SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

# A Linguistics-based Approach for Achieving Sentence-level Differential Privacy

Chaeeun Lee

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

# A Linguistics-based Approach for Achieving Sentence-level Differential Privacy

# Ein auf Linguistik basierender Ansatz zur Erzielung von Differentieller Privatsphäre auf Satzebene

| | |
|---|---|
| Author: | Chaeeun Lee |
| Supervisor: | Prof. Dr. Florian Matthes |
| Advisor: | M.Sc. Stephen Meisenbacher |
| Submission Date: | 2024.04.15 |

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Garching bei München, 2024.04.15 Chaeeun Lee

# AI Assistant Usage Disclosure

## Introduction

Performing work or conducting research at the Chair of Software Engineering for Business Information Systems (sebis) at TUM often entails dynamic and multi-faceted tasks. At sebis, we promote the responsible use of *AI Assistants* in the effective and efficient completion of such work. However, in the spirit of ethical and transparent research, we require all student researchers working with sebis to disclose their usage of such assistants.

For examples of correct and incorrect AI Assistant usage, please refer to the original, unabridged version of this form, located at this link.

## Use of *AI Assistants* for Research Purposes

**I have used AI Assistant(s) for the purposes of my research as part of this thesis.**

Yes ◯ No ◯

**Explanation:**

Grammarly - to improve English writing

I confirm in signing below, that I have reported all usage of AI Assistants for my research, and that the report is truthful and complete.

| | |
|---|---|
| Location, Date | Author |

# Acknowledgments

First and foremost, I would like to express my gratitude to my supervisor, Prof. Dr Florian Matthes for giving me the opportunity to do the research at the chair of Software Engineering for Business Information Systems.

Likewise, I would like to thank Stephen, my advisor, for introducing me to the interesting field of DP / NLP and for his guidance and feedback throughout my work.

Most importantly, I am grateful to my family, especially Ms. Do, for the endless love and emotional support from the other side of the world.

Finally, special thanks to Merlin der Zauberer, for his constant encouragement and magical support.

# Abstract

Differential privacy (DP) is increasingly recognized as a critical framework for preserving privacy in natural language processing (NLP). Despite attempts to apply DP in textual data, most approaches are focused on using the privacy budget at the word level. As such, a research gap exists regarding handling the limited privacy budget within sentences. Particularly, the reasonable distribution of the privacy budget across individual words in a sentence considering preserving privacy is an open question.

The main goal of this thesis is to establish a concept of sentence-level privacy through leveraging a linguistics-based analysis. Through a comprehensive review of research on DP concepts, this study investigates the research gap. Then, we suggest a new approach to achieve sentence-level DP handling the distribution problem with a limited privacy budget. The underlying hypothesis is that words with more information are more likely to be worthy of privacy protection. Therefore, for the distribution mechanism, we integrate linguistic methodologies to quantify word informativeness. We develop a privacy budget distribution framework prototype that distributes given epsilon values to individual tokens in the sentence based on integrated linguistic analysis. The usage of the prototype is presented with examples.

The impact on privacy preservation and utility maintenance of the epsilon distribution applied to different differential privacy mechanisms on NLP will be evaluated on various datasets. The result shows that the data perturbed with the suggested approach have better privacy preservation while maintaining the utility scores in most cases.

By introducing privacy budget distribution based on the analysis at the sentence level, this thesis contributes to advancing the concept of sentence-level differential privacy with a linguistic approach. Moreover, this work provides a practical solution for applying DP in NLP to handle a limited privacy budget. Finally, we suggest directions for future research to improve the here presented approach.

KEYWORDS:
Differential Privacy, Natural Language Processing, Privacy Budget Distribution

# Contents

# 1 Introduction

## 1.1 Motivation

The exponential growth of text-based data in various Natural Language Processing (NLP) applications underscores the importance of safeguarding individuals' privacy and confidentiality. Many NLP domains deal with text data containing highly sensitive information about individuals, e.g. names, addresses, medical conditions or private conversations revealing personal opinions. As data volumes surge, ensuring the protection of personal information contained within textual data becomes increasingly critical.

Differential Privacy (DP) has emerged as a promising framework for addressing data privacy concerns, providing robust guarantees through statistical methods. Its goal is to enable meaningful data analysis while minimizing the impact of individual data inclusion or exclusion on computation or analysis results, thereby safeguarding individuals' privacy.

Despite claims and attempts to apply DP in NLP, its potential applications in this domain remain largely unexplored. Especially, there is a gap in the research regarding the application of DP at the sentence level, particularly concerning the allocation of varying privacy budgets to different parts of the data. This gap highlights a need for studies focusing on distributing privacy budgets at a higher level within textual data than at the word level.

In this work, we propose a new approach: distributing the privacy budget to individual tokens within sentences to apply DP at the sentence level in the context of NLP. The individualized budget is calculated based on the quantified informativeness of each token in the sentence. Then, we evaluate the application of the privacy budget distribution by applying it to actual DP mechanisms.

## 1.2 Research Questions

The following research questions will be addressed to achieve the goals outlined in the motivation.

- RQ1: How can Differential Privacy be effectively applied at the sentence level within Natural Language Processing, considering the intelligent distribution of privacy budgets for individual words within a sentence?
- RQ2: How can the theoretical concepts of sentence-level privacy with informativeness analysis be translated into an implementable framework?
- RQ3: How well does the suggested differential privacy framework protect private data while preserving the utility of the text data?

At the heart of this investigation lies the concept of word informativeness, an essential hypothesis underpinning this study. We assume that words with higher informativeness are more likely to contain sensitive or private information that might connote identity and thus require protection. While the specific words or sections of text necessitating privacy protection may vary based on task objectives, data sources, or individual interpretations of privacy, this research assumes that words with greater informational content have a higher chance of containing private information warranting safeguarding.

## 1.3 Thesis Overview

This thesis is structured as follows:

Introduction: This chapter sets the stage for the thesis by presenting the motivation behind the research, highlighting the significance of applying DP in NLP. It also introduces the research questions addressed in the study and provides a brief overview of the thesis structure.

Theoretical Background: the foundational concepts of DP and its relevance to NLP are explored in detail in this chapter. It lays the groundwork by explaining the fundamental principles of DP. It introduces NLP notions and techniques relevant to this work and the methodologies to quantify word informativeness or significance in NLP.

Literature Review: This chapter examines prior research and studies in the field, identifying gaps, challenges, and opportunities for further investigation.

Methodology: Divided into several sections, this chapter details the methods employed in this thesis. It includes a thorough review of the structure, the implementation of a prototype for the distribution of privacy budget to each word in a sentence, and the evaluation methodology and process adopted to assess the effectiveness of the proposed approach.

Results: Here, the results obtained from prototyping and subsequent prototype evaluation are presented and analyzed. The chapter provides insights into the effectiveness of the suggested framework in preserving privacy while maintaining the utility of the text data.

Discussion: This chapter discusses the key findings and the implications derived from the results. It also addresses any challenges encountered during the study and offers recommendations for future research directions.

Conclusion: The final chapter summarizes the essential points and contributions of the study.

# 2 Theoretical Background

## 2.1 Epsilon Differential Privacy

Differential Privacy, proposed by Dwork in 2006 [1], stands out as a robust framework in the realm of privacy-enhancing technologies. It ensures a high level of privacy while preserving the accuracy of statistical information, thus safeguarding individual data from being discerned. Here, privacy means preventing individual records from being identified. Essentially, it addresses the vital need to protect sensitive information from unauthorized access or disclosure, while still permitting meaningful data analysis.

The basic idea is to introduce noise to data sets, therefore making it harder to identify specific individuals while not interrupting statistical analysis, ensuring that the results are still statistically meaningful. Several techniques, such as the Laplace Mechanism or the Gaussian Mechanism, are used to determine the amount of noise to be added [2].

It sets a privacy budget parameter, epsilon, denoted by $\epsilon$, to limit the probabilistic risks associated with data disclosures and determine the scale of the noise to be added [3]. The privacy budget represents the maximum amount of privacy that can be compromised during the data analysis, where lower epsilon values indicate stronger privacy guarantees. This means the noise magnitude increases with decreasing $\epsilon$, providing stronger privacy guarantees at the cost of increased noise.

Overall, epsilon differential privacy's primary goal is finding the right balance between preserving privacy and maintaining data utility [4]. In the context of DP, maintaining utility involves ensuring that the analysis or processing of data remains accurate and meaningful while preserving privacy. Maximizing utility consists of ensuring that the insights or predictions derived from the data remain as precise and informative as possible despite the privacy-preserving mechanisms applied. However, the noise level added to ensure privacy can potentially degrade the utility of the data for specific tasks.

This observation, however, is general: an inherent trade-off between utility and privacy is a key characteristic of every DP approach [5], therefore, it cannot lead to a one-size-fits-all solution. Increasing privacy protections, such as by adding more noise in DP mechanisms, can often lead to a decrease in data utility, as the noise introduced may distort the original data and compromise the accuracy of analyses or models. Conversely, maximizing utility without adequate privacy protections can result in privacy breaches and the unauthorized disclosure of sensitive information.

## 2.2 Differential Privacy in Natural Language Processing

Differential Privacy is being considered as a solution to address privacy issues in NLP tasks since NLP methods often involve processing large amounts of data, which may include private or personally identifiable information. Especially with emerging interest and research with large language models and other NLP technologies, NLP applications often involve sensitive data, so adapting differential privacy to NLP methods is an active area of study aimed at developing effective techniques for preserving privacy in text-based tasks.

However, as the concept of Differential Privacy was initially designed for structured data sets, it is challenging to apply DP in NLP. The NLP tasks often involve processing large amounts of unstructured text data, making applying DP without compromising the utility challenging [6].

## 2.3 Basics of Natural Language Processing

This section explains the basic notion of NLP used in this research.

