation scheme is shown in Fig. 3. It consists of four steps. *Step 1 (Feature extraction)* extracts fundamental frequency (F0) and bottleneck features and the original speaker's x-vector from the input signal. *Step 2 (X-vector anonymisation)* anonymises this x-vector by averaging N^* candidate target x-vectors from a pool of speakers called the *anonymisation pool*.[12] *Step 3 (Pitch transformation)* is an optional stage which receives the pseudo-speaker target pitch statistics from the anonymisation module and transforms the original pitch. *Step 4 (Speech synthesis)* synthesises a speech waveform from the anonymised x-vector and the original F0 and bottleneck features using an acoustic model (AM) and the NSF model. We refer to [Tom+20] for details on components 1, 2 and 4 of this architecture, which are part of the VPC baseline method. Component 3 is new and detailed below.

4.2.4 Design choices for x-vector based anonymisation

In the algorithm above, the method used to choose the candidate target x-vectors from the anonymisation pool has a strong impact on privacy and utility. We have investigated four design choices, which are summarised in Fig. 4.

[11] `https://www.voiceprivacychallenge.org/`.

[12] There is no guarantee that averaging produces a valid x-vector but all of our experiments show that the synthesised anonymised speech is of good quality.

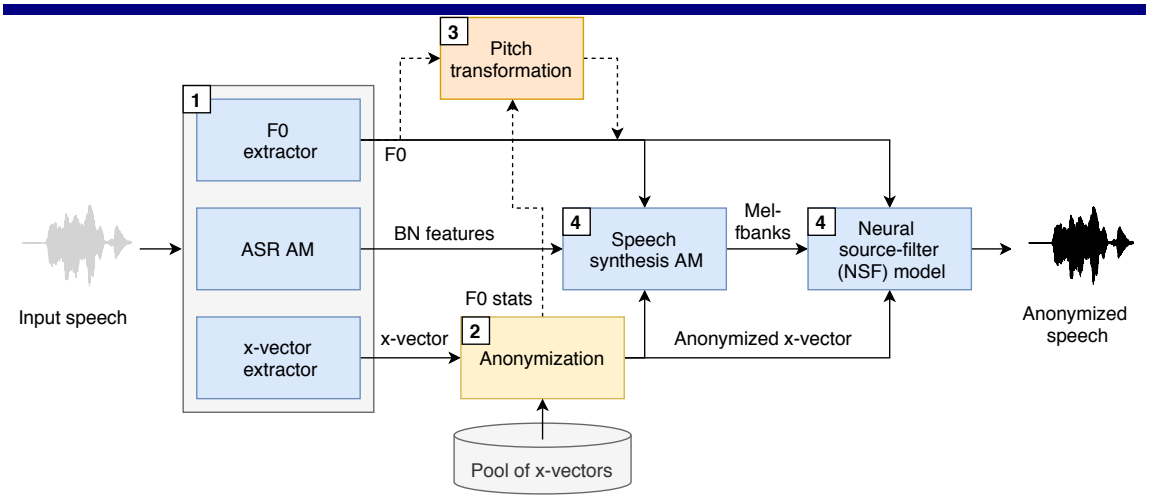


Figure 3: General architecture of our x-vector based anonymisation method.

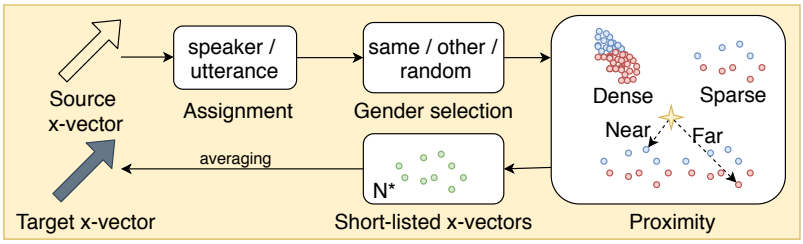


Figure 4: Zoomed-in view of the anonymisation step showing the design choices for the construction of pseudo-speakers.

Distance metric: cosine vs. PLDA We have compared two metrics to identify candidates for target x-vectors. The first one is the cosine distance, which was used by [Fan+19]. For a pair of x-vectors ω_i and ω_j, it is defined as

$$d_{\cos}(\omega_i, \omega_j) = 1 - \frac{\omega_i \cdot \omega_j}{||\omega_i||_2 ||\omega_j||_2} \tag{1}$$

The second one relies on Probabilistic Linear Discriminant Analysis (PLDA) [Iof06]. PLDA models x-vectors ω as $\omega = m + Vy + Dz$, where m is the centre of the acoustic space, the columns of V represent speaker variability (eigenvoices) with y depending only on the speaker, and the columns of D capture channel variability (eigenchannels) with z varying from one recording to another. The parameters m, V and D are trained using x-vectors from the training set for the x-vector model, which is used to generate the anonymisation pool. The log-likelihood ratio of same-speaker ($\mathcal{H}_\text{s}$) and different-speaker ($\mathcal{H}_\text{d}$) hypotheses

$$\text{PLDA} = \log \frac{p(\omega_i, \omega_j | \mathcal{H}_\text{s})}{p(\omega_i, \omega_j | \mathcal{H}_\text{d})} \tag{2}$$

can be computed in closed form [RBS14]. We propose to use minus-PLDA as the "distance" between a pair of x-vectors.

Proximity of pseudo-speaker: random, near, far, dense, or sparse Various criteria can be used to select the region of the x-vector space where pseudo-speakers are located.

The simplest strategy consists in simply selecting N^* x-vectors uniformly at *random* from the same gender as the original speaker in the anonymisation pool. Note that this strategy does not allow us to choose particular regions of interest in the x-vector space.

Alternatively, distances can be used to define regions in x-vector space which closely resemble or least resemble the original speaker S. In essence, we rank all the x-vectors in the anonymisation pool in increasing order of their distance from S and select either the top N (*near*) or the bottom N (*far*). To introduce some randomness, $N^* < N$ x-vectors are selected out of these N, uniformly at random. The variability of results is controlled by a fixed random seed.

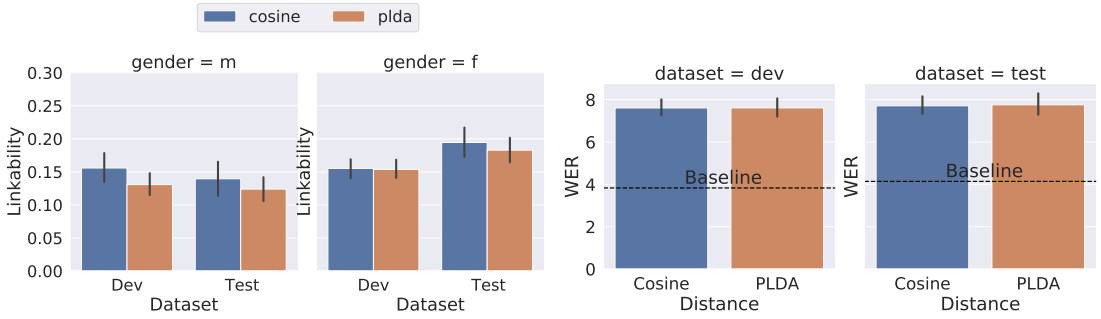
Yet another alternative is to identify clusters of x-vectors in the anonymisation pool and select clusters with a low (*sparse*) or a high (*dense*) density (i.e., number of members). These clusters are constructed using the Affinity Propagation [Due09] algorithm. We randomly select half of the members of the desired cluster and average them to generate a *pseudo-speaker*. The fraction of clusters to be selected and the random seed control the reproducibility of the *pseudo-speaker*.

Gender selection: same, opposite, or random We propose gender selection as a design choice to study its impact on anonymisation and intelligibility. We have the gender information for the original speaker as well as the speakers in the *anonymisation pool*. Hence this design choice can be combined with all *proximity* choices. We study three different types of gender selection: *same* where the candidate target x-vectors are constrained to be of the same gender as the original, *opposite* where they are constrained to be of the opposite gender, and *random* where the target gender is selected at random before picking candidate x-vectors of that gender.

Assignment: speaker-level or utterance-level In theory, all the utterances spoken by a given speaker must exhibit the same x-vector. In practice, however, deviations can be observed due to channel, duration, and phonetic variability [Raj+19]. It is therefore natural to examine whether assigning the same target pseudo-speaker to all utterances of a given original speaker or a different target pseudo-speaker for each utterance impacts privacy and utility.

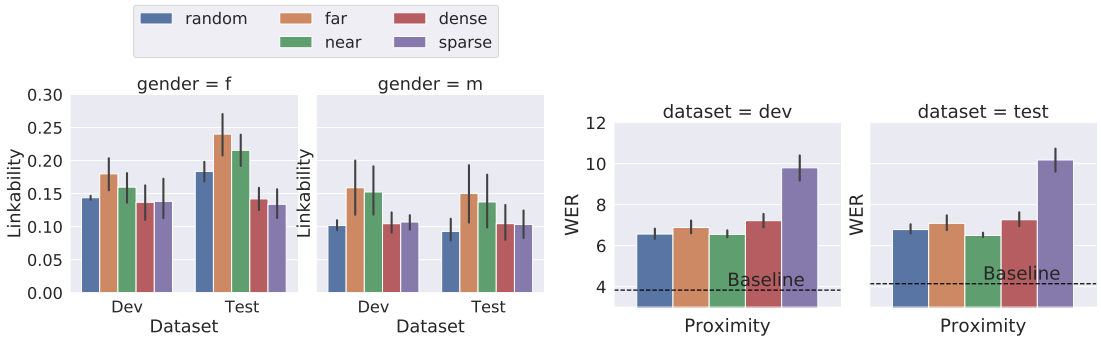
Results We have introduced the concept of attacker in Section 4.2.1, which is an entity determined to breach the privacy of speakers by revealing their true identities. We simulate such attackers using a state-of-the-art x-vector-PLDA speaker verification setup and evaluate the privacy protection achieved by a given set of design choices while selecting the pseudo-speaker. The weakest attacker called the *Ignorant* attacker knows nothing about the anonymisation method. A more advanced attacker called the *Lazy-Informed* attacker knows the anonymisation method and the selected design choices, and applies them to the sample data they have recorded/found for each speaker before performing speaker verification trials with an x-vector extractor and a PLDA metric trained on untransformed data. The *Semi-Informed* attacker operates similarly to the Lazy-Informed attacker, except that they also apply the anonymisation method to the dataset used to train the x-vector extractor and the PLDA metric used for speaker verification. Note that, due to the randomness of the target pseudo-speaker selection method, the target pseudo-speakers assigned to the anonymised utterances and the sample utterances for a given speaker are typically different.

In Figures 5–8, we report the privacy protection achieved against the Lazy-Informed attacker as well as the loss of utility incurred when a given set of design choices are selected. We use the same training, development, and test datasets as in [Tom+20], where the development and test sets are built from LIBRISPEECH *dev-clean* and *test-clean*, respectively. Privacy protection is measured in terms of Linkability, as defined in Section 4.2.1. Lower Linkability implies higher privacy protection since it is hard for the attacker to link the original identities to the anonymised speech. Utility is measured by the word error rate (WER) achieved when decoding anonymised speech with an STT model trained on untransformed speech. Based on the observed level of privacy and utility, we can conclude that **PLDA** distance, **random** or **dense** proximity, **random** gender selection and **speaker-level** assignment form an effective combination of design choices in the x-vector space.



