Review of "ADePT: Auto-encoder based Differentially Private Text Transformation"

Arian Tashakkor

December 6, 2024

1 General Overview of ADePT

In this research, an auto-encoder based differentially private algorithm called "ADePT" is introduced for NLP use cases. In short, it aims to transform the private portions of a document by first training an LSTM-based auto-encoder on clean data and then adding a controlled amount of noise sampled from either Laplacian or Gaussian distributions to the latents obtained from the encoder and decoding the result and finally feeding the decoded documents into a downstream model. It is shown that this provably differentially private procedure retains enough information to be utility-preserving with respect to the downstream model.

The efficacy of this method is tested against a Membership Inference Attack and it's experimentally shown to be more effective than the Redactive baseline.

2 Motivation

The authors state that their motivation for this research is to "prove robustness against privacy attacks and offer practical solutions for transforming and releasing datasets without compromising privacy." This is specially important when working with or intending to publish a dataset within which there exist personal information. Differential Privacy is cited by the authors to provide a "strong [and formal] definition of privacy" which can be used in a practical setting to tackle privacy concerns.

According to the authors, there already exist several solutions for transforming datasets containing private information. However, the main concern is that the existing methods are not necessarily utility-preserving for downstream NLP tasks due to the amount of noise added to the text and the sensitivity of text as a modality to noise, compared to other modalities of data such as image. The main purpose of this research is to provide a utility-preserving, differntially private procedure which can be successfully applied to the textual domain.

3 Methodology

ADePT works by first estimating the probability distribution of a given domain of text through an auto-encoder and then altering the latents that are input to the decoder with the hope of altering the private parts of a piece of text without damaging its utility for the downstream task.

Therefore, from a bird's eye view, ADePT consists of the following components:

- **Encoder**: A LSTM-based text encoder that accepts tokenized pieces of text as input and outputs latent vectors of a predetermined dimensionality.
- **Decoder**: The counterpart to the encoder is a LSTM-based decoder which takes as input vectors of latents and outputs sequences of tokens.
- **Privatizer**: A module that acts in between the encoder and the decoder. It is responsible for clipping the latents to reside within a hypersphere of radius C and then adding a pre-defined amount of noise to the latents produced by the encoder before feeding them into the decoder. The privatizer functions differently during training and inference in that during the former, it does not noisify the latents to make sure the auto-encoder is trained properly.

With these components in mind, here is how ADePT is trained and then employed for inference:

- **Pre-processsing**: In the pre-processing step which is mutual between training and inference, the labels are prepended to each input in each record of the dataset being used for evaluation. For instance, in this specific scenario with intent classification being the downstream task, the intent of each record is prepended to the its respective piece of text with a special token to demarcate it from the rest of the text. This step helps with preserving utility of the dataset if the decoder is forced to learn to recreate it during training.
- **Training**: The auto-encoder is given the pre-processed pieces of text as input and is expected to reproduce them verbatim as output. The privatizer in this step simply clips the latents to encourage the encoder to produce more compact and regularly sized embeddings.
- **Inference**: At inference, the trained auto-encoder is given the training portion of the data intended for the downstream model. This time, the privatizer also adds the requisite gaussian or laplacian noise for ensuring differential privacy. The intent labels and texts produced during inference are then used as training data for the downstream model.

ADePT is then tested and compared to the Redactive algorithm as baseline and is shown to be effective against MIA attacks as a DP procedure while being more utility-preserving than the baseline.

4 Critiques

Here I will be delving into what I deem to be the strengths and weaknesses of this paper.

4.1 Strengths

1. The paper is concise and to the point, often to a fault but it gets the main idea across very quickly by doing away with unnecessary explanations.
2. Utility preservation through annotation-aware auto-encoders is a novel idea which merits further investigation although it doesn't seem to have a rigorous mathematical reasoning to back it up.
3. The approach itself is easy to understand and doesn't involve any overly complex components.
4. The procedure is provably differentially private. The mathematical guarantee is very valuable in practice.

4.2 Weaknesses

In favor of brevity, only the most important weak points are discussed here. Grammatical errors, out-of-scope pieces of text, formatting and similar nitpicks are not mentioned.

4.2.1 Regarding §3

1. Eq. (2) is not sufficiently talked about. Specifically the constant C is never defined in the equation and the role of clipping isn't immediately comprehensible.
2. The proof given for differential privacy of the procedure depends too much on the reference text. Although upon revisiting said text the proof is more sensible, I believe at the bare minimum a sketch of the proof should have been given and the reader would then have been referred to the original reference for a complete proof.
3. A block diagram figure of the workflow of ADePT at training and inference should have been presented to facilitate the understanding of §3.

4.2.2 Regarding §4

The weakest section of the paper in my opinion is §**4 Experimental setup**. It suffers from several critical issues:

1. This paper was published in 2021 while BERT came out in 2018. I don't see any reason why LSTMs should be used as feature extractors in the paper even if it's just to demonstrate the efficacy of the procedure.
2. Architectural details are very vague and the loss function is not well-defined and yet no source code for this paper is available. This makes it very difficult to reproduce the results and brings the veracity of the paper into question.
3. The threat model and extent of knowledge of the attacker should have been formally discussed. In §4.5, it is said that "we train the attack model on confidence scores returned by a shadow IC model trained similarly as the target IC model." This implies that attacker has full knowledge of the training data and model architecture but this is never explicitly mentioned which make it subject to misinterpretation.

4.2.3 Regarding §5

1. In the charts presented in Fig. 1, the y-axis would be better defined as "Attack AUC" so as to not cause any confusion.
2. The headings in Tab. 1 are misaligned.
3. The so-called "Redactive" mechanism is mentioned only once in the paper in this section without any reference to the original work.
4. No investigation is made into the non-monotonicity of the Accuracy-AUC curve with respect to variance values and why the gaussian mechanism seems to outperform the laplacian mechanism.

5 Generative AI Disclaimer

GPT-4o was used to help with understanding Eq. 2 of the paper. Here is the prompt used:

```
  Consider this paper:
"ADePT: Auto-Encoder based differentially private text transformation"

In it, the authors claim to have applied clipping to ensure that the latent representations are encouraged to reside
within a hyper-sphere of radius C.

First:  how does the clipping term ensure in equation (2) ensure this?
Second:  why must this encouragement regarding latents be made?

Please look up the paper before answering to make sure your response is factual.
```