### 2.3.1 Preprocessing

Preprocessing in NLP refers to the steps and techniques applied to raw text data before it is used for analysis or modelling tasks. It aims to clean, normalize, and transform the raw text into a format suitable for downstream NLP tasks [7]. The preprocessing typically involves several common steps:

- Tokenization: Breaking the text into smaller linguistic units called tokens. These tokens can be words, sub-words, or characters, depending on the granularity required for the task.
- Lower-casing: Converting all text to lowercase ensures consistency and reduces vocabulary size. This step helps in treating words with different cases (e.g., "Hello" and "hello") as the same token.
- Removing punctuation: Eliminating punctuation marks such as periods, commas, and quotation marks, which do not typically carry meaningful information for many NLP tasks.
- Stemming and lemmatization: Reducing inflected words to their base or root form. Stemming involves removing suffixes to obtain the word stem, while lemmatization maps words to their dictionary form (lemma). This step helps reduce word variations and improve text normalization.

### 2.3.2 Embedding

An embedding is a dense vector representation of words or phrases in a continuous vector space. These embeddings are learned representations that capture semantic and syntactic information about words based on their context in a text corpus [8]. Word embeddings are typically learned as part of the training process of neural network-based models, such as word2vec and GloVe (Global Vectors for Word Representation).

Similarly to word embedding, sentence embedding refers to a fixed-size numerical representation of a text sequence or sentence. The primary goal of sentence embeddings is to capture the semantic meaning and context of the input text in a continuous vector space [9]. These embeddings are typically learned through deep learning models, particularly transformer-based architectures such as BERT [10], GPT [11], and their variants.

Sentence embeddings are helpful in various NLP tasks, including text classification, semantic similarity calculation, sentiment analysis, machine translation, and information retrieval. By encoding the semantic information of a sentence into a fixed-length vector, sentence embeddings enable downstream models to effectively process and understand textual data [9].

### 2.3.3 Transformer

Transformer is a deep learning model architecture primarily used in natural language processing tasks. Introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017 [12], it revolutionized NLP with its self-attention mechanism which is the key concept behind the transformer architecture.

The self-attention mechanism enables the model to weigh the importance of different words and capture relationships between words in a sequence [13]. These attention scores are then used to compute weighted sums of the input embeddings, generating context-aware representations for each word.

Transformer models showed high performance on different NLP tasks. They can be used to pretrain models on large-scale corpora using self-supervised learning objectives such as masked language modeling or next-sentence prediction [13]. Also, the pretrained model can be fine-tuned to task-specific datasets to adapt them to specific NLP tasks, which is also used for the evaluation of the prototype.

## 2.4 Linguistic Methodologies Related to Quantifying Informativeness

The following is a theoretical explanation of computer-linguistic methodologies used to quantify informativeness in this research. The usage of methodologies is based on the hypothesis introduced at the end of the section 1.2, which suggests that words with more informativeness could potentially include more private or sensitive information and, thus, are more important in terms of privacy protection.

### 2.4.1 Information Content

In the context of NLP, information content (IC) refers to the amount of knowledge or meaning contained within a word or a piece of text [7]. It measures how specific or rare the meaning of a word is within a given context or dataset. The key concept behind information content is based on the idea that words with more specific or uncommon meanings carry more information and are, therefore, more informative [14].

WordNet, a lexical database of English, is mainly used to quantify information content. WordNet organizes words into sets of synonyms called synsets and represents relationships between them, such as hypernyms (more general concepts) and hyponyms (more specific concepts). The information content is often calculated using WordNet by measuring the specificity of a word's meaning based on its position in the WordNet hierarchy [15]. The IC value is typically computed based on the frequency of occurrence of a word's synsets in a large corpus of text. Words with lower IC values are considered more informative as they represent more specific or rare concepts.

### 2.4.2 Part of Speach and POS Tag

Part of speech (POS) refers to the grammatical category of a word in a sentence. It indicates a word's role in a sentence's structure. POS tagging is the process of automatically assigning the appropriate part of speech tag to each word in a text corpus [7].

The key concept of POS tagging is to categorize words into their respective grammatical categories, such as nouns, verbs, adjectives, adverbs, pronouns, prepositions, conjunctions, and determiners. This categorization helps understand the syntactic structure of sentences and aids in various NLP tasks like parsing, information extraction, and machine translation [16].

Some commonly used POS tags include [17]:

- `Noun (NN)`
- `Verb (VB)`
- `Adjective (JJ)`
- `Adverb (RB)`
- `Pronoun (PRP)`
- `Preposition (IN)`
- `Conjunction (CC)`
- `Determiner (DT)`
- `Interjection (UH)`

where he letters in parentheses are the tags for the corresponding POS.

The Natural Language Toolkit (NLTK) library [7] is commonly used for POS tagging in the prototype of this thesis. NLTK is a widely used Python library for NLP tasks, including tokenization, POS tagging, parsing, and more. One of the most commonly used POS taggers in NLTK is the PerceptronTagger[1], which utilizes a statistical approach. The PerceptronTagger is based on the perceptron algorithm, a type of linear classifier used in machine learning. The perceptron algorithm learns a set of weights for each feature in the input data to make predictions about the output class (POS tag).

### 2.4.3 Named Entity Recognition

Named Entity Recognition (NER) is a NLP technique used to identify and classify named entities within a text into predefined categories such as person names, organization names, locations, dates, and numerical expressions. The primary goal of NER is to extract and label specific entities from unstructured text data, enabling automated information extraction and analysis [7, 16].

NER typically employs machine learning algorithms, particularly sequence labeling models such as Conditional Random Fields (CRF) [18] or Bidirectional Encoder Representations from Transformers (BERT) [10] trained on annotated datasets. These models analyze the linguistic features of words in a sentence, such as their context, part-of-speech tags, and neighboring words, to predict the presence and type of named entities.

The entities commonly recognized by NER systems include [19]:

- Person Names: Individuals' names, including first names, last names, and titles.
- Organization Names: Names of companies, institutions, agencies, etc.
- Location Names: Place names, such as cities, countries, streets, landmarks, etc.
- Date and Time Expressions: Temporal references like dates, times, duration, etc.
- Numerical Quantities: Numeric expressions such as percentages, monetary values, measurements, etc.

For NER, the open-source library spaCy[2] provides pretrained models that have been trained on a large corpus and can be used [20]. Using these models offers efficient and fast performance, and it provides a simple and user-friendly interface for incorporating NER functionality into NLP pipelines or applications, making it suitable for real-time or large-scale processing tasks.

The key technology behind spaCy's NER capability is its use of machine learning models, specifically neural networks. The models are trained using labeled datasets containing examples of text with annotated named entities, allowing them to learn to generalize and identify similar entities in the new text.

[1]`https://www.nltk.org/_modules/nltk/tag/perceptron.html`
[2]`https://spacy.io/api/doc`

### 2.4.4 Sentence Transformer and Similarity Calculation

Sentence Transformers are a type of deep learning model designed explicitly for encoding and processing text at the sentence level. Unlike traditional word-level embeddings, which represent individual words or tokens, sentence transformers generate fixed-length vector representations for entire sentences [21]. These representations capture the semantic meaning and context of the sentences, enabling a wide range of NLP tasks such as semantic similarity calculation, text classification, and information retrieval.

The key idea behind sentence transformers is to leverage pre-training techniques such as self-supervised learning on large-scale text corpora to learn rich, context-aware sentence embeddings [22]. Using this embedding, we can calculate the similarities between different sentences. There are several methods to calculate embedding similarity, depending on the nature of the embeddings and the specific task. Cosine Similarity is one of the most widely used similarity measures for embeddings [23]. It measures the cosine of the angle between two vectors and ranges from -1 (perfectly dissimilar) to 1 (perfectly similar). Cosine similarity is calculated as the dot product of the two vectors divided by the product of their magnitudes, *i.e.*

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}, \tag{2.1}$$

where **a** and **b** are $n$-dimensional vectors.

# 3 Literature Review

This chapter provides an overview of existing research on differential privacy in NLP and highlights the gaps and limitations in current approaches. Additionally, it identifies the need for a new approach to achieving sentence-level DP by intelligent privacy budget distribution.

## 3.1 Related Studies on Application of DP in NLP

A number of research studies and implementations have used differential privacy in the context of NLP tasks. However, the studies so far are mainly focused on word-level embedding and there is a lack of research regarding the detailed allocation of the privacy budget in textual data.

A comprehensive review by [24] examines recent advances in DP deep learning models in NLP tasks, focusing on two main approaches for achieving privacy in DP-NLP: gradient perturbation-based methods and embedding vector perturbation-based methods. Additionally, [25] places emphasis on preserving privacy in analyses of textual data, particularly within large-scale analyses, addressing concerns surrounding customer-supplied data. As another example, [26] proposes a new notion of privacy called selective differential privacy, dealing with the sparseness of private information in natural language data to provide strict privacy guarantees on the sensitive part of the data to increase the utility of the model.

However, critiques of word-level DP approaches have been raised. In [27], they question word-level DP mechanisms used for text anonymization, advocating for strategies that offer more flexibility in text generation. They propose an anonymization approach based on paraphrasing, outperforming traditional word-level mechanisms in experiments. Moreover, they highlight the linear growth of the privacy budget in DP mechanisms, indicating that the budget required for privatizing an entire sequence may increase linearly with its length.

Efforts to achieve DP at the sentence level have also emerged, such as SentDP, as introduced in [28]. This technique ensures strong privacy guarantees at the sentence level for document embeddings. However, while it focuses on maintaining indistinguishability in embeddings, the paper does not explicitly address epsilon or privacy budgets.

Despite the progress made in word-level DP mechanisms and the emergence of approaches targeting sentence-level DP, a gap remains in research concerning the application of individual privacy budgets at the sentence level.