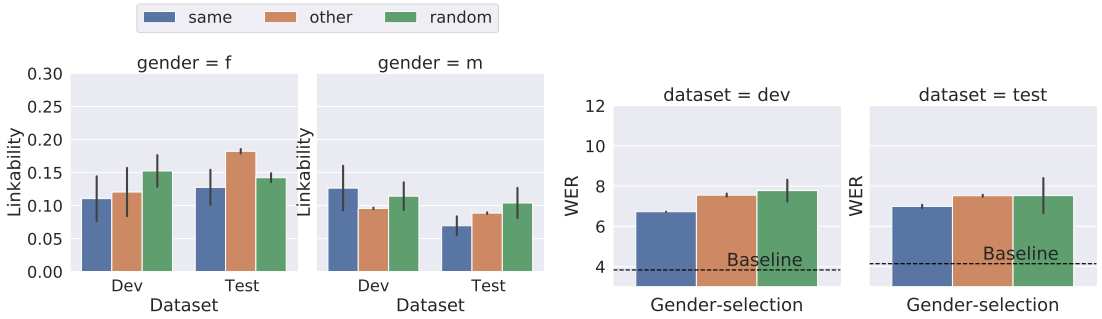
(a) Privacy (b) Utility

Figure 5: Privacy protection against a Lazy-Informed attacker and utility achieved depending on the 'distance metric' design choice. The results displayed are the mean and standard deviation over the other design choices (proximity, gender selection, assignment).



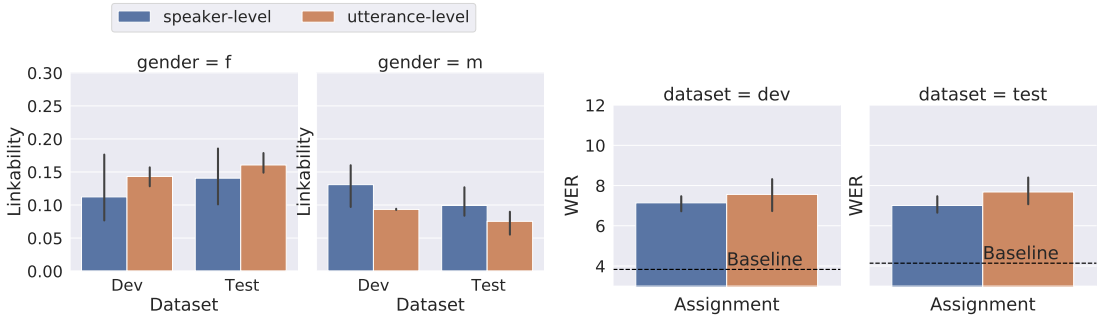
(a) Privacy (b) Utility

Figure 6: Privacy protection against a Lazy-Informed attacker and utility achieved depending on the 'proximity' design choice. The results displayed are the mean and standard deviation over the other design choices (gender selection, assignment) with PLDA as the distance metric.



(a) Privacy (b) Utility

Figure 7: Privacy protection against a Lazy-Informed attacker and utility achieved depending on the 'gender selection' design choice. The results displayed are the mean and standard deviation over the other design choices (proximity, assignment) with PLDA as the distance metric.

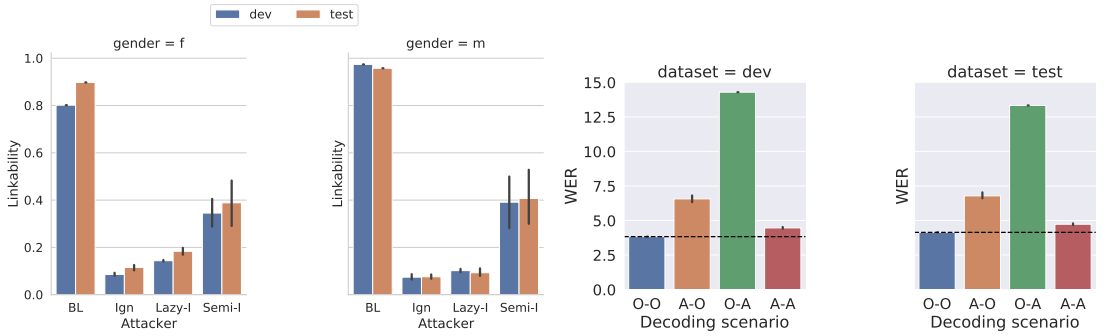


(a) Privacy (b) Utility

Figure 8: Privacy protection against a Lazy-Informed attacker and utility achieved depending on the 'assignment' design choice. The results displayed are the mean and standard deviation over the other design choices (proximity, gender selection) with PLDA as the distance metric.

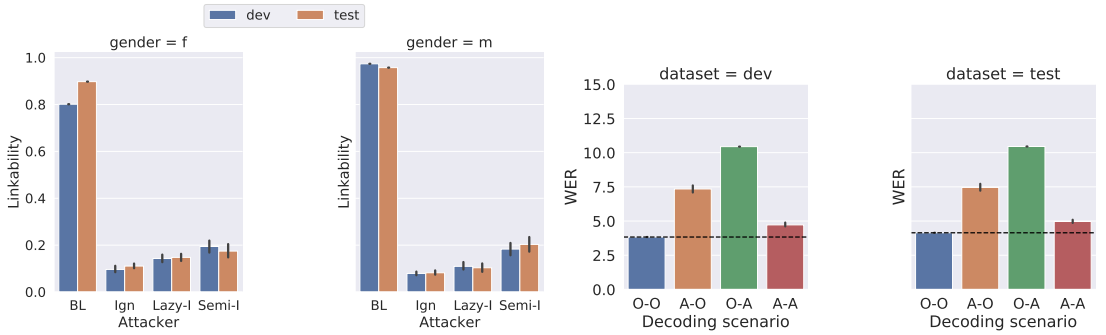
In Figures 9(a) and 10(a), we further evaluate the best design choices against all attackers, and compare them with the privacy protection offered by original (untransformed) speech. We observe that the value of Linkability gradually increases as we move from the weakest (*Ignorant*) to the strongest (*Semi-informed*) attacker, both in the case of *Random* and *Dense* anonymisation. The mean Linkability remains below 0.2 for *Dense* and below 0.4 for *Random*, and its standard deviation is smaller for *Dense* than for *Random*. This indicates the superiority of *Dense* proximity over *Random*. This design choice is capable of cutting Linkability down to a quarter of its baseline value against an attacker with complete knowledge of the anonymisation algorithm and its parameters, who only lacks the knowledge of exact pseudo-speaker targets.

Figures 9(b) and 10(b) report the WERs achieved when decoding original or anonymised speech with an STT acoustic model trained on either original or anonymised speech. The baseline O-O



(a) Privacy (b) Utility

Figure 9: Privacy protection against various attackers (BL = original speech, Ign = Ignorant, Lazy-I = Lazy-Informed, Semi-I = Semi-Informed) and utility achieved using Random proximity. The results displayed are the mean and standard deviation over the other design choices (gender selection, assignment) with PLDA as the distance metric.



(a) Privacy (b) Utility

Figure 10: Privacy protection against various attackers (BL = original speech, Ign = Ignorant, Lazy-I = Lazy-Informed, Semi-I = Semi-Informed) and utility achieved using Dense proximity. The results displayed are the mean and standard deviation over the other design choices (gender selection, assignment) with PLDA as the distance metric.

corresponds to original speech decoded by the original STT model (trained on original speech), A-O to anonymised speech decoded by the original model, O-A to original speech decoded by a model trained on anonymised speech, and A-A to anonymised speech decoded by a model trained on anonymised speech. The high WERs in the A-O and O-A cases are due to a mismatch between the training and test data. The WER degradation compared to the baseline is larger in the O-A case than the A-O case, which suggests that the STT model trained on anonymised speech exhibits lower generalisation capability than the one trained on original speech. Nevertheless, we observe that the WER in the A-A case is very similar to the baseline. This indicates that anonymisation

does not incur a major loss of utility for STT acoustic model training, provided that STT models trained on anonymised speech are applied to anonymised (rather than original) speech.

Pitch transformation In the above experiments, only the x-vector of the original utterance has been anonymised, while the two other sets of features (F0 and bottleneck) have been left unchanged. Yet, prosodic features such as F0 can might reveal some information about the speaker identity [Far+08]. Hence, we evaluate two pitch transformation methods to better conceal the identity as well as to enhance the naturalness of the output speech.

The first method is Gaussian normalisation [LZY07; CJL21], where the original log-F0 is transformed into the target log-F0 via the affine transform

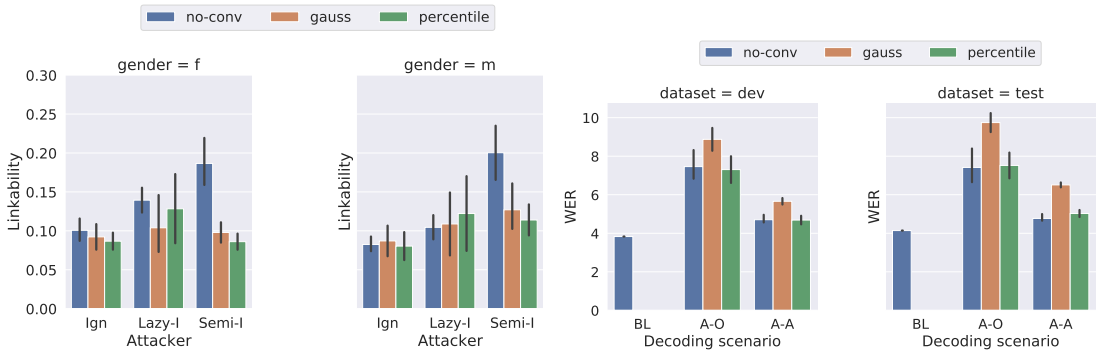
$$\log(\mathrm{F0}_{\mathrm{tgt}}) = \frac{\log(\mathrm{F0}_{\mathrm{org}}) - \mu_{\mathrm{org}}}{\sigma_{\mathrm{org}}}\sigma_{\mathrm{tgt}} + \mu_{\mathrm{tgt}} \tag{3}$$

where μ_{org}, σ_{org}, and μ_{tgt}, σ_{tgt} are the mean and standard deviation of the original and target speaker's log-F0, respectively.

The second transformation method is based on mapping each percentile of the original speaker's pitch to the corresponding percentile of the target speaker's pitch. To the best of our knowledge, this method is novel and has not been reported in previous literature. The target pitch is computed as

$$\mathrm{F0}_{\mathrm{tgt}} = F^{-1}_{\mathrm{tgt}}(F_{\mathrm{org}}(\mathrm{F0}_{\mathrm{org}})) \tag{4}$$

where $F_{\mathrm{org}}(.)$ and $F_{\mathrm{tgt}}(.)$ are the cumulative distribution functions of the original and target speaker F0, respectively. One advantage of percentile-based mapping is that the resulting pitch values all come from a set of valid pitch values, while in the case of Gaussian normalisation the transformed pitch might not be within the valid range of pitch values. Note that unvoiced frames are not taken into account when performing both transformation methods.



(a) Privacy (b) Utility

Figure 11: Privacy protection against various attackers (Ign = Ignorant, Lazy-I = Lazy-Informed, Semi-I = Semi-Informed) and utility achieved depending on the choice of pitch transformation. The results displayed are the mean and standard deviation over the other design choices (gender selection, assignment) with PLDA as the distance metric and Dense proximity.

Figure 11 shows that both pitch transformation methods significantly reduce the Linkability, especially in the *Semi-Informed* attack. This implies that the conversion of pitch removes some of the residual speaker information in the anonymised speech, thereby improving privacy protection. Furthermore, Gaussian pitch normalisation significantly increases the WER, while the percentile-based method maintains the original WER. We also found by informal listening that the naturalness of cross-gender voice conversion noticeably improves after percentile-based pitch transformation as compared to no transformation.

4.2.5 Evaluation of anonymisation with a large speaker population

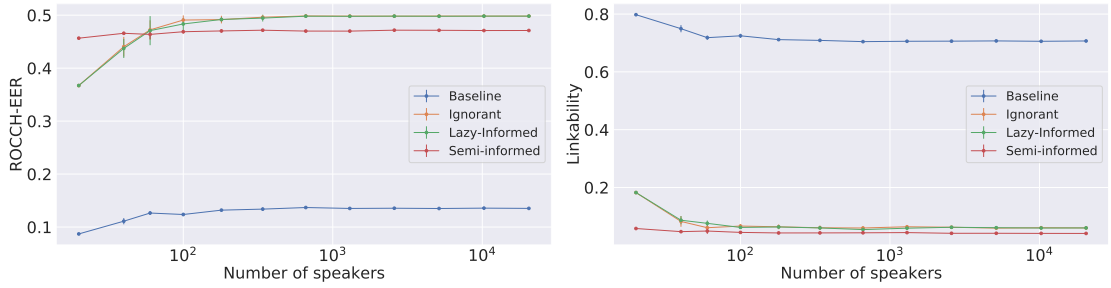
The above experiments have followed the Voice Privacy Challenge setup, which considers a set of 29 original speakers only. The number of speakers is considered as part of the attacker's prior knowledge since the attacker possesses their original and anonymised data. We now study whether the speaker's identity gets hidden in the crowd or can still be revealed in a larger population.

To do so, we employ Mozilla's Common Voice English dataset.[13] We select 24,616 speakers as the total population possessed by the attacker from the set of male speakers whose total duration is more than 10 s after removing silent frames using voice activity detection (VAD). We limit the maximum duration for each speaker in the population to 2 min, and we reject utterances whose Signal-to-Noise ratio (SNR) is lower than 75 dB, as measured using WADA-SNR [KS08].[14] Among these speakers, we select 20 speakers whose total duration is greater than 5 min for testing. The remaining utterances from these 20 speakers represent the publicly released data subjected to re-identification attacks. After computing PLDA scores between the speaker population and the test speakers, we get 4,696 same-speaker scores and 115,563,864 different-speaker scores. Initially we select the subset of scores which are computed only among the 20 test speakers. Then, we iteratively double the number of speakers in the population and include the scores corresponding to these speakers. The newly added speakers are randomly sampled 5 times from the entire speaker population to ascertain whether the sampling process induces a bias due to proximity with speakers.

Figure 12 assesses the performance of an open-set speaker verification attack and reports the

[13] `https://commonvoice.mozilla.org/`

[14] https://gist.github.com/johnmeade/d8d2c67b87cda95cd253f55c21387e75



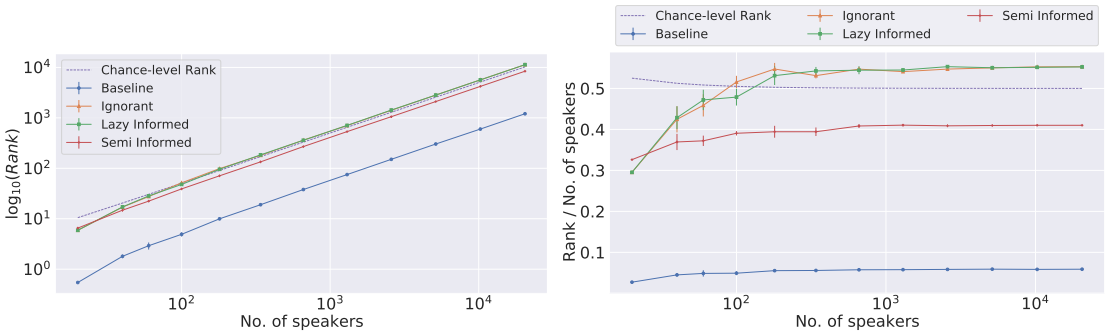
(a) Equal Error Rate (b) Linkability

Figure 12: Open-set speaker verification performance achieved by different attackers as a function of the number of speakers in the population.

results in terms of binary decision metrics (EER and Linkability) as in Section 4.2.4. The baseline EER on original speech starts at 7% and increases with the number of speakers until it converges to 12%. The three attackers perform quite poorly on anonymised speech: the EER achieved by the *Ignorant* and *Lazy-Informed* attackers starts at 37% and quickly stalls at 50%, while the *Semi-Informed* attacker achieves a consistent EER of 47% irrespective of the number of speakers. This observation indicates that the measurement of EER becomes more reliable with a larger number of verification trials. Hence the previous experiments with a small number of speakers can be considered as a lower bound on privacy, while the privacy improves with more speakers in the population. The Linkability performance is consistent with the observations in EER except that the *Semi-Informed* attacker displays the lowest value at all steps.

As an alternative to the open-set speaker verification attack above, we perform closed-set identification and report metrics such as *Rank* and *top-k* membership performance. To so do, we compute the PLDA score for a particular speaker with all the speakers in the selected population (which usually contains the true speaker) and sort it in descending order. Ideally the true speaker should attain the highest score and be ranked first but, if anonymisation has been successful, its rank will greatly increase. We report the rank of the true speaker along with the top-k precision which indicates the binary presence of the true speaker within the initial k ranks. The rank and top-k metrics help us assess the relative difficulty of speaker identification that will be faced by the different attackers in the presence of a large population. We also plot the *chance-level* (topline) rank, that the expected rank when the attacker is selecting a speaker at random, which is given by $\mathbb{E}(R) = \frac{N_s+1}{2}$, with N_s the total number of speakers in the population. Since adding more speakers to the population naturally increases the rank, we also report the normalised rank, that is the absolute rank divided by the number of speakers in the population, to assess the bona fide improvement in privacy as a function of the number of speakers.

The results of this open-set speaker verification attack are reported in Figure 13. In Figure 13(a), we notice a steep rise in the value of absolute rank, i.e., a decline in the identification performance, on both original and anonymised data as the number of speakers increases. However the baseline performance remains well below the chance-level rank, even when there are thousands of speakers



(a) Un-normalised Rank

(b) Normalised Rank

Figure 13: Closed-set speaker identification performance achieved by different attackers as a function of the number of speakers in the population.

in the population, which indicates the distinctive characteristics of these speakers. All the attackers start with better performance than chance-level but soon converge very close to the chance-level rank. The normalised rank plot in Figure 13(b) resembles the EER plot in Figure 12(a). The identification performance obtained by the *Ignorant* and *Lazy-Informed* attackers quickly degrades and converges to a value worse than the chance-level normalised rank, while the *Semi-Informed* attacker maintains a consistent performance, which is a little better than chance-level.

We further study the top-k precision obtained by different attackers and compare them to their corresponding baseline performance. In Figure 14 we only show the results for the top-20 precision since it exhibits the general observation noticed in all the top-k results and realistically an attacker might look at the top-20 results when it wants to shortlist the probable speaker identities. We observe that the precision drops much faster after anonymisation as compared to the baseline i.e., no anonymisation. This indicates that after anonymisation it is much more difficult for an attacker to identify a speaker as compared to the baseline. In other words, hiding the identity of an *anonymised* speaker in a crowd of n speakers is equivalent to hiding the *original* speaker in a crowd of N speakers, and N increases with a much faster rate as compared to n.

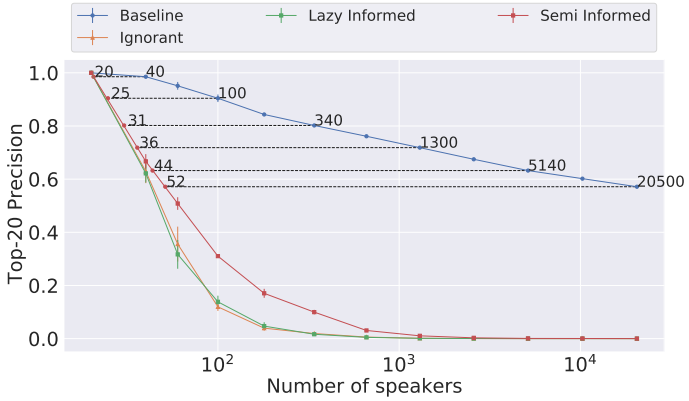


Figure 14: Top-20 precision of speaker identification for different attackers as a function of the number of speakers in the population. The numbers of speakers needed before anonymisation (N on blue curve) and after anonymisation (n on red curve) to achieve an equivalent drop in precision are highlighted.

4.2.6 Slicing utterances

In this section, we propose to further improve the privacy of speech data by reducing the length of the utterances. To do so, we anonymise the original utterances by means of x-vector based voice conversion and we cut them into segments of smaller lengths. By reducing the utterance duration, speaker verification becomes much more difficult, even when no anonymisation is performed. On top of that, the amount of personal information that can be extracted from the corresponding text is also reduced, since small parts of text are isolated from their context which is "hidden in the crowd" of all other parts of text. We observed that gathering small pieces of text back together to rebuild the initial sentences is an extremely difficult task. By removing contexts, small slices of

sentences could be interpreted in multiple ways, so that they lose their characterisation as personal data.

Slicing method In order to slice a dataset which contains utterances of various lengths into short utterances of a parameterised length δ, we make use of the text/speech alignment. For any given utterance, we iteratively take the first p words in order to obtain a new utterance whose length is at least δ.

Privacy/utility experiments When slicing, we need to manage the trade-off between privacy (shorter is better) and utility (longer is better). First of all, we explore what utterance lengths are acceptable in order to train the acoustic model of an STT system. Table 1 report the WERs achieved in four cases, depending on whether the training data has been anonymised or not before slicing and whether the test data has been anonymised or not. We use the anonymisation method previously introduced with the Dense proximity and Random gender selection. In each case, we present the results for different slicing lengths as well as for the initial length. As already mentioned above, we notice an increase in the WER when decoding anonymised data with an STT model trained on original data and vice-versa, which can be attributed to the mismatch between the training and test data. As for the effect of slicing itself, we notice that for original training data 1.5 s is the shortest possible duration: below that, the WER degrades significantly. With anonymised training data, the duration can also be shortened to 1.5 s when decoding anonymised speech but a duration of 3 s is preferable to maintain a reasonable utility when decoding original data.

Table 1: Utility achieved when training an STT model on sliced original or anonymised data.

Training data →	Original (Untransformed)				Anonymised			
Slicing	None	Sliced			None	Sliced		
Slicing length (s)		2	1.5	1		4	3	1.5
WER (%) Test Original	4.3	4.4	4.7	6.1	10.9	11.8	13.4	21.5
WER (%) Test Anonymised	7.6	7.6	8.0	11.4	4.9	4.9	4.9	4.9

In terms of privacy, we present in Figure 15 the results of speaker verification with utterances of different lengths. The setting of this experiment differs from Table 1. In this verification, the test data are sliced (instead of the training data). We put ourselves in the setting where the attacker tries to do a linkage attack on the data made accessible by the COMPRISE app. Also, the slicing in this experiment is conducted by doing hard cuts on the utterances (i.e., for each utterance we take the first x seconds) in order to explore what happens to utterances shorter than words. The results on original data illustrate the positive impact of slicing on the Linkability, especially for lengths lower than 1 s. Unfortunately, our utility experiment demonstrated that utterances shorter than 1 s are too short to train an STT system. Also, for values higher or equal to 1.5 s the level of privacy offered by slicing alone is insufficient. For this reason, slicing must be used together with anonymisation. In that case, the Linkability becomes close to zero when the duration of the utterance is shorter than 5 s.

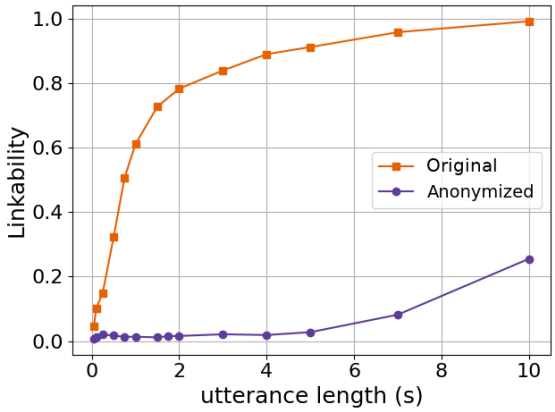


Figure 15: Privacy achieved on sliced original or anonymised data.