## 3.2 Studies on Privacy Budget Distribution

In the realm of statistics with structured data, researchers have explored various aspects of privacy budget allocation and the application of differential privacy [29, 30]. However, the domain of NLP has received comparatively less attention, particularly concerning the granularity of privacy budget allocation at the sentence level. Despite existing efforts to explore privacy budget utilization, there remains a notable gap in intelligently distributing these budgets within textual data to achieve sentence-level differential privacy.

One notable contribution regarding privacy budget distribution is the work by Rosenblatt et al. [29], which introduces ensemble methods aimed at distributing the privacy budget wisely to maximize predictive accuracy in models trained on differentially private data. This approach considers feature importance in informing the distribution of the privacy budget, enhancing model performance while ensuring privacy preservation.

Similarly, Bakria et al. [30] investigate the optimal distribution of privacy budgets in differential privacy, particularly in adaptive multi-data consumer scenarios. Their study tackles the challenge of allocating a given privacy budget among various data consumers, proposing a method to distribute the budget optimally based on consumer characteristics and data utilization contexts.

In particular, until now, methods of handling personal information protection budgets in natural language processing have been naively applied to units (words or tokens). Usually, a budget for each word is set and the privacy budget for the entire is calculated by multiplying by the number of words in the text as the budget. Or given a budget for the whole of the sentence, the budget is divided equally among each word and applied, which is the method that has been used so far.

This approach does not treat the personal information protection budget as a budget and does not reflect the actual situation where the budget is limited. Therefore, distribution and allocation as a budget have not yet been explored, and further research is needed to effectively distribute and apply the privacy budget within sentence-level text data to implement sentence-level DP.

## 3.3 Linguistics Approach for Measuring Informativeness

To explore the academic background of quantifying word importance in terms of privacy within the sentence, we also explored related studies on relevant linguistics approaches. While there isn't a direct method for pinpointing informativeness in a text, there are approaches within the realm of term extraction that are commonly utilized.

For example, [31] proposes an approach for measuring word significance in a corpus by utilizing vector length and term frequency. It builds upon the concept of distributed representations of words as real-valued vectors in a low-dimensional space, commonly employed for extracting syntactic and semantic features from large text corpora. The research showed the incorporated approach can reliably measure word significance within a corpus.

Another study [32] introduces a method for computing term specificity, a crucial aspect in understanding word significance, particularly in the context of information retrieval and summarization. Leveraging Latent Semantic Analysis (LSA), the proposed method models the learning rate of word meaning, offering insights into effective measuring term specificity.

Furthermore, [33] suggests a method for measuring term informativeness in context. By utilizing web knowledge to encode term informativeness instead of existing methods that rely on statistical information from corpora. Assessing the semantic similarity between a term and its most featured context in a knowledge base such as Wikipedia effectively captures the significance of terms within their contextual environments.

Using attention [12] could also be considered as one way to extract word informativeness. Attention mechanisms are primarily used to identify the relevance or importance of different parts of a sentence or sequence during sequential data processing, such as in recurrent neural networks (RNNs) or transformer models. While attention mechanisms can help capture the contextual significance of words within a sentence, they are not typically used to measure the informativeness or importance of individual words.

Additionally, recent research [34] questions about the faithfulness of importance measures in language models using attention, demonstrating that the faithfulness of importance measures is found to be both model-dependent and task-dependent, contradicting previous evaluations in the field.

Beyond these approaches, various attempts have been made to measure or quantify term significance by leveraging linguistic analysis. For instance [35] introduces an approach for measuring word significance by incorporating POS relevance weights to enhance word embeddings. The research utilizes position-dependent POS relevance weighting matrices within context windows.

Similarly, [36] proposes a new type of term weight from part-of-speech n-gram statistics, which represents how informative a term is in general, based on the POS contexts in which it generally occurs in language.

These approaches offer alternative perspectives on measuring term significance based on linguistic features.

# 4 Methodology

This section presents the overall structure and detailed explanation of each component of our privacy distribution framework prototype. Also, we introduce the necessary settings and methodologies to measure and evaluate the influence of the suggested approach on the data.

## 4.1 Prototype Architecture and Implementation

As mentioned at the end of section 1.2, the base idea of building the distributor is the assumption that words with more informativeness could potentially include more detectable information regarding privacy, thus requiring a lower privacy budget allocation to ensure stronger privacy protection. Here, it is explained how the prototype is designed and structured and how the epsilon distribution is computed with input data.

### 4.1.1 Methods for Scoring Informativeness

First, we present methods that can practically quantify the importance of individual words in a sentence from a natural language processing perspective. These methods are able to properly reflect statistical and semantic aspects. Accordingly, five numerical methods are applied.

The following explains the importance of those methods and their benefits:

- Information Content (IC): IC helps identify words that are less common or more specific within a domain. Words with lower IC scores are considered more informative because they carry more meaning due to their relative rarity. By incorporating IC, we can prioritize words that are more likely to convey the core concept of the sentence.
- POS tag weights: POS tags indicate the grammatical function of a word (noun, verb, adjective, etc.). Assigning different weights to different POS tags reflects their typical importance in conveying meaning. For example, verbs often play a more central role in a sentence compared to prepositions. This helps distinguish between function words (like articles and prepositions) and content words (nouns, verbs, adjectives) that contribute more to the core meaning [37].
- Named Entity Recognition (NER): NER identifies and classifies named entities like people, locations, and organizations. These entities are often crucial aspects of a sentence's meaning.

- Similarity difference between the sentence and the single word: This method compares the semantic similarity between the entire sentence and each individual word. Words with a larger difference in similarity likely contribute more to the overall meaning, as they introduce new concepts not already conveyed by the rest of the sentence.

- Similarity difference between the sentence and the sentence without each word: This method compares the semantic similarity of the original sentence to versions where each word is removed. Words whose removal causes a more significant drop in similarity are considered more important because their absence significantly impacts the overall meaning. This approach helps identify words that are crucial for conveying the core meaning of the sentence, as their removal significantly alters the semantic representation.

### 4.1.2 Prototyping

The framework, therefore, is a combination of five methods mentioned in section 4.1.1 that can evaluate and quantify the importance of words in a sentence.

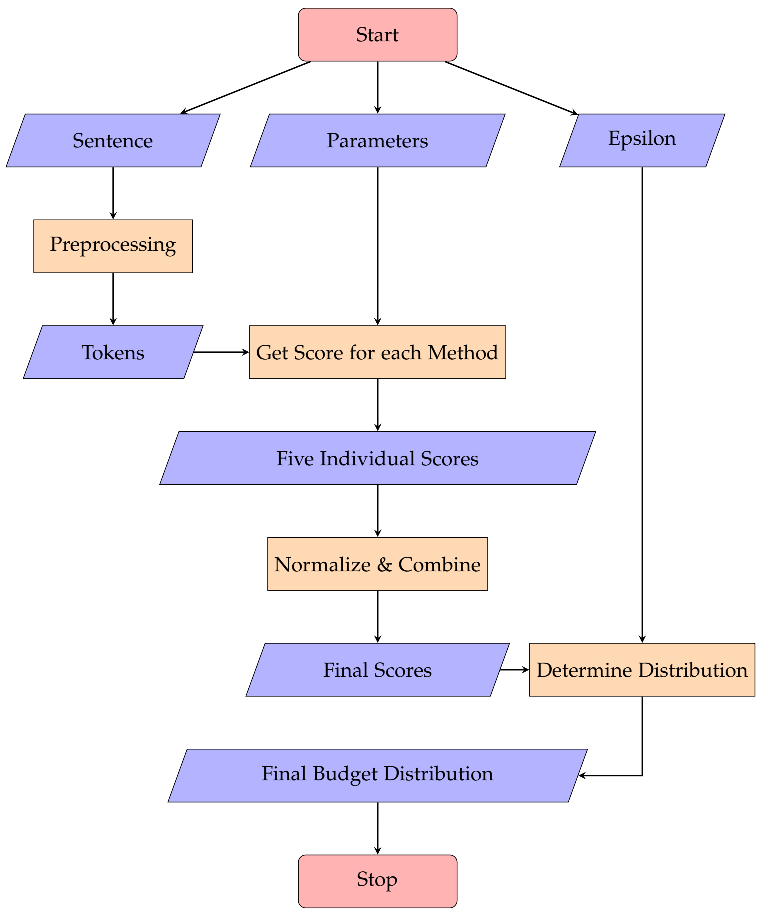


Figure 4.1: Flowchart of the Epsilon Distributor prototype

Figure 4.1 shows the overall flow of the calculation of the epsilon distribution pipeline of the prototype. When it calculates, it takes as arguments a text, total epsilon value and the method parameters. The latter decides whether to include a specific scoring method in the calculation.

First, the input sentence is preprocessed, namely tokenization and punctuation removal applied. Then, each scoring method scores the processed tokens. The individual scores are normalized and then combined together as final scores. The final scores are then used to determine the distribution based on the total epsilon value, and the final distribution of the epsilon for each token in the sentence is calculated.

The prototype allows for customization of parameters such as deciding which scoring methods to use, setting different POS weights, and giving the total epsilon of choice. Additionally, other methods to evaluate informativeness can be easily added to the prototype. This gives the user the flexibility to adapt the framework to their goals and circumstances.

When the epsilon distributor is initialized, it sets up the necessary tools for scoring, such as POS weights or lemmatizer. The `get_distribution()` function is provided to compute the final distributed epsilon based on the chosen scoring methods and parameters. It takes the target sentence and the total epsilon value as parameters and additional parameters to disable each scoring method.