Reversibility of the slicing We can only benefit from the slicing method if the method is not reversible. Given a dataset with anonymised sliced utterances, we need to study the capacity of an attacker to distinguish whether two utterances are successive or not.

From the speech signal perspective, conducting speaker verification is already a hard problem for the attacker (Linkability close to zero). The task of "successive speech verification" is an even harder one. We have explored it without success so far. We think that, fortunately for the privacy requirements, this task is too difficult.

From the text perspective, we constructed an attacker that leverages language models in order to construct a "successive text score" that is similar to the linkage function for speaker verification. The language model helps estimate the probability of a given sentence $W = w_1 w_2 \dots w_n = w_{1:n}$ with w_i the i-th word. In practice, we used a 3-gram model:

$$P(w_{1:n}) = \prod_{k=1}^{n} P(w_k \mid w_{1:k-1}) \overset{\text{3-gram}}{\approx} P(w_1)P(w_2 \mid w_1) \prod_{k=3}^{n} P(w_k \mid w_{k-2:k-1}). \tag{5}$$

The attacker uses the language model in order to compute a score function between two sentences $W = w_{1:n}$ and $V = v_{1:m}$. The higher the score, the higher the chance that the sentences are successive. In our case, since we use a 3-gram model, only the last two words of the first sentence and the first two words of the second sentence are relevant:

$$SF(W, V) = P(w_{n-1} w_n v_1 v_2). \tag{6}$$

Similarly to speaker verification, using the distribution of scores, we can compute a *Text Linkability* metric $D_{\leftrightarrow}^{\text{sys}}$ for the binary task "successive or not". The results are presented in Table 2. We notice that they are all low (0.20 Text Linkability) and they are mostly independent of the slicing length. This is because we use a 3-gram model and utterances of length 1 s often have at least two words.

Overall, these results show that slicing is hardly reversible and sliced, anonymised utterances of 3–4 s length offer the best compromise between privacy and utility for STT acoustic model training.

Table 2: "Successive text verification" results.

Slicing length (s)	1	1.5	3	4
Text Linkability $D_{\leftrightarrow}^{\text{sys}}$	.20	.20	.20	.18

4.3 Text transformation

Besides speech transformations, COMPRISE has spent a considerable effort on protecting private information in text. In the context of voice assistants, this refers first and foremost to the utterances of the users of such assistants, i.e., the STT output. Private information about the user is not only present in the voice signal itself but of course, also in the spoken contents.

This section reports on advances of Work Package 2 in protecting speaker privacy by applying privacy-preserving text transformations. We begin by briefly reiterating the core achievements of our previous work which serve as a basis for our progress report. Since Deliverable D2.2, we have mainly focused on two extensions: support for multilingualism, and defences against the inference of certain speaker traits from text.

4.3.1 Replacement strategies

In Deliverable D2.2, we formulated different text transformation techniques in terms of *Differential Privacy* (DP) and compared their performance-privacy trade-off. Table 3 shows an example for the replacement strategies. More details are in our paper [Ade+20].

Redact Here, the private tokens are replaced with a non-word placeholder that is typically not part of the vocabulary of the original text e.g., `IIIII`.

Typed placeholder (a.k.a. **value-class membership** [THT04]) This is akin to using private category markers like LOCATION as the replacement token. This is a strategy similar to redaction, providing the same level of privacy. However, it provides additional information about a replaced token's category and might thus be more useful than redaction for certain Natural Language Processing (NLP) tasks.

Named placeholder A fixed category exemplar is used to replace all private tokens of that category [Pes+07], e.g, all locations are replaced by "London".

Word-by-word replacement We can distinguish between *value distortion* [THT04] if the replacement tokens are from an external source, and *value dissociation* [THT04] when the surrogate tokens are from the same corpus. The latter keeps the distribution of tokens in the resulting document unchanged, which might be relevant for some tasks.

Full entity replacement: Text coherence could be improved if original tokens were consistently replaced by the same surrogates. Another downside of the word-by-word strategy is that multi-word expressions could lead to nonsensical replacements, e.g., "Frankfurt Airport" could be transformed

Table 3: Examples of the replacement strategies, using color codes for PER, LOC, ORG, and TIME.

Replacement strategy	**Transformed text**
No Replacement	Hi Mister Miller, the Lufthansa flight from Frankfurt Airport to Rome is leaving by six pm
Redact	Hi Mister IIIII, the IIIII flight from IIIII to IIIII is leaving by IIIII
Typed-Placeholder	Hi Mister PER, the ORG flight from LOC to LOC is leaving by TIME
Named-Placeholder	Hi Mister Smith, the SAP flight from London to London is leaving by afternoon
Word by word	Hi Mister John, the BOSCH flight from New Boston to Berlin is leaving by eleven morning
Full entity	Hi Mister John, the BOSCH flight from New York to Berlin is leaving by twelve pm

to "New Francisco". A variant is thus to replace full entities instead of single words. The expected gain in coherence might benefit downstream tasks.

4.3.2 Towards multilingual NER

We have extended the Text Transformer from the English language to eight other European languages using available NER datasets. We have used the following datasets: CoNLL2002 [Tjo02] (Dutch and Spanish), CoNLL2003 [TD03] (English and German), Latvian [Gru+18], Italian [Mag+06], French [Neu16], and Portuguese [San+06]. With the success of multilingual transformer models [Dev+19; Con+20] like BERT on several tasks, we are able use the same NER model for several languages with good performance. Multilingual BERT supports over 100 languages including most of the European languages.

Table 4 compares the performance of the individual monolingual NER and multilingual corpus (aggregation of all language training data). The models were trained by *fine-tuning* multilingual BERT end-to-end on the available NER corpus. For all languages, the multilingual model has competitive F1-score as the monolingual NER model. On average, the multilingual model is better with around 0.3% F1-score improvement. The advantage of this approach is that we have a single model for all languages.

4.3.3 Protecting speaker traits

In Deliverable D2.2, we showed that sensitive user attributes like gender and age can be predicted from all speech transcripts of a given user. Here, we evaluate how much a single utterance of a given user is predictive of gender or age. Also, we apply a text anonymisation model based on the *paraphrasing* approach using BART [Lew+20], a text generation model.

Language	Monolingual	Multilingual
German (DE)	84.0	**84.82**
English (EN)	92.35	**92.82**
Spanish (ES)	87.65	**87.87**
French (FR)	**75.64**	74.79
Italian (IT)	83.37	**84.17**
Latvia (LV)	**86.20**	85.65
Dutch (NL)	91.65	91.62
Portuguese (PT)	74.33	**75.72**
Average	84.40	**84.68**

Table 4: Comparing the F1-scores obtained from the Monolingual and Multilingual (i.e., aggregating all language training data) NER models. Model based on fine-tuning multilingual BERT

Table 5: Number of sentences in each of the demographic attributes in the VERBMOBIL dialogue corpus

Gender	number of sentences
Female	2,648
Male	8,006

Age Groups	number of sentences
Young (<= 21)	6,453
Old (> 21)	4,201

Dataset For this experiment, we use the English VERBMOBIL corpus [Wah00] with 726 dialogues and 304 users. We exclude users without demographic information and sentences with less than five words. This results into 193 users and 10,654 sentences. Table 5 shows the number of sentences by gender and age where $2,648$ utterances are spoken by `females` and $8,006$ by `males`. We choose an age threshold of 21 to split the sentences into two age groups: `young` (age <= 21) and `old` (age > 21), with 6,453 and 4,201 speakers respectively.

Problem scenario We consider two possible scenarios for an attacker to predict demographic attributes: (1) *Using original data that has been previously collected to train a model to predict the gender or age of new users' utterances.* (2) *Using anonymised data aggregated from many users for training a demographic classification model.* We evaluate the success of the classification models when they are tested on either original or anonymised users' utterances.

Paraphrase model We employ the BART generation model to train a paraphrase model. BART is popular for training text generation tasks with labelled data, such as summarisation, dialogue response generation, and Q&A tasks. We trained a paraphrase model using an aggregated corpus collected from *Paranmt filtered* [KWI20] (created using back-translation of machine translation models), the *Microsoft paraphrase detection corpus* [DB05], *PAWS* [ZBH19], and the *Quora questions detection corpus* [GT17] (only 45 k sentences with more than 50 words). For the detection corpus, we only make use of the positive examples. We trained a paraphrase model by *fine-tuning*

the BART transformer model on the paraphrase pairs of sentences. On a test set, we obtained a ROUGE-1 score of 63.7 and a ROUGE-2 score of 42.1, where ROUGE-1 measures the overlap of unigrams between the model's output and the reference sentence and ROUGE-2 measures the overlap of bigrams between the model's output and the reference sentence (the higher, the better).

Results For the classification models, we divide the VERBMOBIL corpus for gender and age into a training/test split with a 75%/25% ratio. Table 6 shows the F1-score achieved in the two different attack scenarios on VERBMOBIL. If the classifier is trained on original data, paraphrasing can reduce the inference of demographic attributes significantly with −8% and −5% drops in F1-score for gender and age, respectively. However, if the attacker knows that the user is sending anonymised data to the cloud and trains a demographic classification model on such data, the gain obtained through paraphrasing is less than 2%.

Table 6: F1-score achieved by the demographic classification models.

Test set	**Gender**	**Age**
Original training data		
Original test set	70.3	65.4
Paraphrased test set	62.1	60.6
Anonymised training data		
Original test set	68.5	61.1
Paraphrased test set	66.7	60.5

4.3.4 Unintended memorisation in discriminative sequence models

In this section and in Section 4.3.5, we evaluate the need for text transformation as performed by our Text Transformer by inspecting learned models. Indeed, a known issue with neural network models is that they tend to memorise their training data [SRS17; Zha+17; Car+19]. Of particular interest is the effect of *unintended* memorisation, which occurs when models retain information that is orthogonal to the actual learning task. Memorisation raises severe privacy concerns in cases where such models are trained on datasets that contain sensitive information, such as e.g., credit card numbers, passwords, etc.

Carlini et al. [Car+19] define a metric, named *exposure*, that is based on comparing the perplexity $\mathrm{Px}(s)$ of a random phrase s inserted into the training set with the perplexities of other phrases from the same random space. The basic tenet is that a significantly lower perplexity of the inserted phrase vs. those of the other random phrases signals that the neural model has unintentionally memorised that phrase. Specifically, for a random phrase s inserted into the training set of a model θ, exposure is defined as:

$$\mathbf{exposure}_\theta(s) = -\log_2 \Pr_{r\in\mathcal{R}}[\mathrm{Px}_\theta(r) \leq \mathrm{Px}_\theta(s)] \tag{7}$$

where $\mathcal{R}$ is the random space of all such phrases. Note that high memorisation, i.e., low perplexity, is reflected by high exposure values. The authors test their definition empirically and conclude that

memorisation is not directly linked to overfitting but rather to the learning process itself, making memorisation a prevalent issue in state-of-the-art neural models. However, the authors' approach is limited to generative sequence models because the definition of exposure is based on perplexity. Therefore, we introduce a similar notion suitable for discriminative sequence models, using NER as a task for empirical evaluation [HKK20].

Our approach utilises ambiguous data points, i.e., words that could be labelled with different labels depending on the circumstances. For example in the case of NER, the word "Jordan" could refer to a person (e.g., Michael Jordan), a location (e.g., the country of the same name), or an organisation (e.g., The Jordan Company). Given a fixed phrase that has a word $\boldsymbol{s}$ with multiple possible class labels, we insert the s into the training set and train the model θ. d-exposure for label classes C_i is then defined as

$$\textbf{d-exposure}_\theta(s) = \frac{1}{N} \sum_{C_i} - \log_2 \Pr_{w \in C_i} [\text{conf}(w) \geq \text{conf}(s)] \tag{8}$$

where $\text{conf}(s)$ is the confidence returned by the model when labelling $\boldsymbol{s}$.

To test the effectiveness of this definition empirically, we perform a number of experiments on the CoNLL-2003 dataset [TD03]. Into that dataset, we insert a phrase s for which we choose the ambiguous format: "There are many people who like _____". This allows multiple entities to fill the blank. Focusing on three named entity labels, we insert the word "Williams" as S-ORG, "Chelsea" as S-PER and "Melbourne" as S-LOC. We chose these entities because their occurrences in the training set are more balanced than others. Table 7 shows the number of occurrences of these entities as each class. We found that the general behaviour of d-exposure does not change based on the chosen entities, as long as they are not highly imbalanced towards one class, nor does it change based on the format, as long as it is ambiguous.

Table 7: Number of occurrences of the chosen entities in the training set of CoNLL-2003.

Word	**S-PER**	**S-ORG**	**S-LOC**
Williams	7	8	0
Chelsea	5	6	0
Melbourne	0	4	5

For the model, we use a BiLSTM with GloVe embeddings, SGD optimiser, dropout (50%) and learning rate decay, implemented with Targer,[15] a neural tagging library [Che+19]. This model achieves an F1-score of 90.0% on the CoNLL-2003 dataset.

Repeated occurrences in the training set In the first experiment, we test whether d-exposure increases with the number of times the chosen phrase appears in the training set. The intuition is that, the more the model sees the sentence, the higher the incentive to memorise it. For this matter, we insert the chosen sentences 4, 16, 64, 128 and 256 times and observe d-exposure for each category. Figure 16 shows the effect of the number of repetitions of the inserted sentence on d-exposure. As expected, d-exposure generally increases with the number of repetitions, implying that repeated occurrence of a sentence in the training set tends to produce higher memorisation.

[15] `https://github.com/achernodub/targer`

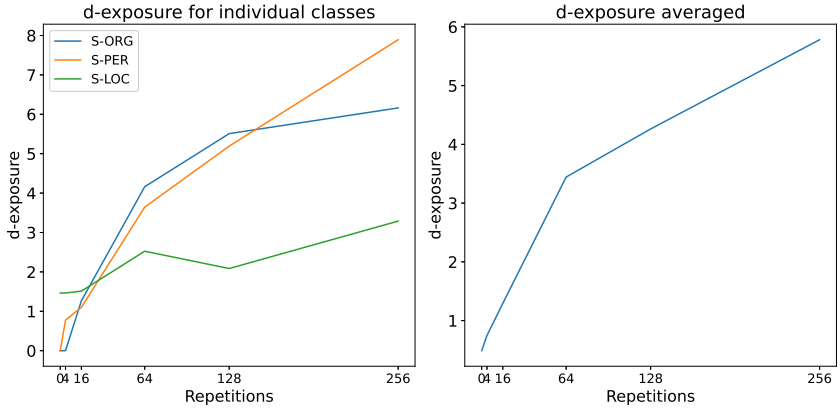


Figure 16: d-exposure vs. repetitions for individual classes and averaged on CoNLL-2003 at 150 epochs.

Table 8: d-exposure for the classes S-LOC, S-PER, and S-ORG for different numbers of repetitions (rows) and epochs (columns).

	S-LOC						S-PER						S-ORG					
	25	**50**	**75**	**100**	**125**	**150**	**25**	**50**	**75**	**100**	**125**	**150**	**25**	**50**	**75**	**100**	**125**	**150**
0	1.76	1.42	1.51	1.33	1.55	1.46	0.00	0.00	0.00	0.31	0.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	1.87	1.68	1.47	1.71	1.57	1.46	0.00	0.43	0.48	0.59	0.72	0.78	0.00	0.00	0.00	0.82	0.00	0.00
16	1.87	1.74	1.72	2.00	1.78	1.51	0.49	0.58	0.82	0.98	0.79	1.10	0.00	1.12	0.67	1.28	1.28	1.26
64	2.68	2.27	2.30	2.47	2.68	2.52	2.71	1.40	3.57	2.47	4.47	3.64	2.38	3.14	2.96	2.89	4.16	4.16
128	3.27	3.00	3.50	2.51	2.09	2.09	4.61	4.50	3.94	4.76	4.80	5.19	2.24	4.11	5.01	4.24	4.68	5.51
256	4.52	3.03	3.43	3.57	3.83	3.29	8.89	8.31	8.31	9.89	7.89	7.89	3.46	4.21	6.16	5.21	6.06	6.16

Another observation is that d-exposure does not behave equally in all classes. Rather, it is much lower for S-LOC than the other two. This validates our claim that exposure is to be measured per-class as different classes occur in different contexts but the exact reasons for the differing behaviour require further investigation. In additional experiments with other model architectures not detailed here, we found the same general trend in the curves but the behaviour of S-PER and S-LOC reversed. Table 8 shows d-exposure evaluated at different epochs (columns) and number of repetitions (rows) for the three classes. The first row is the value of d-exposure when the phrase is not inserted in the training set.

Overfitting In a second experiment, we observe the behaviour of d-exposure against overfitting. We conduct this analysis to confirm that exposure is not a measure of overfitting but rather of memorisation. If it were a measure of overfitting, we would expect it to reach its maximum value for all classes when overfitting begins or to keep increasing while the model is overfitting. To make the model overfit, we train it only on 10% of the training data, increase the number of epochs to 250 and disable learning rate decay and dropout. Figure 17 shows the results when the phrases are repeated 16 times. d-exposure increases as the model is learning and stops increasing when overfitting begins. In addition, maximum d-exposure for S-LOC (8.0) or S-ORG (8.1) is not

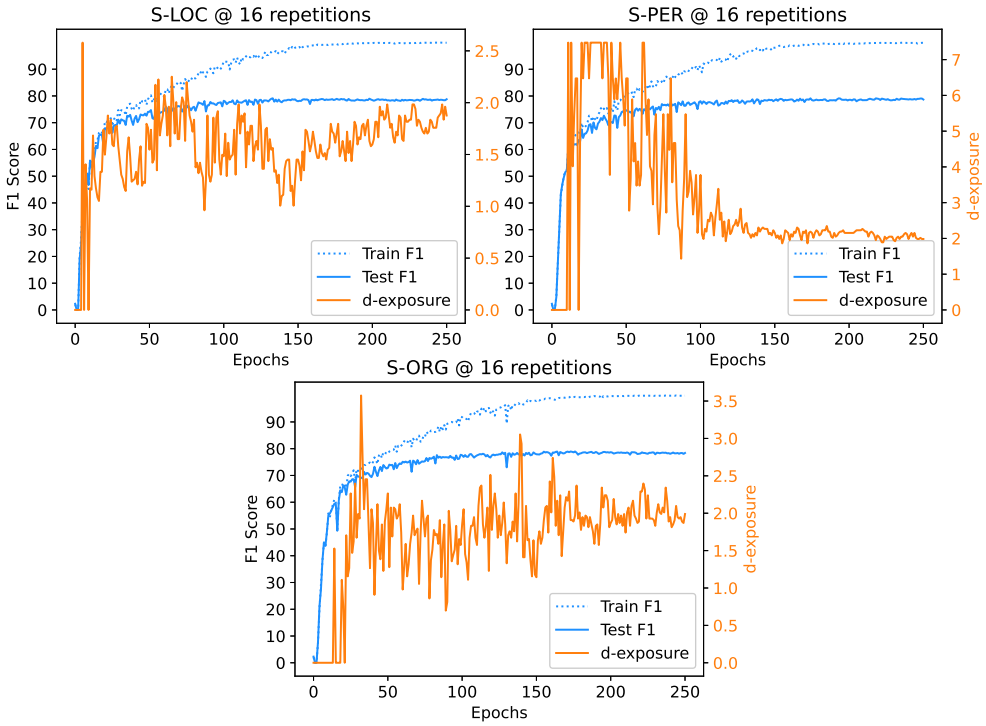


Figure 17: d-exposure vs. overfitting for S-LOC, S-PER, and S-ORG on CoNLL-2003.

reached at any point. Recall that the maximum d-exposure for a class C_i is $\log_2 |C_i|$, where $|C_i|$ is the number of entities belonging only to that class. For S-PER, however, maximum d-exposure (7.5) is reached only at stages where the model has not yet overfitted. Therefore, we conclude that d-exposure is not correlated with overfitting and, for the case of S-PER, the model has higher memorisation. Similar results were found for different numbers of repetitions.

In summary, our definition of d-exposure allows us to confirm Carlini et al.'s findings for discriminative sequence models. In particular, we observe 1) higher d-exposure values for repeated insertions of a test phrase into the training data; and 2) independence of d-exposure from model overfitting. The first finding confirms that the number of occurrence of a phrase in the training data, the expected memorisation of that phrase in the model, and the proposed metric all correlate positively. The second finding sets our approach apart from methods such as membership inference attacks (see [Tru+19]) which are prone to significant performance drops for overfitted models.

4.3.5 Impact of pre-trained word embeddings on memorisation in neural networks

In this section, we aim to quantify the leakage of sensitive information in pre-trained word embeddings, namely GloVe, ELMo, and BERT when they are used for downstream NLP tasks. Recently,

the leakage of sensitive information has been studied for text generation tasks [SS19] like machine translation, dialogue generation and language modelling. This leakage can also be viewed as neural networks memorising secret information which was proven to be true [Car+19]. A simple attack on a language model is predicting sensitive information like credit card number when given the context in which the secret information appears, this is even more probable when we limit the space of most likely words, say from all words in the vocabulary to only numbers. If indeed word embeddings leads to better performance on many NLP tasks including language modelling, does it also make this attack easier?

Computing the amount of sensitive information captured by word embeddings without using them to train an NLP task is not straightforward. So, we have made use of a simple language modelling task with these word vectors as input features. We address the problem of quantifying the leakage of sensitive information in pre-trained embeddings by investigating if they exacerbate the problem of memorisation in a language model when used as input features. We quantify the amount of information memorised by neural networks or exposed at inference using the *exposure* metric proposed by [Car+19]. Specifically, we compare the exposure of sensitive information on using different kinds of embeddings: distributed embeddings obtained by GloVe [PSM14] and contextualised embeddings, i.e., ELMo [Pet+18] and BERT [Dev+19]. In our experiments, we observe that the leakage in higher-dimensional word vectors is greater than or equal to the leakage observed in lower-dimensional vectors. This is particularly concerning because oftentimes, higher-dimensional embeddings yield better performance when used as features for downstream NLP tasks [Dev+19; Rad+18]. Training differentially private language models [McM+18] helps to drastically reduce the exposure of private information, thus providing better privacy [Car+19].

Measuring memorisation of secrets Following the approach introduced by Carlini et al. [Car+19], we analyse the effect of different word representations on memorisation. We make use of an LSTM language model to compare the levels of exposure while using different pre-trained embeddings as input features.

First, we augment the training data with an additional sequence with a secret such as "my PIN number is 2467 " i.e., $\mathcal{D} \leftarrow \mathcal{D} \cup s[r]$. For multiple insertion of secrets, we insert multiple sequences, $s[r_k]$ with K different PIN numbers, where $s[r_1] = s[r]$. Next, we obtain pre-trained embeddings for all the training sequences. We represent all the input embeddings with Z which is passed into the LSTM model. The trained weights θ of the LSTM model are learned after training and used to compute the log-perplexity $\tau_{\hat{r}}$ for all the possible secret values $\hat{r} \in R$. More details can be found in our paper [Tho+20].

Experimental setup We conduct two sets of experiments on the Penn Treebank (PTB) dataset for different word representations: GloVe, ELMo and BERT. We train a 1-layer LSTM language model with 256 hidden units on 2,000 sentences from the PTB text data that consists of 35,000 sequences (assuming a minimum context size of one) and 12,921 vocabulary words. The first set of experiments is trained on the dataset augmented with secret sequence(s) using the Adam optimiser [KB15], while the second set of experiments helps to reduce memorisation using Differentially Private Stochastic Gradient Descent (DPSGD) (see Section 4.4). We consider pre-trained embeddings with various dimensions $d = 100, 300, 768, 1024$, i.e., GloVe-{100d, 300d}, ELMo-1024d, and BERT-{768d, 1024d}.