### 4.1.3 Precondition and Initialization

To run the prototype, the following libraries should be available in the environment:

- `NLTK` (Natural Language Toolkit) for tokenization, part-of-speech tagging, and WordNet access.
- `spaCy` for Named Entity Recognition (NER) and linguistic processing.
- `SentenceTransformer` for encoding sentences into dense vectors.
- `NumPy` for numerical computations.
- `string` for text processing functions.

The prototype initializes with the necessary components, including the SentenceTransformer [21] model for encoding sentences and the spaCy library for scoring methods such as NER. It also defines a set of default parameters and weights for different aspects of word informativeness in case the user wants to compute the scoring and distribution in various settings. This ensures the prototype is ready for use out-of-the-box and can be easily customized to suit specific requirements or experimental setups.

### 4.1.4 Scoring Methods

Concretely, the following scoring methods are implemented:

- **Information Content (IC):** `_get_ic()`
  Utilizes WordNet [15] to calculate words' Information Content (IC) based on their semantic relevance. WordNet organizes words into synsets, which are groups of synonymous words that share an ordinary meaning. The hierarchical structure of WordNet synsets allows for calculating IC values, which indicate the specificity or rarity of a word's meaning within the database. Words with lower IC values are considered more informative as they are less common or specialized.

  In this function, the tokens are lemmatized and converted from a word to its lemma. Then, the function retrieves possible synsets (sets of synonyms) for each lemma. Here, it uses pre-computed IC values for different sources like SemCor, Brown corpus, etc, which were stored when the distributor was initialized. The final IC value for the word is computed by averaging the IC values of all its retrieved synsets. If no synsets are found, the IC is set to 1.0 (presumably a neutral value).

- **Weight based on POS Tag:** `_get_pos_informativeness()`
  Tags POS tags to preprocessed tokens from the original text using NLTK POS tagger for the English language[1] and assigns weights to words based on their Part-Of-Speech (POS) tags, considering the inherent informativeness of different POS categories. The default weights used in this prototype are based on POS tags statistics [38, 39] on tweet data, considering function words and content words [37]. The default weights are assigned as follows: `{'NN': 15, 'PR':14, 'VB':8, 'CD':6, 'JJ':3.7, 'RB':3.4 }` . When a POS tag of a token is not in the list, 0.1 is assigned as the base weight.

- **Named Entity Recognition (NER):** `_get_ner_weights()`
  Identifies the named entities in the sentence utilizing the spaCy library's NER functionality[2] and assigns higher weights to words corresponding to recognized entities. The method starts by loading the spaCy English language model using *en_core_web_sm* pipeline[3] and checks if each word in the sentence is classified as a named entity. Named entities labels are like the following: {CARDINAL, DATE, EVENT, FAC, GPE, LANGUAGE, LAW, LOC, MONEY, NORP, ORDINAL, ORG, PERCENT, PERSON, PRODUCT, QUANTITY, TIME, WORK_OF_ART}[4]. By doubling the weights assigned to these entities, this method emphasizes the protection of such entities in the privacy budget distribution process.

- **Similarity Difference:**
  Measures the difference in semantic similarity between the original sentence and a modified version with the word removed, both at the word and sentence levels. To get sentence and word embedding, *all-MiniLM-L6-v2*[5] from SentenceTransformer library is used.

[1]https://www.nltk.org/api/nltk.tag.pos_tag.html
[2]https://spacy.io/usage/linguistic-features#entity-types
[3]https://spacy.io/models/en
[4]https://spacy.io/models/en#en_core_web_sm
[5]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

- **Word Similarity (Sentence vs. Word):** `_get_similarity_single_word_score()`
  Computes the cosine similarity between the sentence embeddings of the original sentence and the embedding of the individual word in the sentence. The importance measure of the word is determined by taking the absolute value of the similarity score. Words with higher similarity scores are assumed to be more crucial for preserving the overall semantic content of the sentence.
- **Sentence Similarity (Sentence vs. Sentence):** `_get_similarity_except_word_score()`
  Calculate the cosine similarity between the embedding of the original sentence and a modified sentence without each word. The score is calculated as the absolute difference with 1 (1-cosine similarity) since the higher similarity indicates a lower word significance.

### 4.1.5 Combining and Deriving Final Scores

The prototype integrates the scores obtained from each scoring method to compute a comprehensive combined score for each word, reflecting its overall informativeness within the sentence context. The following steps outline the process of combining scores:

Once individual scores for each word are calculated using the implemented scoring methods, such as Information Content (IC), Weight based on POS Tag, Named Entity Recognition (NER), and Similarity Difference at both word and sentence levels, they are normalized to ensure comparability across different scoring methods. Normalization adjusts the range of scores to a common scale, facilitating their combination and interpretation. This step prevents biases introduced by variations in the magnitude or distribution of scores across different methods.

The normalized scores obtained from each method are combined to derive a unified measure of word informativeness. This combination process involves aggregating the individual scores using a predefined algorithm or weighting scheme, which may assign different weights to scores from each method based on their relative importance or reliability.

After combining the individual scores, the prototype computes a final score for each word, representing its overall level of informativeness within the sentence. The final score reflects the collective impact of various linguistic factors, including semantic relevance, grammatical role, named entity status, and contextual significance.

### 4.1.6 Epsilon Distribution

Once the combined scores for each word are computed, the prototype allocates the privacy budget (epsilon) based on these scores. This allocation process ensures that words deemed more informative receive a larger share of the privacy budget, thereby enhancing the protection of sensitive information associated with those words. At the same time, it strives to minimize the impact on data utility by distributing the total epsilon in a manner that preserves the overall coherence of the text.

Words with higher final scores, indicative of greater informativeness, are prioritized for the allocation of the privacy budget. These words are considered to carry more sensitive

```
distributor = epsilon_distributor()
example_sent = "After graduating from Stanford University,
                John Smith moved to Munich to start his new job at SAP,
                where he works as a software engineer"
distributor.get_distribution(example_sent, total_epsilon=30)
distributor.print_scores()
```

Figure 4.2: An example code for executing the epsilon distribution prototpye.

information and warrant a larger share of the privacy budget to safeguard privacy effectively.

The prototype calculates the proportion of epsilon allocated to each word based on its final score relative to the total score of all words in the sentence. This ensures that more informative words receive a higher share of the privacy budget.

The user can also check the final result with the `print_score()` function, which prints the input sentence and the user parameter settings as well as the final information scores with the sum of 1 and the final epsilon distribution from the given total epsilon. Executing the sample code presented in fig. 4.2 leads to the output shown in fig. 4.3.

## 4.2 Experiment and Evaluation

This research aims to develop a method for intelligently distributing the privacy budget within a sentence to achieve sentence-level differential privacy. The experiments aimed to validate this approach by assessing the impact on privacy and utility metrics across various datasets and differential privacy mechanisms.

This section outlines the methodology for conducting the experiments and evaluating the proposed differential privacy framework. To assess the privacy and the utility, we first apply naive transformations to the text data in each data set and then perturb it using the proposed epsilon distributor mechanism. Subsequently, we fine-tune pre-trained language models on the naively transformed and perturbed data sets with epsilon distribution. By evaluating the performance of the fine-tuned model on perturbed data, we aim to compare the performance scores obtained under different conditions. This comparison allows us to gauge the impact of the suggested privacy budget distribution method on the utility of the perturbed data and assess the effectiveness of the proposed approach in preserving privacy.

### 4.2.1 Dataset

This section describes the datasets utilized in our experiments, detailing their characteristics, composition, and evaluation metrics. Table 4.1 shows overall information about each dataset used for the experiment.

For privacy measurement, the following two data sets are used:

- **Trustpilot Reviews Gender Dataset (US English)** [40]: This dataset comprises a collection of user reviews from Trustpilot, a consumer review website, focusing on gender-

```
All happy families are alike; each unhappy family is unhappy in its own way.
Parameters :  Total Epsilon: 1400 |  Ner Weight: True |  Use IC: True
|  Use POS: True |   Use Similarity Sentence: True | Use Similarity Word: True |
Token       Final Score    Final Epsilon Proportion    Final Epsilon Distribution
------------------------------------------------------------------------------
All            0.03625           0.09412                     131.77274
happy          0.06462           0.05281                     73.93140
families       0.16760           0.02036                     28.50348
are            0.07710           0.04426                     61.95993
alike          0.07130           0.04786                     67.00167
each           0.03003           0.11363                     159.08596
unhappy        0.05253           0.06496                     90.94904
family         0.14475           0.02357                     33.00210
is             0.05816           0.05867                     82.13458
unhappy        0.05273           0.06471                     90.60006
in             0.01454           0.23462                     328.47007
its            0.05544           0.06155                     86.17292
own            0.03619           0.09428                     131.98729
way            0.13876           0.02459                     34.42876
------------------------------------------------------------------------------
Total          1.00000           1.00000                     1400.00000
```

Figure 4.3: An example output of using the `print_scores()` function on the example sentence = "All happy families are alike; each unhappy family is unhappy in its own way." and total_epsilon = 1400. The output shows the input parameters at the top, the final score, the epsilon proportion, and the finally distributed epsilon for each word in the sentence at the bottom.

related topics. The dataset contains the gender of the author as the label. The original dataset contains 366,210 reviews. Due to computational constraints, a random subset of 36,621 reviews was selected for the privacy measurement experimentation. Each review includes text content along with gender information.

- **Yelp Dataset** [41]: The Yelp dataset consists of user reviews and ratings of various businesses and services on the Yelp platform. It encompasses diverse topics such as restaurants, hotels, and retail establishments. The dataset contains the 10 author ids as the label.

For utility measurement, we incorporated the General Language Understanding Evaluation (GLUE) [42] benchmark, which provides a diverse set of tasks for evaluating language understanding capabilities across various domains. GLUE aggregates multiple individual datasets, each targeting specific NLU tasks such as sentiment analysis, textual entailment, and more.