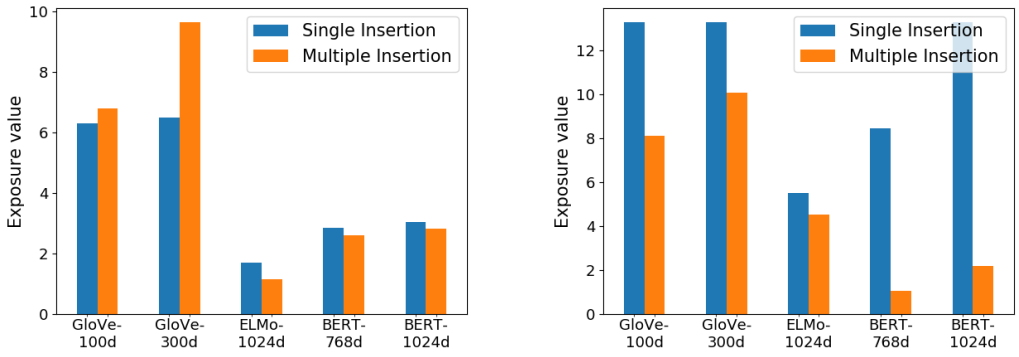
To study how the length of a secret affects its memorisation, we use two types of secrets, namely *single word - disease* and *four digit - PIN*. The *disease* type of secret is inserted as follows:

"< name > is suffering from < disease >". For all the experiments with this type of secret augmented, we compute the exposure value of the sequence $s[r]$ *"john is suffering from alzheimers"* after training the model on a dataset including this sentence. Since the number of diseases is too small to be considered as sample space, we assume the vocabulary as the sample space from which the diseases are drawn from. In the *PIN* type of secret, the inserted secret is of the form *"< name > atm PIN number is < #### >"*. Here, the sample space size for computing the exposure is 10^4.

The memorisation could also be affected by the presence of multiple secrets in the dataset, which may confuse the model. In order to analyse the effect of having multiple secrets, we use two types of insertion of secrets:

- **Single insertion** with a pair of secrets: in this case, we augment the dataset with a single sentence that contains either the *disease* or *PIN* type secret.
- **Multiple insertion** with unique pairs of secrets: in this case, we augment the dataset with multiple sentences all of which contains either *disease* types or *PIN* types. For example, we test the exposure of the sequence $s[r]$ after augmenting the dataset with M additional secrets like *"oliver is suffering from influenza"*, *"laura is suffering from cholera"*, etc.. In the experiments, $M = 16$ for the *disease* type and $M = 10$ for the *PIN* type secret.

Results Figure 18 shows the exposure values for different kinds of embeddings for single and multiple insertion of secrets. We observe a pattern between the exposure value and the embedding dimension regardless of the secret type, or insertion type for GloVe and BERT embeddings. Higher exposure levels are observed for higher dimensions except when the exposure values were already maximum. This indicates that for the same embedding type, representations with higher dimensions may memorise more. We also observe that multiple insertion of secrets decreases the exposure values except for GloVe embeddings with the *disease* secret type. This suggests that the presence



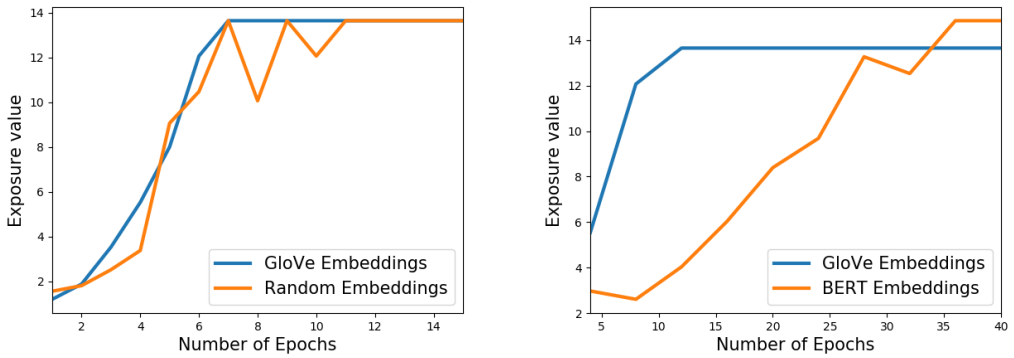
(a) Secret-type: disease

(b) Secret-type: 4-digit PIN

Figure 18: Exposure values for single insertion of a sequence with a pair of secrets and multiple insertion of sequences with unique pairs of secrets. The number of epochs for the disease and the PIN type secrets is 5 and 40, respectively.

of multiple instances of the same type of secret could confuse the model and helps lower the exposure levels. The length of the secret was also found to affect the memorisation. One interesting finding during the experiments was that the length of the secret affects the stage of training when the exposure values reach the maximum. For the *disease* type, the exposure values were already maximum at 40 epochs unlike in the case of *PIN*. This is the reason for the lower number of epochs for the *disease* type of secret in Figure 18(a).

Figure 19 shows the exposure levels of random GloVe embeddings and pre-trained BERT embeddings at different stages of training. Evidently, the memorisation in the case of GloVe saturates much earlier in training compared to BERT. The memorisation in BERT representations is seen to happen later in training, reaching the maximum exposure only after the 36th epoch as compared to the 12th epoch in the case of GloVe. The exposure values in the case of GloVe embeddings reach their maximum value earlier than in the case of random embeddings. This shows that, although pre-trained embeddings result in improved performance over random embeddings, they are at a higher risk of exposing sensitive information in the training dataset.



(a) GloVe vs Random embeddings

(b) GloVe vs BERT embeddings

Figure 19: Exposure values for random, GloVe and BERT embeddings at different stages of training for single insertion of a disease-type secret.

Lastly, Table 9 report the results achieved by training the LSTM models with DPSGD (with parameters $\varepsilon = 10$, $\delta =$ 2e-5, resulting noise level $\sigma = 0.44$). We observe a drastic reduction in exposure values, especially for models with maximum exposure (BERT-768d and BERT-1024d) from 14.85 to less than 2.21 for both single insertion and multiple insertions of *disease*-type secrets. Our observation confirms Carlini et al.'s [Car+19] observation that differential privacy helps in reducing memorisation. Note however that DPSGD is considerably slower than standard Adam, e.g., training the non-DP and DP models using GloVe embeddings takes 12 min and 14 h, respectively, using one Nvidia Titan X GPU. All the reported exposure values have a maximal standard deviation of 1.09.

Conclusion In summary, we investigated memorisation in word representations commonly used as features for training the state-of-the-art natural language understanding tasks. We compare the degree of memorisation of three different word embedding types (GloVe, ELMo and BERT).

Table 9: Exposure values of the *disease* type of secret for different embedding types with differentially private (DP) vs. non-DP training. The number of epochs for both versions is 40.

	Single Insertion		Multiple insertion	
Embedding Type	**Non-DP**	**DP**	**Non-DP**	**DP**
GloVe-100d	5.87	1.61	7.9	1.72
GloVe-300d	6.03	1.17	13.65	2.19
ELMo-1024d	13.39	2.04	9.33	0.65
BERT-768d	14.85	0.12	14.85	1.62
BERT-1024d	14.85	0.27	14.85	2.20

All the embedding types were found to expose sensitive information up to a certain extent. This observation implies a possible privacy threat when they are used in applications with private and sensitive data. We observed an increase in the exposure levels (except when the exposure value is already maximum) with the embedding dimension for GloVe and BERT, and multiple instances of the sensitive information in the training dataset is seen to lower memorisation. Further, we observed that different embedding types start memorising at different stages of training. The GloVe embeddings were found to reach maximum exposure level earlier in training compared to random embeddings of the same dimension.

As a conclusion, we empirically prove the need of performing private word substitution as we propose in the Text Transformer. Indeed, even if we consider scenarios where we do not release data publicly but just only models learned from clean data, privacy leaks can occur. The next section is about our recent research work that proposes an efficient solution to remedy this issue. This contribution goes beyond the core COMPRISE scenario by complementing the privacy-preserving approaches presented above, and is generally applicable to neural network-based applications.

4.4 Improving privacy during learning

In some situations, for instance when privacy transformations lead to unacceptable performance drops in subsequently trained models, alternative approaches to privacy are required. However, as Carlini et al. have shown, even when an attacker does not have access to the training set, simply deploying a model trained on that dataset might give the attacker access to private information contained therein [Car+19]. Carlini et al.'s work was limited to neural generative sequence models. Here, we extend this work to discriminative sequence models as well.

DP training methods (see, e.g., [Yan+17]) offer a possible remedy. In practice, DP algorithms are obtained from non-private algorithms by means of appropriate randomisation [DR+14]. DP has been integrated into deep learning [SS15; Aba+16]. The proposed method in [Aba+16] is based on clipping gradients and adding random noise to them at each iteration of stochastic gradient descent (SGD). Combined with a *moments accountant* method for tracing the privacy loss, this DPSGD technique has enabled deep neural networks to be trained under a modest privacy budget at the cost of a manageable reduction in the model's test accuracy. However, for low privacy budgets (i.e., small ε, see below), which corresponds to a large privacy guarantee, the accuracy drops significantly.

We investigate the impact of batch normalisation (BN) [IS15] and layer normalisation (LN) [BKH16] on the performance of DPSGD training. These normalisation layers are key ingredients in

modern deep learning: they make it possible to robustly train deep neural networks without carefully designing the nonlinearities [Kla+17] or choosing a specific initialisation scheme [SZ15]. They also improve generalisation by preventing overfitting when training very deep networks [ZDM19].

In particular, we show that BN can be integrated with DPSGD without any additional loss in privacy during the training process. We compare our proposal with the current state-of-the-art methods by conducting a series of experiments on Computer Vision and NLP tasks. In summary, our contributions are as follows:

- We propose an efficient method for using BN layers without incurring an additional privacy loss in the training procedure. To the best of our knowledge, our work is the first to apply a DP mechanism in the presence of BN.
- We establish new accuracy records for DP trained deep networks on the MNIST and CIFAR10 datasets.

DPSGD and normalisation DP is a systematic approach to quantifying privacy while querying a dataset. DP protects privacy by bounding the influence of any sample on the outcome of queries. It provides a provable guarantee for individuals. We proceed by briefly recalling some preliminaries on DP and then propose our approach for training deep neural networks in a DP way.

Let us denote the domain of data points by χ. We call two datasets $D_1, D_2 \in \chi$ *neighbouring* if they differ exactly in a single data point, i.e., $d(D_1, D_2) = 1$, where $d(.,.)$ is the Hamming distance.

Definition 4.1. *(Differential Privacy [Dwo+06]). A randomised algorithm $\mathcal{M} : \chi \to \mathcal{R}$ with domain χ and range $\mathcal{R}$ is (ε, δ) differential private if for all measurable sets $S \in \mathcal{R}$ and for all neighbouring datasets D_1 and D_2, it holds that*

$$\Pr[\mathcal{M}(D_1) \in S] \leq \exp(\varepsilon) \Pr[\mathcal{M}(D_2) \in S] + \delta. \tag{9}$$

Intuitively an (ε, δ)-differentially private mechanism guarantees that the absolute value of privacy leakage will be bounded by ε with probability at least $1 - \delta$ for adjacent datasets. The higher the value of ε, the greater the information leakage.

A standard approach for achieving DP is to add some random noise r to the output of queries, $q(D) + r$, and to tune the noise r by the *sensitivity* of the query. L_p sensitivity is defined as the maximum change in the outcome of a query for two neighbouring datasets and measures the maximum influence that a single data point can have on the result of the query:

$$S_p = \max_{d(D_1, D_2)=1} \|q(D_2) - q(D_1)\|_p. \tag{10}$$

The special case where r is calibrated with S_2 sensitivity and sampled according to the normal distribution is of special importance and is termed as Gaussian mechanism:

$$G(D) \colon = q(D) + \mathcal{N}(0, S_2^2 \sigma^2). \tag{11}$$

Here $\mathcal{N}(0, S_2^2 \sigma^2)$ is the normal distribution with mean zero and standard deviation $S_2 \sigma$. It can be shown that this mechanism satisfies (ε, δ)-DP provided that $\sigma \geq \frac{\sqrt{2 \ln \frac{1.25}{\delta}}}{\varepsilon}$ [DR+14].

DP has been integrated into deep learning in [SS15] and subsequently in [Aba+16] for the setting where an adversary has access to the network architecture and learned weights, $f(\boldsymbol{\theta}^*, .)$. In particular, [Aba+16] preserves privacy by adding noise to the SGD updates:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \, \mathbf{g}_t + \frac{\eta}{L} r, \tag{12}$$

where $\mathbf{g}_t$ is the averaged gradient, η is the learning rate and r is sampled according to the Gaussian distribution $\mathcal{N}(0, \sigma^2)$. To control the influence of training samples on the parameters, the gradients are clipped by the L_2-norm

$$\pi(g_i) = g_i \,.\, \min(1, C/\|g_i\|_2), \tag{13}$$

where g_i is the gradient corresponding to the i-th sample and C is the clipping factor. It has been shown in [Aba+16] that each step of DPSGD is (ε, δ)-differentially private once we tune the noise as $\sigma = C\,z$ with $z = \frac{\sqrt{2\ln\frac{1.25}{\delta}}}{\varepsilon}$.

It is known that BN is not consistent with DP training. Indeed, in a non-private setting, one usually keeps track of the running averages of mean and variance statistics during the training procedure and reuses this collected information at test time to normalise the inputs to neurons. More specifically, the update rule for running averages at each iteration is as follows:

$$\boldsymbol{\mu}^{\text{Batch}} \leftarrow (1-\alpha)\boldsymbol{\mu}^{\text{Batch}} + \alpha\,\boldsymbol{\mu}_t^{\text{Batch}}\,, \tag{14}$$

$$\boldsymbol{\sigma}^{2,\text{Batch}} \leftarrow (1-\alpha)\boldsymbol{\sigma}^{2,\text{Batch}} + \alpha\,\boldsymbol{\sigma}_t^{2,\text{Batch}}, \tag{15}$$

where $(\boldsymbol{\mu}^{\text{Batch}}, \boldsymbol{\sigma}^{2,\text{Batch}})$ are the estimated mean and variance statistics, $(\boldsymbol{\mu}_t^{\text{Batch}}, \boldsymbol{\sigma}_t^{2,\text{Batch}})$ are new observed values at iteration t of training, and α is the momentum of moving averages.

Since these running averages are also part of the model's output, in private training we have to add noise to the statistics at each iteration, and distribute the privacy budgets among the weights and moving averages to make the overall procedure differentially private. Additionally, we need to truncate the summed inputs of neurons to bound the sensitivity of means and variances, which are given by [Swa+19]:

$$S_\mu = \frac{C'}{L}, \tag{16}$$

$$S_{\sigma^2} = C'^2\left(\frac{3}{L} - \frac{3}{L^2} + \frac{1}{L^3}\right), \tag{17}$$

where C' is the clipping threshold for neuron activations and L is the batch size. Empirically, we have found that if we tune the noise with the worst-case scenario according to the above sensitivities, the performance of the model drops drastically. Therefore we employ a more sophisticated approach to deal with BN situation as shown in Algorithm 1.

First of all, we do not track the running averages during the training phase and instead freshly computed statistics from the current batch are used to normalise the neurons. But to be able to deal with BN at test time we concatenate a fixed amount of data points $\hat{X}$ with size M taken from a public dataset (disjoint from the training data) to the input of the network, both in the training and the test phase. These samples only contribute to the statistics and not to the cost function directly.

Therefore, in the training phase, the cost is computed via $\mathcal{L}(\boldsymbol{\theta}_t, X \cup \hat{X})[:L]$, where X is a batch of size L from the training data, and $[:L]$ denotes the slice of the first L elements. At test time, when iterating over the dataset, the same public data points $\hat{X}$ are also concatenated to each test sample, x, and the output of network is computed as $f(\boldsymbol{\theta}_T, x \cup \hat{X})[0]$. This leads to privacy preserving BN and allows us to compute the normalisation statistics over a batch of size $1 + M$ without any reference to the training data.

Algorithm 1: DPSGD with Batch Normalisation: Training

Input: dataset $\mathcal{D} = \{(x_1, y_1), \cdots\}$ of size N, a public dataset $\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_M\}$, loss function $\mathcal{L}(\boldsymbol{\theta}, .)$, learning rate η_t, noise multiplier z, sample size L, gradient norm bound C, number of iterations T.

for $t = 0$ **to** $T - 1$ **do**

- Take a random sample, $X \sim \mathcal{D}$, with size L and selection probability $\frac{L}{N}$.
- Concatenate the public data to each lot
  $X \leftarrow X \cup \hat{X}$
- Compute lost for the first L elements
  $\mathcal{L}(\boldsymbol{\theta}_t, x_i) = \mathcal{L}(\boldsymbol{\theta}_t, X)[i]$
- Compute gradient
  $\mathbf{g}_t(x_i) \leftarrow \nabla_{\boldsymbol{\mu}_t} \mathcal{L}(\boldsymbol{\theta}_t(\boldsymbol{\mu}_t), x_i)$.
- Clip gradient
  $\mathbf{g}_t(x_i) \leftarrow \mathbf{g}_t(x_i)\ .\ \min(1, C/\|\mathbf{g}_t(x_i)\|_2)$
- Add noise
  $\mathbf{g}_t \leftarrow \frac{1}{L}\big(\sum_i \mathbf{g}_t(x_i) + \mathcal{N}(0, C^2 z^2)\big)$
- Update parameters:
  $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta_t\ \mathbf{g}_t$

Test Phase

Input: test dataset $\mathcal{D}_{test} = \{(x_1, y_1), \cdots\}$ of size N_{test}, the public dataset $\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_M\}$, trained model $f(\boldsymbol{\theta}_T, .)$.

- Initialise Y $\leftarrow \emptyset$

for $i = 0$ **to** $N_{test} - 1$ **do**

- Concatenate the public dataset to each data point x_i
  $X \leftarrow x_i \cup \hat{X}$
- Compute the network output corresponding to the data point x_i
  $y_i = f(\boldsymbol{\theta}_T, X)[0]$
- Append y_i to the results
  $Y \leftarrow Y \cup y_i$

end for

Return outputs of network Y.

end for

Experiments We evaluate the impact of normalisation layers on DP training using two image recognition benchmarks, MNIST and CIFAR-10, as well as a text classification task using the AG News Corpus. The purpose of these experiments is two-fold: we show that (1) similar to non-private training, normalisation layers improve the performance of models trained with DPSGD, and (2) training very deep networks is feasible with our private version of BN.

All models, as well as DPSGD, have been implemented in PyTorch [Pas+17]. To track the privacy loss over the whole training procedure, we employ the Rényi-DP technique [Mir17], which provides a tighter bound on the privacy loss compared to the strong composition theorem [DRV10]. We use the open-source implementation of the Rényi DP accountant from the TensorFlow Privacy

package.[16] The total privacy loss ε is computed as a function of the noise multiplier z, the dataset size N, the lot size L, the number of iterations T, and δ.

Table 10 shows the accuracy of the LeNet-5 model on the MNIST test set for ε ranging from high to very low privacy budgets. We have trained LeNet-5 with and without normalisation layers, using DPSGD. For training the model augmented with BN, we employed 128 images of KMNIST [Cla+18] as the public dataset. The probability δ parameter is set to 10^{-5} in all our experiments. The results illustrate that the use of normalisation layers consistently improves the performance of DPSGD for all finite values of privacy loss. Further, we observe that the effect of BN is greater than that of LN. Remarkably, we gain around 7% and 10% for very low privacy budgets of $\varepsilon = 0.1$ and $\varepsilon = 0.05$ with our private BN technique.

Table 10: Classification accuracy on the MNIST test set with $\delta = 10^{-5}$.

DP Algorithm	**privacy budget (ε)**						
	∞	7	3	1	0.5	0.1	0.05
DPSGD (LeNet-5)	99.20%	97.01%	96.34%	94.11%	91.10%	83.00%	78.96%
DPSGD (LN-LeNet-5)	99.20%	97.35%	97.05%	96.68%	94.81%	87.45%	75.76%
DPSGD (BN-LeNet-5)	99.20%	**98.68%**	**98.18%**	**97.61%**	**96.83%**	**90.68%**	**88.15%**

We now turn our attention to the CIFAR-10 dataset. Table 11 summarises the results of DPSGD on the TensorFlow tutorial model considered in [Aba+16]. We follow the same experimental setting as in [Aba+16], i.e., we fine-tune the linear layers of a pre-trained model trained on the CIFAR-100 dataset. For training models with BN, we also use 128 images of CIFAR-100 as our public dataset, which has completely different image examples and classes from those of CIFAR-10. As Table 11 illustrates, using BN results in better accuracy than LN as well as the original TensorFlow tutorial model. We have also shown the results of training a lightweight VGG model in this table. The non-private accuracy of this model is comparable with that of the TensorFlow tutorial model but it leads to a much lower gap for finite privacy budgets. It should be mentioned that training such models without BN is infeasible since the training is extremely unstable. Therefore our privacy friendly BN technique allows training much deeper and complex networks and establishing new scores for the performance of DP models.

Table 11: Classification accuracy on the CIFAR-10 test set with $\delta = 10^{-5}$.

DP Algorithm	**privacy budget (ε)**			
	∞	8	4	2
DPSGD (TF-tutorial) [Aba+16]	80.0%	73.0%	70.0%	67.0%
DPSGD (LN-TF-tutorial)	80.0%	73.3%	70.6%	67.0%
DPSGD(BN-TF-tutorial)	80.0%	**74.1%**	**71.2%**	**69.8%**
DPSGD (BN-VGG)	80.7%	**79.5%**	**79.1%**	**77.4%**

Finally, we extend our experiments to a more complex natural language text classification task.

[16] `https://github.com/tensorflow/privacy`

For this, we trained a BiLSTM model with one linear/dense layer (DL) with/without LN on the AG News Corpus [17], a popular text classification dataset with 4 categories of news: *World*, *Sports*, *Business* and *Sci/Tech*. Table 12 shows the results of this experiment. As before, LN has a large impact on the performance of the model.

Table 12: Classification accuracy on the AG News Corpus with $\delta = 10^{-5}$.

DP Algorithm	**privacy budget (ε)**					
	∞	7	3	1	0.5	0.1
DPSGD (BiLSTM-DL)	88.47%	83.86%	80.00%	81.14%	77.88%	37.49%
DPSGD (LN-BiLSTM-DL)	88.18%	**84.34%**	**82.51%**	**82.03%**	**79.16%**	**50.09%**

5 Summary

In this report we have presented the results of Work Package 2. We have proposed Docker containers for the COMPRISE Voice and the Text Transformers together with RESTful services. Concerning the evaluation of the privacy of the learning branch, we have reported several contributions. We have presented an analysis of the privacy risks in voice-enabled services and proposed some data governance impacts. We have also showed how the COMPRISE suite answers the questions raised in this analysis.

On the speech processing side, we introduced VPC, an x-vector based voice conversion method. We rigorously evaluated how design choices and attacker's knowledge can impact privacy, using different privacy measures. We intensively evaluated the gains of privacy, by selecting the most adequate choices. We measured the privacy improvement that occurs when a larger population of users is considered. We also proposed to reduce the length of utterances, so as to decrease context and offer optimal privacy guarantees. The cost of introducing privacy is that the variability of speech is decreased and a small impact on the WER is observed. We are currently working at improving this, using both original (untransformed) and anonymised data in the learning phase.