We utilized the following tasks from the GLUE benchmark:

| Type | Dataset | Size | Metric | Avg. word count | Total $\epsilon$ (1-Diffractor) | Total $\epsilon$ (DP-MLM) |
|---|---|---|---|---|---|---|
| Privacy | Trustpilot | 36621 | Accuracy | 45 | 45 | 4500 |
| | Yelp | 17336 | Accuracy | 182 | 182 | 18200 |
| Utility | CoLA | 8551/1043 | Accuracy | 8 | 8 | 800 |
| | SST-2 | 30000/872 | Accuracy | 9 | 9 | 900 |
| | MRPC | 3668/408 | Accuracy & F1 Score | 22 | 22 | 2200 |
| | RTE | 2490/277 | Accuracy | 43 | 43 | 4300 |
| | STSB | 5749/1500 | Pearson-Spearman correlation | 10 | 10 | 1000 |

Table 4.1: Dataset information table with its size, evaluation metric, average word length of the texts, the total epsilon used for the perturbation per mechanism. The datasets used for utility evaluation have training and evaluation data separately, as indicated by the fractions in the "size" column. The "Total $\epsilon$" is based on the standard $\epsilon$ value mentioned in section 4.2.2.

- **Corpus of Linguistic Acceptability (CoLA)**: A dataset for evaluating grammatical acceptability in natural language processing tasks. It contains sentences labeled as grammatically correct or incorrect.
- **Stanford Sentiment Treebank (SST-2)**: A sentiment analysis dataset of movie reviews labeled as positive or negative. Due to its large size (originally $60,000$ examples), a random subset of $30,000$ examples was used as the training data for the utility measurement.
- **Microsoft Research Paraphrase Corpus (MRPC)**: A paraphrase identification dataset containing pairs of sentences labeled as paraphrases or not.
- **Recognizing Textual Entailment (RTE)**: A dataset for natural language inference, where each example consists of a pair of sentences labeled as either entailing or not entailing.
- **Semantic Textual Similarity Benchmark (STSB)**: A benchmark dataset for measuring semantic similarity between pairs of sentences, with similarity scores ranging from 0 to 5.

### 4.2.2 DP Mechanism

Two different text DP mechanisms are used for the data perturbation with- and without applying the distributed privacy budget.

- **1-Diffractor:** 1-Diffractor is a DP mechanism for text obfuscation based on word-level Metric Local Differential Privacy (MLDP) mechanisms. The standard privacy of this mechanism was set to 1.

- **DP-MLM:** DP-MLM is a DP text rewriting mechanism that leverages masked token prediction in BERT-based models. The standard privacy of this mechanism was set to 100.

They are both based on the word-level approach in terms of applying the epsilon to the tokens, but can also take the predefined distribution of the epsilon values for each part of the input text.

### 4.2.3 Training and Evaluation Data Construction

The overall flow of the training data construction is visualized in fig. 4.4.

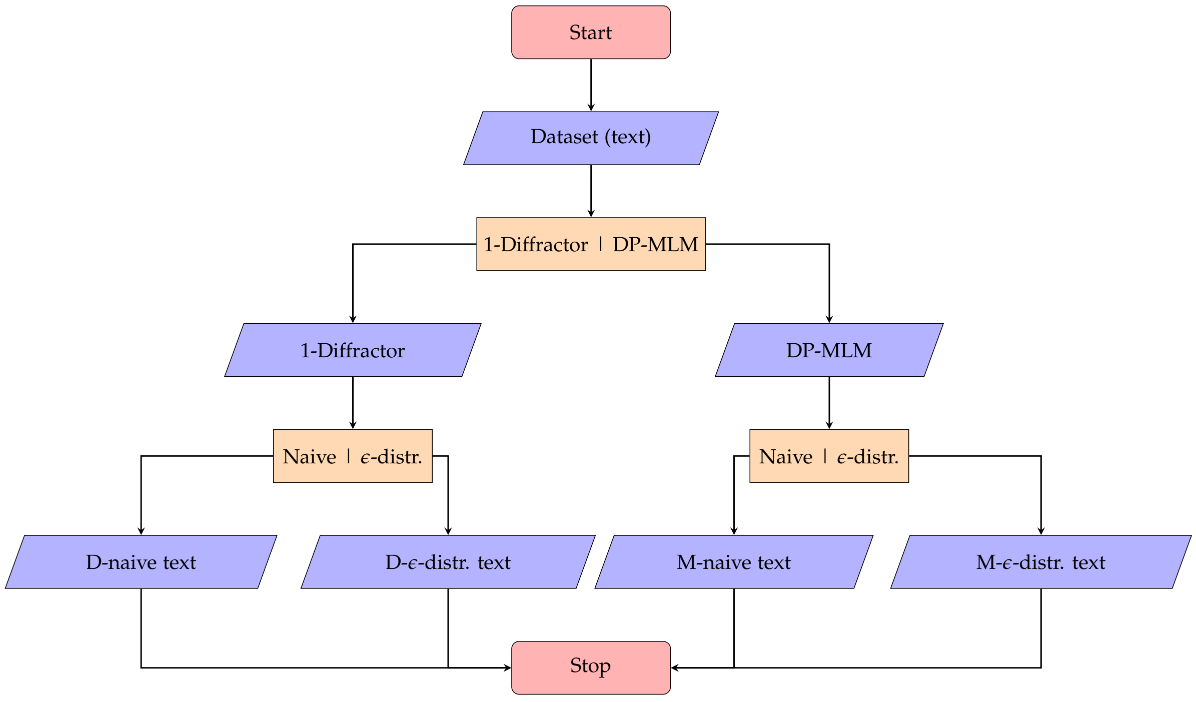
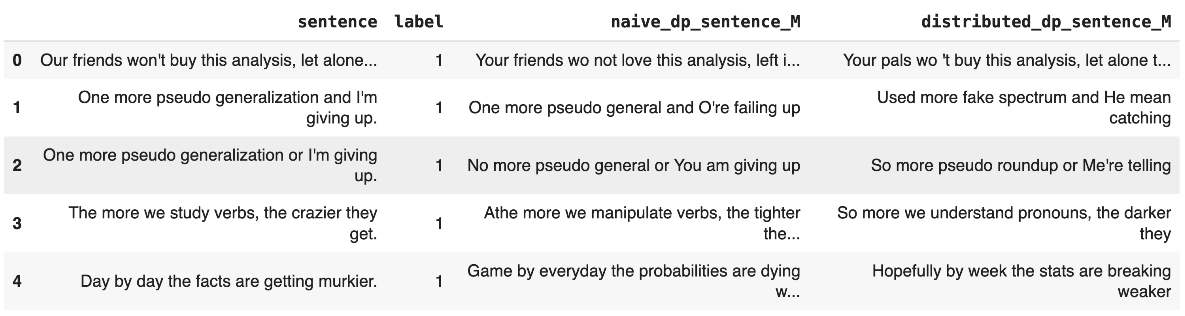


Figure 4.4: Flowchart visualizing how the Training- and Evaluation datasets are created. Each dataset went through perturbation of both mechanism, once with naively divided privacy and once with distributed budget using epsilon distributor. As the result of this process, four variations of the original texts are generated.

To construct the training data, two different differential privacy mechanisms, 1-Diffractor and DP-MLM, were applied to the previously listed datasets with predefined epsilon values. The choice of epsilon values was determined based on the average text length in each dataset. It's important to note that in this approach, the same epsilon value was applied to each word in the text. Figure 4.5 and fig. 4.6 show examples of the perturbed dataset.

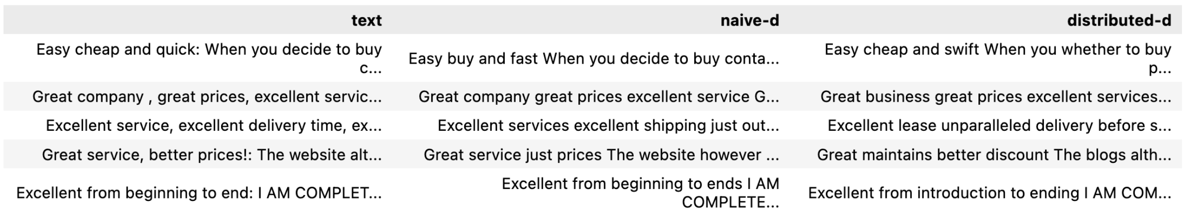
The original datasets were again perturbed for comparative analysis using the same epsilon values. However, this time, the epsilon distributor was utilized. Unlike the previous approach, the epsilon distributor assigns different epsilon values to each word based on the outcome of the epsilon distribution process. This enables a more fine-grained allocation of privacy budgets, where words with higher informativeness receive relatively higher epsilon values than less informative words.

Constructing training data using both approaches allows for directly comparing their effectiveness in preserving privacy while maintaining data utility. By applying different differential privacy mechanisms with varying strategies for epsilon allocation, we aim to evaluate their impact on the utility and privacy preservation of the datasets.



| | sentence | label | naive_dp_sentence_M | distributed_dp_sentence_M |
|---|---|---|---|---|
| 0 | Our friends won't buy this analysis, let alone... | 1 | Your friends wo not love this analysis, left i... | Your pals wo 't buy this analysis, let alone t... |
| 1 | One more pseudo generalization and I'm giving up. | 1 | One more pseudo general and O're failing up | Used more fake spectrum and He mean catching |
| 2 | One more pseudo generalization or I'm giving up. | 1 | No more pseudo general or You am giving up | So more pseudo roundup or Me're telling |
| 3 | The more we study verbs, the crazier they get. | 1 | Athe more we manipulate verbs, the tighter the... | So more we understand pronouns, the darker they |
| 4 | Day by day the facts are getting murkier. | 1 | Game by everyday the probabilities are dying w... | Hopefully by week the stats are breaking weaker |

Figure 4.5: Example of the perturbed data. We present the first four data entries from the CoLA training dataset, perturbed with DP-MLM. The first column shows the original text in the dataset, while the third and the fourth column show respectively the output of the naive- and distributed perturbation.



| text | naive-d | distributed-d |
|---|---|---|
| Easy cheap and quick: When you decide to buy c... | Easy buy and fast When you decide to buy conta... | Easy cheap and swift When you whether to buy p... |
| Great company , great prices, excellent servic... | Great company great prices excellent service G... | Great business great prices excellent services... |
| Excellent service, excellent delivery time, ex... | Excellent services excellent shipping just out... | Excellent lease unparalleled delivery before s... |
| Great service, better prices!: The website alt... | Great service just prices The website however ... | Great maintains better discount The blogs alth... |
| Excellent from beginning to end: I AM COMPLET... | Excellent from beginning to ends I AM COMPLETE... | Excellent from introduction to ending I AM COM... |

Figure 4.6: Example of the perturbed data from Trustpilot dataset, perturbed with 1-Diffractor. The first column shows the original text in the dataset while the second and the third column show respectively the output of the naive- and distributed perturbation.

### 4.2.4 Fine-tuning the Language Model

The DeBERTa-v3-base model [43] was chosen for fine-tuning due to its state-of-the-art performance in various NLP tasks and its capability to capture intricate linguistic patterns in textual data.

The fine-tuning process used the Hugging Face library[6] and PyTorch[7] framework for

[6]https://huggingface.co/docs/hub/models-libraries
[7]https://pytorch.org/

intuitive implementation and train parameter setting. The same training setup and arguments were employed for fine-tuning, including training for the single epoch.

The training dataset was shuffled before the training to introduce randomness and prevent the model from overfitting to specific patterns within the data. This ensures better generalization and robustness of the fine-tuned model across different samples.

For privacy measurement, the model was fine-tuned on the original dataset. This approach was adopted to better evaluate the effectiveness of the Epsilon distributor in preserving privacy compared to conventional methods. Conversely, the model was fine-tuned for utility measurement based on the perturbed training dataset, which had undergone privacy-preserving transformations using differential privacy mechanisms.

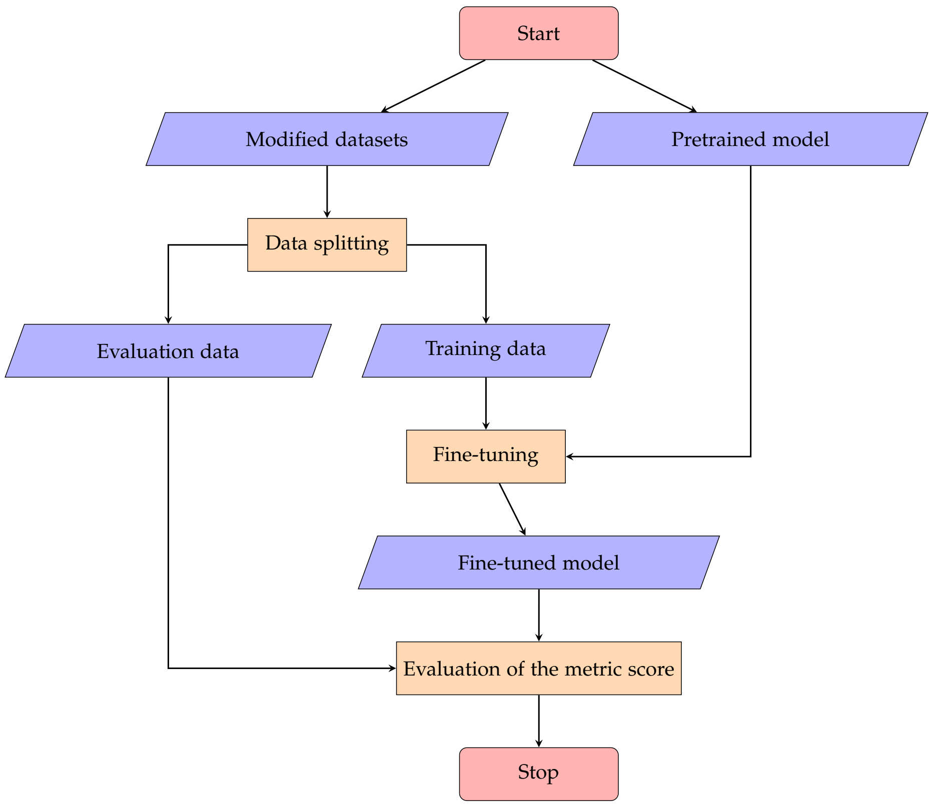


Figure 4.7: Flowchart of the general evaluation process, including fine-tuning the pre-trained model and obtaining the evaluation results.

### 4.2.5 Evaluating the Model

Finetuned models on each dataset are used for the evaluation. For privacy evaluation, two models, trained on Trustpilot and Yelp datasets, were assessed using the perturbed dataset with and without applying the epsilon distributor. The evaluation focused on accuracy as the primary metric, measuring the model's performance in correctly predicting labels: gender in

Trustpilot and author in Yelp. The lower the model's prediction performance indicates the better the privacy of the data perturbation results.

The evaluation process was conducted three times on random subsets of the dataset (20%), and the results were averaged to obtain a representative accuracy score. This approach helped mitigate the impact of dataset variability.

For the utility evaluation, evaluation datasets are used for each GLUE dataset used for pretraining respectively. The models trained on the perturbed training datasets with- and without applying the epsilon distributor were evaluated on the perturbed datasets with and without applying the epsilon distributor respectively. Each dataset was evaluated using its respective metric, which can be found in table 4.1.

Like the privacy evaluation, the utility evaluation process involved executing the evaluation three times and averaging the results. This evaluation approach ensured a comprehensive review of the quality of the perturbed data, which affects the model's performance across different NLP tasks.

# 5 Results

This chapter presents the prototype's interim result and privacy and utility evaluation results.

## 5.1 Interim Findings

To show the outcome of the prototype, an example sentence and total privacy budget is applied to the prototype with the default settings.

The prototype was initialized with the following settings:

- Example Sentence: *"After graduating from Stanford University, John Smith moved to Munich to start his new job at SAP, where he works as a software engineer."*
- Total Epsilon: 30
- Method Parameters: Default parameters
  - Information Content (IC): Enabled
  - Part-of-Speech (POS) Tag Weighting: Enabled
  - NER Weighting: Enabled
  - Word Similarity: Enabled
  - Sentence Similarity: Enabled

The numeric value of scores from the scoring methods and the final score, as well as the distribution of the privacy budget, can be found in table 5.1.

The left side of table 5.1 shows the interim results of each scoring method, the final normalized score, and the distributed epsilon value of each token. Figure 5.1 visualizes the normalized scores from the scoring methods, providing a better visual comparison.

- **IC Score**: By IC scoring method, words such as "software" received relatively higher scores due to their specificity, followed by "John", "moved", and "works".
- **POS Tag Score**: Based on the pre-defined weights, nouns such as "job", "software" and "engineer" got higher scores.
- **NER Score**: In the NER scoring method, words like "Standard" and "John" are identified as named entities and are weighted.
- **Sentence Similarity Score**: By this method, words such as "Munich", "Smith", or "SAP" got high scores since the similarity between the original sentence and sentence excluding those words was low.

- **Word Similarity Score**: Words such as "SAP", "Stanford", or "University" showed a higher similarity to the original sentence and therefore got higher scores.

| Token | IC | POS | NER | Sentence Sim. | Word Sim. | Final Score | $\epsilon$ |
|---|---|---|---|---|---|---|---|
| After | 1.0 | 0.1 | 0 | 0.0176 | 0.0510 | 0.0106 | 2.1480 |
| graduating | 185.17 | 8 | 0 | 0.0195 | 0.2069 | 0.0340 | 0.6698 |
| from | 1.0 | 0.1 | 0 | 0.0115 | 0.0431 | 0.0077 | 2.9379 |
| Stanford | 44.9 | 15 | 1 | 0.0298 | 0.3098 | 0.0544 | 0.4185 |
| University | 7410.33 | 15 | 1 | 0.0180 | 0.2483 | 0.0604 | 0.3767 |
| John | 17607.56 | 15 | 1 | 0.0492 | 0.1420 | 0.0844 | 0.2696 |
| Smith | 1740.58 | 15 | 1 | 0.1129 | 0.2719 | 0.0851 | 0.2675 |
| moved | 16475.69 | 8 | 0 | 0.0317 | 0.1485 | 0.0675 | 0.3373 |
| to | 1.0 | 0.1 | 0 | 0.0135 | 0.0534 | 0.0093 | 2.4536 |
| Munich | 129.2 | 15 | 1 | 0.1239 | 0.2350 | 0.0828 | 0.2749 |
| start | 5149.93 | 8 | 0 | 0.0111 | 0.0703 | 0.0303 | 0.7510 |
| his | 1.0 | 14 | 0 | 0.0167 | 0.1135 | 0.0326 | 0.6971 |
| new | 1.0 | 3.7 | 0 | 0.0138 | 0.0276 | 0.0119 | 1.9199 |
| job | 14954.66 | 15 | 0 | 0.0132 | 0.1162 | 0.0638 | 0.3568 |
| at | 10.4 | 0.1 | 0 | 0.0143 | 0.0551 | 0.0097 | 2.3423 |
| SAP | 300.48 | 15 | 1 | 0.0679 | 0.3545 | 0.0723 | 0.3148 |
| where | 1.0 | 0.1 | 0 | 0.0140 | 0.0693 | 0.0107 | 2.1205 |
| he | 135.9 | 14 | 0 | 0.0153 | 0.1048 | 0.0317 | 0.7177 |
| works | 17173.41 | 8 | 0 | 0.0100 | 0.0169 | 0.0505 | 0.4505 |
| as | 53.84 | 0.1 | 0 | 0.0131 | 0.0084 | 0.0057 | 4.0204 |
| a | 48.2 | 0.1 | 0 | 0.0116 | 0.0491 | 0.0083 | 2.7288 |
| software | 37852.6 | 15 | 0 | 0.0250 | 0.1338 | 0.1169 | 0.1948 |
| engineer | 1549.1 | 15 | 0 | 0.0233 | 0.2604 | 0.0512 | 0.4446 |

Table 5.1: Summary of the outcome of executing the budget distribution with the prototype for the example given in fig. 4.2. "IC", "POS", "NER", "Sentence Sim.", "Word Sim." represents the scores of each scoring method mentioned in section 4.1.1, and "Final Score" represents the combined and normalized final scores. $\epsilon$ is the assigned budget out of the total budget which was set to 30.