On the text processing side, we have proposed different replacement strategies for named entities that carry private information. We used a technique based on randomised response to obtain privacy guarantees in the DP setting. We extended the text transformation techniques from the English language to eight other European languages using available NER datasets. We also provided directions to address more traits in speech.

As deep networks tend to memorise training data we proposed a set of evaluations in order to measure it and countermeasures using DP training. We introduced a new measure to evaluate the amount of (private) information memorised in deep neural networks for discriminative tasks. We evaluated the impact of the use of trained embeddings. Finally, we proposed an efficient method for using BN layers without incurring an additional privacy loss in the training procedure and show that this also impacts memorisation.

[17] `http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`

References

[Aba+16] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. ACM. Vienna, Austria: Association for Computing Machinery, 2016, pp. 308–318.

[Ade+20] D. I. Adelani, A. Davody, T. Kleinbauer, and D. Klakow. "Privacy Guarantees for De-Identifying Text Transformations". In: *Proceedings of INTERSPEECH 2020 – 21st Annual Conference of the International Speech Communication Association*. 2020, pp. 4666–4670.

[BKH16] J. L. Ba, J. R. Kiros, and G. E. Hinton. "Layer normalization". In: *arXiv preprint arXiv:1607.06450* (2016).

[Car+19] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. "The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks". In: *Proceedings of the 28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 267–284.

[Che+19] A. Chernodub, O. Oliynyk, P. Heidenreich, A. Bondarenko, M. Hagen, C. Biemann, and A. Panchenko. "TARGER: Neural Argument Mining at Your Fingertips". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 195–200.

[CJL21] P. Champion, D. Jouvet, and A. Larcher. "A study of F0 modification for X-vector based Speech Pseudo-anonymization across gender". In: *submitted to IEEE Spoken Language Technology Workshop* (2021).

[Cla+18] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. "Deep Learning for Classical Japanese Literature". In: *Proceedings of NeurIPS 2018 Workshop on Machine Learning for Creativity and Design*. Montréal, Canada, 2018.

[Con+20] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. "Unsupervised Cross-lingual Representation Learning at Scale". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 8440–8451.

[DB05] W. B. Dolan and C. Brockett. "Automatically Constructing a Corpus of Sentential Paraphrases". In: *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. 2005.

[Dev+19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019.

[DR+14] C. Dwork, A. Roth, et al. "The algorithmic foundations of differential privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[DRV10] C. Dwork, G. N. Rothblum, and S. Vadhan. "Boosting and differential privacy". In: *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 51–60.

[Due09] D. Dueck. "Affinity propagation: clustering data by passing messages". PhD thesis. Department of Electrical and Computer Engineering, University of Toronto, 2009.

[Dwo+06] C. Dwork, F. McSherry, K. Nissim, and A. Smith. "Calibrating noise to sensitivity in private data analysis". In: *Proceedings of the Theory of cryptography conference*. Springer. 2006, pp. 265–284.

[Fan+19] F. Fang, X. Wang, J. Yamagishi, I. Echizen, M. Todisco, N. Evans, and J.-F. Bonastre. "Speaker Anonymization Using x-vector and Neural Waveform Models". In: *Proceedings of the 10th ISCA Speech Synthesis Workshop*. 2019, pp. 155–160.

[Far+08] M. Farrús, M. Wagner, J. Anguita, and J. Hernando. "How vulnerable are prosodic features to professional imitators?" In: *Proceedings of Odyssey 2008: The Speaker and Language Recognition Workshop, Stellenbosch, South Africa, January 21-24, 2008*. ISCA, 2008, p. 2.

[Gru+18] N. Gruzitis, L. Pretkalnina, B. Saulite, L. Rituma, G. Nespore-Berzkalne, A. Znotins, and P. Paikens. "Creation of a Balanced State-of-the-Art Multilayer Corpus for NLU". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018.

[GT17] L. Guo and H. Tian. *Duplicate Quora Questions Detection*. http://semanticscholar.org. 2017.

[HKK20] M. Helali, T. Kleinbauer, and D. Klakow. "Assessing Unintended Memorization in Neural Discriminative Sequence Models". In: *Proceedings of the 23rd International Conference on Text, Speech and Dialogue (TSD 2020)*. Sept. 2020.

[Iof06] S. Ioffe. "Probabilistic linear discriminant analysis". In: *Proceedings of Computer Vision – ECCV 2006*. Springer. Berlin, Heidelberg: Springer, 2006, pp. 531–542.

[IS15] S. Ioffe and C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 448–456.

[KB15] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *Proceedings of the International Conference on Learning Representations*. San Diego, CA, USA, 2015.

[Kla+17] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. "Self-normalizing neural networks". In: *Advances in neural information processing systems*. 2017, pp. 971–980.

[KS08] C. Kim and R. M. Stern. "Robust signal-to-noise ratio estimation based on waveform amplitude distribution analysis". In: *Proceedings of the Ninth Annual Conference of the International Speech Communication Association*. 2008.

[KWI20] K. Krishna, J. Wieting, and M. Iyyer. "Reformulating Unsupervised Style Transfer as Paraphrase Generation". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 737–762.

[Lew+20] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 7871–7880.

[LZY07] K. Liu, J. Zhang, and Y. Yan. "High quality voice conversion through phoneme-based linear mapping functions with STRAIGHT for Mandarin". In: *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*. Vol. 4. 2007, pp. 410–414.

[Mag+06] B. Magnini, E. Pianta, C. Girardi, M. Negri, L. Romano, M. Speranza, V. Bartalesi Lenzi, and R. Sprugnoli. "I-CAB: the Italian Content Annotation Bank". In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. Genoa, Italy: European Language Resources Association (ELRA), May 2006.

[McM+18] B. McMahan, D. Ramage, K. Talwar, and L. Zhang. "Learning Differentially Private Recurrent Language Models". In: *Proceedings of the International Conference on Learning Representations*. Vancouver, Canada, 2018.

[Mir17] I. Mironov. "Rényi differential privacy". In: *Proceedings of 2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE. 2017, pp. 263–275.

[MJ20] Á. Moretón and A. Jaramillo. "How can private information recorded by voice-enabled systems be identified?" In: *European Data Protection Law Review* 6.3 (Oct. 2020), pp. 464–469.

[MJC21] Á. Moretón, A. Jaramillo, and C. Castillo. *Anonymisation and re-identification risk report*. Under submission. Jan. 2021.

[Neu16] C. Neudecker. "An Open Corpus for Named Entity Recognition in Historic Newspapers". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016.

[Pas+17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. "Automatic Differentiation in PyTorch". In: *Proceedings of NeurIPS 2017, Autodiff Workshop*. Long Beach, CA, USA, 2017.

[Pes+07] J. P. Pestian, C. Brew, P. Matykiewicz, D. Hovermale, N. Johnson, K. B. Cohen, and W. Duch. "A shared task involving multi-label classification of clinical free text". In: *Proceedings of Biological, translational, and clinical language processing*. Prague, Czech Republic: Association for Computational Linguistics, June 2007, pp. 97–104.

[Pet+18] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 2227–2237.

[PSM14] J. Pennington, R. Socher, and C. Manning. "Glove: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543.

[Qia+17] J. Qian, H. Du, J. Hou, L. Chen, T. Jung, X. Li, Y. Wang, and Y. Deng. "VoiceMask: Anonymize and Sanitize Voice Input on Mobile Devices". In: *arXiv preprint arXiv:1711.11460, 2017* abs/1711.11460 (2017).

[Qia+18] J. Qian, H. Du, J. Hou, L. Chen, T. Jung, and X.-Y. Li. "Hidebehind: Enjoy Voice Input with Voiceprint Unclonability and Anonymity". In: *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. SenSys '18. Shenzhen, China: Association for Computing Machinery, 2018, pp. 82–94.

[Rad+18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. *Language Models are Unsupervised Multitask Learners*. Tech. rep. OpenAI, 2018.

[Raj+19] D. Raj, D. Snyder, D. Povey, and S. Khudanpur. "Probing the Information Encoded in X-vectors". In: *Proceedings of 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. Sentosa, Singapore: IEEE, 2019, pp. 726–733.

[RBS14] J. Rohdin, S. Biswas, and K. Shinoda. "Constrained discriminative PLDA training for speaker verification". In: *Proceedings of ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 1670–1674.

[San+06] D. Santos, N. Seco, N. Cardoso, and R. Vilela. "HAREM: An Advanced NER Evaluation Contest for Portuguese". In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. Genoa, Italy: European Language Resources Association (ELRA), May 2006.

[SN03] D. Sundermann and H. Ney. "VTLN-based voice conversion". In: *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (IEEE Cat. No.03EX795)*. 2003, pp. 556–559.

[Sny+18] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. "X-vectors: Robust DNN embeddings for speaker recognition". In: *Proceedings of ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5329–5333.

[SRS17] C. Song, T. Ristenpart, and V. Shmatikov. "Machine Learning Models That Remember Too Much". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 587–601.

[SS15] R. Shokri and V. Shmatikov. "Privacy-preserving deep learning". In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM. 2015, pp. 1310–1321.

[SS19] C. Song and V. Shmatikov. "Auditing Data Provenance in Text-Generation Models". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 196–206.

[Swa+19] M. Swanberg, I. Globus-Harris, I. Griffith, A. Ritz, A. Groce, and A. Bray. "Improved Differentially Private Analysis of Variance". In: *Proceedings on Privacy Enhancing Technologies* 2019.3 (2019), pp. 310–330.

[SZ15] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *Proceedings of the International Conference on Learning Representations*. San Diego, CA, USA, 2015.

[TD03] E. F. Tjong Kim Sang and F. De Meulder. "Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition". In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 2003, pp. 142–147.

[Tho+20] A. Thomas, D. I. Adelani, A. Davody, A. Mogadala, and D. Klakow. "Investigating the Impact of Pre-trained Word Embeddings on Memorization in Neural Networks". In: *Proceedings of the 23rd International Conference on Text, Speech and Dialogue*. Brno, Czech Republic, Sept. 2020.

[THT04] M. Tang, D. Hakkani-Tür, and G. Tur. "Preserving Privacy in Spoken Language Databases". In: *Proceedings of the International Workshop on Privacy and Security Issues in Data Mining, ECML/PKDD*. 2004.

[Tjo02] E. F. Tjong Kim Sang. "Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition". In: *Proceedings of COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*. 2002.

[Tom+20] N. Tomashenko, B. M. L. Srivastava, X. Wang, E. Vincent, A. Nautsch, J. Yamagishi, N. Evans, J.-F. Bonastre, P.-G. Noé, M. Todisco, et al. *The VoicePrivacy 2020 Challenge Evaluation Plan*. Online. 2020.

[Tru+19] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei. "Demystifying Membership Inference Attacks in Machine Learning as a Service". In: *IEEE Transactions on Services Computing* (2019).

[Wah00] W. Wahlster, ed. *Verbmobil: Foundations of Speech-to-Speech Translation*. Berlin: Springer, 2000.

[WTY19] X. Wang, S. Takaki, and J. Yamagishi. "Neural Source-filter-based Waveform Model for Statistical Parametric Speech Synthesis". In: *Proceedings of ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 5916–5920.

[Yan+17] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao. "A survey on security and privacy issues in Internet-of-Things". In: *IEEE Internet of Things Journal* 4.5 (2017), pp. 1250–1258.

[ZBH19] Y. Zhang, J. Baldridge, and L. He. "PAWS: Paraphrase Adversaries from Word Scrambling". In: *Proceedings of NAACL-HLT 2019*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 1298–1308.

[ZDM19] H. Zhang, Y. N. Dauphin, and T. Ma. "Residual Learning Without Normalization via Better Initialization". In: *Proceedings of the International Conference on Learning Representations*. New Orleans, LA, USA, 2019.

[Zha+17] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. "Understanding deep learning requires rethinking generalization". In: *Proceedings of the International Conference on Learning Representations*. Toulon, France, 2017.