Each word's final score and distributed epsilon is presented on the right side of the table 5.1. Both results are also visualized in fig. 5.2. The result shows that words such as "software" and "Munich" received high scores (0.1169 and 0.0828, respectively). Those words with higher scores received smaller portions of epsilon. Words such as "from" and "to", which received fewer informativeness scores, received bigger allocations of epsilon. For example, the word "software" received 0.1948 from the total privacy budget. Conversely, the word "as" received 4.0204 out of the total epsilon.

## 5.2 Privacy & Utility Evaluation Result

### 5.2.1 Main Privacy & Utility Evaluation Result

Table 5.2 presents the complete privacy and utility evaluation results. The key information about each dataset, including the total privacy budget per text data point used for perturbation, is stated in table 4.1. The detailed experimental setup is mentioned in section 4.2.

Additionally, fig. 5.3 visualizes the result from table 5.2, emphasizing the difference between conventional (naive) and suggested (distributed) approach. The data points on the left side of the vertical line (the scores obtained using Trustpilot and Yelp datasets) in fig. 5.3 present the privacy evaluation results. The Figure compares the accuracy of privacy-preserving mechanisms, namely 1-Diffractor and DP-MLM, with and without applying the epsilon distributor across two datasets: Trustpilot and Yelp reviews. We further show for comparison the baseline score with the original dataset in fig. 5.3 and table 5.2.

The utility evaluation results for different datasets are presented on the right side of the vertical line in the panels of fig. 5.3. The model is trained and evaluated on datasets perturbed by two different mechanisms, with and without applying the epsilon distributor, to assess the impact on utility.

| Dataset | Baseline | 1-Diffractor | | DP-MLM | |
|---|---|---|---|---|---|
| | | naive | $\epsilon$-distr. | naive | $\epsilon$-distr. |
| Trustpilot | 0.701 | 0.645 | 0.622 | 0.637 | 0.610 |
| Yelp | 0.325 | 0.258 | 0.220 | 0.180 | 0.175 |
| CoLA | 0.847 | 0.718 | 0.699 | 0.691 | 0.693 |
| MRPC-acc | 0.877 | 0.850 | 0.868 | 0.762 | 0.704 |
| MRPC-F1 | 0.903 | 0.887 | 0.906 | 0.839 | 0.809 |
| SST-2 | 0.950 | 0.905 | 0.885 | 0.876 | 0.834 |
| RTE | 0.722 | 0.613 | 0.560 | 0.502 | 0.531 |
| STSB | 0.894 | 0.803 | 0.657 | 0.693 | 0.416 |

Table 5.2: Privacy and utility results with 1-Diffractor and DP-MLM various datasets. The scores represent the average of the results of three runs. The scores in the "Baseline" column represent the results on original data, while "1-Diffractor" and "DP-MLM" shows the results on the modified data, perturbed by them respectively. Meanwhile "naive" contains the result from the perturbed data, with equally allocated privacy budget to each token, while "$\epsilon$-distr." shows the result from the ones with distributed $\epsilon$ values calculated by the prototype. The evaluation index for all datasets is the accuracy. For MRPC the extra metric (F1 score) is used. For the STSB dataset, the Pearson-Spearman correlation is used.

### 5.2.2 Sub-experiment: Stop-word Filtering

Additional experiments and privacy evaluation were conducted, with datasets perturbed without the stop-words filtering option of the DP mechanisms in section 4.2.2 and with a stricter privacy budget. For this sub-experiment, the half of the Trustpilot sample dataset was used. Table 5.3 shows the results of this additional experiment, in which the Trustpilot dataset is used. The first row represents the result of the main experiment from table 5.2 of Trustpilot, which was perturbed in default mechanism parameters including stop-word filtering. The other two rows present the results obtained by using the Trustpilot dataset, perturbed without the stop-word filtering option in each mechanism. For the Trustpilot (Stop 1/2), half of the default budget was applied for comparison.

| Dataset | Baseline | 1-Diffractor | | | | DP-MLM | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | budget | naive | $\epsilon$-distr. | diff. | budget | naive | $\epsilon$-distr. | diff. |
| Trustpilot (Ref.) | 0.693 | 45 | 0.645 | 0.622 | $-0.023$ | 4500 | 0.637 | 0.610 | $-0.027$ |
| Trustpilot (Stop) | 0.693 | 45 | 0.628 | 0.612 | $-0.016$ | 4500 | 0.584 | 0.581 | $-0.003$ |
| Trustpilot (Stop 1/2) | 0.693 | 22 | 0.595 | 0.576 | $-0.019$ | 2200 | 0.579 | 0.562 | $-0.017$ |

Table 5.3: Privacy score (accuracy) comparison of stop words filtering and applying a smaller privacy budget. As a reference, the first row, "Trustpilot (Ref.)", is taken from the Trustpilot results in table 5.2. "Trustpilot (Stop)" is a dataset which is perturbed without the stop-word filtering option of the two mechanisms. "Trustpilot (Stop 1/2)" is a dataset generated from data perturbation without stop-word filtering and with a stricter (lower) privacy budget. The "diff." column shows the difference between the score obtained with the $\epsilon$ distribution and the score obtained with the naive distribution. All shown scores are the average of three independent runs.

### 5.2.3 Sub-experiment: Word-level vs. Sentence-level Privacy Budget

Another sub-experiment was conducted to show the impact of privacy budget distribution in individual privacy budget applications. It is one of the conventional word-level approaches for deciding the epsilon values of the text data. Contrary to the main experiment, where the epsilon values of the entire dataset are fixed with the average length of the text data points, this approach uses individual privacy budgets for each data point based on the size of the respective single text data entry. Table 5.4 presents the results of the privacy evaluation on the Trustpilot and Yelp data, for which the data were perturbed with 1-Diffractor.

| Dataset | Baseline | Individual budget | | | | Fixed budget | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | budget | naive | $\epsilon$-distr. | diff. | budget | naive | $\epsilon$-distr. | diff. |
| Trustpilot | 0.693 | len(text) | 0.671 | 0.618 | $-0.053$ | 45 | 0.645 | 0.622 | $-0.023$ |
| Yelp | 0.325 | len(text) | 0.303 | 0.195 | $-0.108$ | 182 | 0.258 | 0.220 | $-0.038$ |

Table 5.4: Results of the privacy scores for different budget settings. For the data perturbation in individual budget setting, the privacy budget is determined individually based on the word counts of each data point and applied respectively. The fixed budget setting refers to the default setting used in table 4.1. The "budget" column represents the total privacy budget used for each text in the datasets, and the others are the same as described in table 5.3. 1-Diffractor is used for the data perturbation. The random subsets (10%) of the datasets are used for the evaluation.

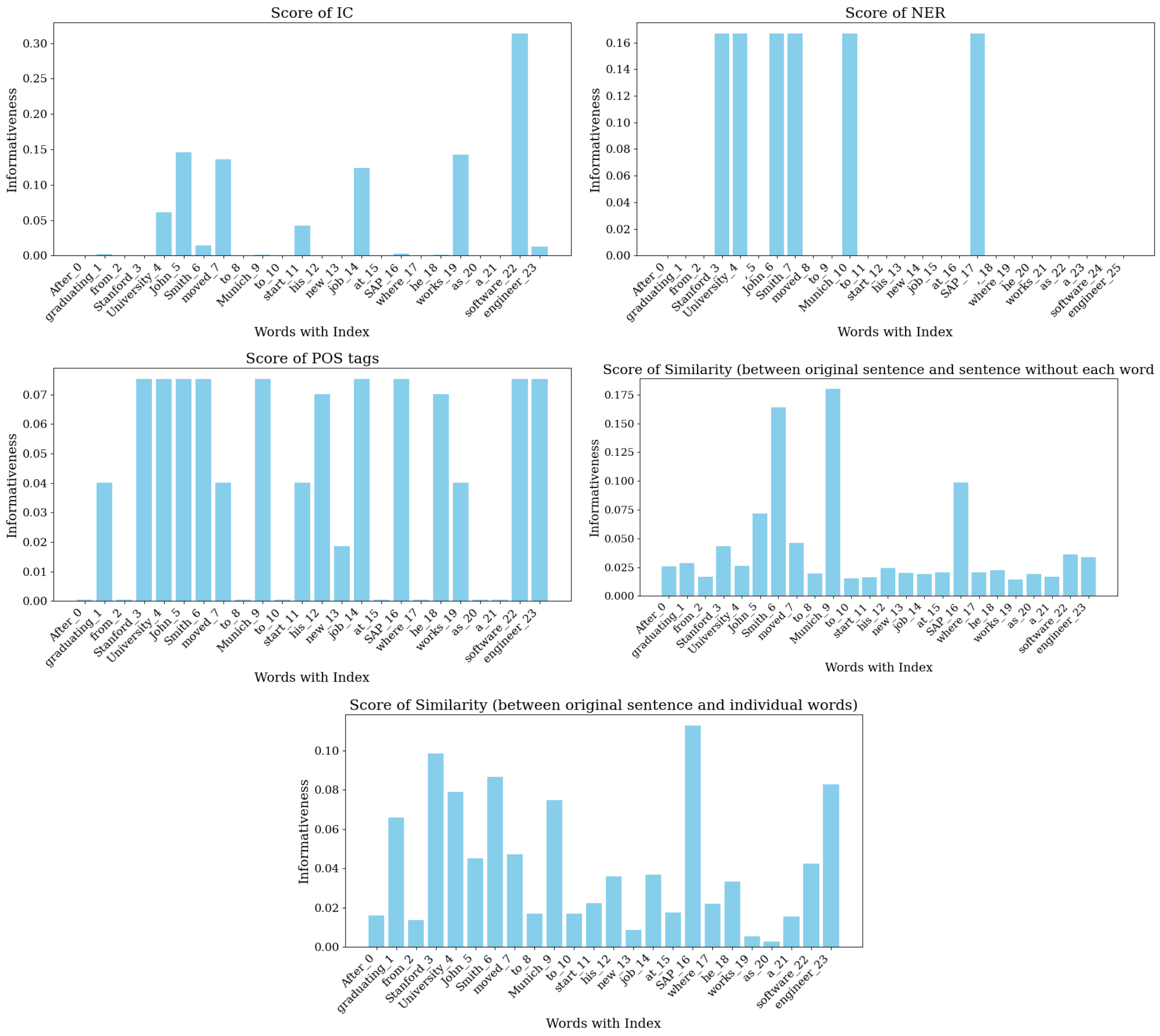


Figure 5.1: Normalized results of each scoring method. The values on the vertical axis correspond to the normalized informativeness score of each scoring method.

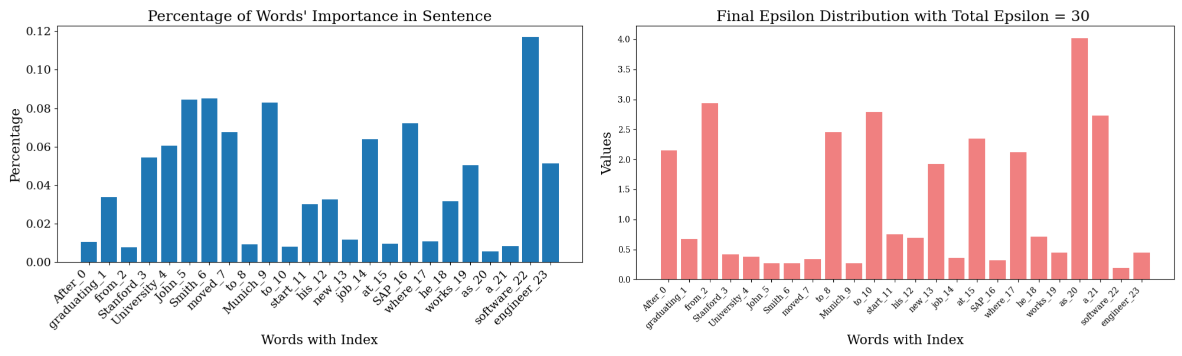


Figure 5.2: Result of combining scores from scoring method (see fig. 5.1) and final distribution of epsilon. The values on the vertical axis of the left chart correspond to the relative informativeness of corresponding words in the sentence as percentages. On the right plot the distributed epsilon values are shown on the vertical axis, given the total epsilon of 30.

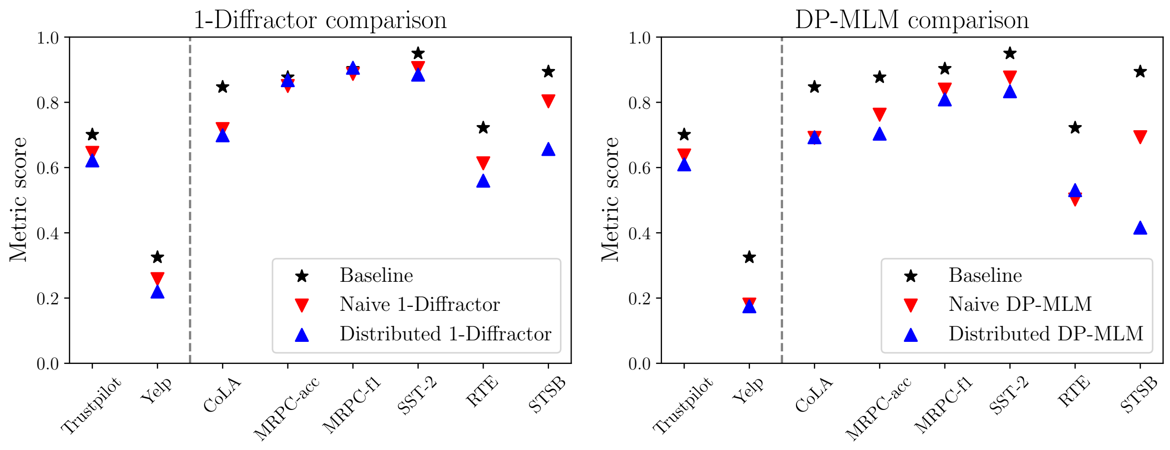


Figure 5.3: Comparison between 1-Diffractor (left) and DP-MLM (right) utilizing the naive- and the distributed approach for different datasets. The values on the vertical axis correspond to the score on the dataset-dependent metric; see table 4.1. Contrary to the other datasets, a lower metric score indicates higher privacy for Trustpilot and Yelp.

# 6 Discussion

## 6.1 Key Findings

The analysis of the experiment results revealed several key findings.

**The consistent improvement of privacy** First, the experiments demonstrated that the proposed method of intelligently distributing the privacy budget within sentences can consistently enhance privacy. The result of the main experiments, shown in tables 5.2 to 5.4, proves that, in all considered configurations, the data perturbed with the privacy budget distribution prototype recorded lower scores. This reflects better privacy preservation.

In the Trustpilot dataset, applying the epsilon distributor leads to a decrease in accuracy from 0.645 to 0.622 for 1-Diffractor and from 0.637 to 0.610 for DP-MLM. Similarly, in the Yelp dataset, the accuracy decreases from 0.258 to 0.220 for the 1-Diffractor and from 0.180 to 0.175 for DP-MLM when using the epsilon distributor. In fig. 5.3, this impact of the epsilon distributor on the data is visualized. On the left side of the vertical line in both panels, scores from the perturbation with a naively distributed privacy budget (red points) are higher than those using the suggested epsilon distributor prototype (blue points), indicating consistent privacy improvement.

Also, the result from the other two sub-experiments—in which stop-word filtering was disabled or different epsilon values were applied—indicates a consistent improvement in terms of privacy protection. Table 5.4 shows that with the Trustpilot dataset, the accuracy scored lower when the $\epsilon$ distribution is applied, no matter which mechanism was used. Furthermore, privacy protection showed better results when the data is perturbed with and without stop word filtering, as shown in table 5.3. For instance, with the stop-word replacement option, the accuracy decreased from 0.628 to 0.612 with 1-Diffractor. Similarly, with DP-MLM, privacy protection improved slightly (from 0.584 to 0.581). With stricter (smaller) epsilon values, the degree of improvement is more significant. The accuracy shrunk from 0.595 to 0.576 with 1-Diffractor and from 0.579 to 0.562 with DP-MLM. Compared to the above results, the magnitude of the difference has increased from 0.016 to 0.019 and from 0.003 to 0.017.

In summary, the results demonstrate that in general, the epsilon distributor effectively enhances privacy preservation regardless of the setting, including choosing a privacy budget or perturbation mechanism.

**Maintenance and loss of utility** Furthermore, in many cases, the utility has been maintained. Overall, the results suggest that applying the epsilon distributor has maintained the utility, mostly resulting the similar performance scores observed across the datasets.

In table 5.2, with datasets like SST-2, MRPC, and CoLA, the utility shows certain consistency between the naively perturbed dataset and the dataset perturbed with the epsilon distributor. For example, in the CoLA dataset, the accuracy scored lower when using epsilon distributor on 1-Diffractor (from 0.718 to 0.699); however, the accuracy increased on DP-MLM (from 0.691 to 0.693). Similarly, the accuracy with 1-Diffractor showed a slight improvement from 0.850 to 0.868, while the score with DP-MLM decreased from 0.762 to 0.704. Also, in RTE, the metric showed both improvement and decline depending on the perturbation mechanism.

However, it was also observed that there were instances where the utility scores decreased, particularly in certain datasets and with specific differential privacy mechanisms. Some datasets, such as STSB and SST-2, display noticeable differences in utility between the perturbed datasets with and without applying the epsilon distributor. The result in the SST-2 dataset shows a loss of utility scores. The accuracy score declined when using 1-Diffractor with the epsilon distributor compared to the naively perturbed dataset (from 0.905 to 0.885). The score drops from 0.876 to 0.834 using DP-MLM. In the STSB dataset, the loss was bigger. Using 1-Diffractor with the epsilon distributor decreases the Pearson-Spearman correlation from 0.803 to 0.657 compared to the naively perturbed dataset. Similarly, the correlation coefficient decreases from 0.693 to 0.416 in the same scenario but using DP-MLM.

In fig. 5.3, the comparison of utility measurements can be seen on both panels' right side of the vertical line. The utility scores generally remained similar or decreased by a relatively small difference. However, in some datasets, the loss is noticeable.

These variations in utility scores across datasets and metrics may suggest that the impact of applying the epsilon distributor may depend on the dataset characteristics, task requirements, and specific differential privacy mechanisms used. For example, the STSB dataset, which is, in contrast to other datasets, a regression task, where models predict a continuous similarity score between pairs of sentences. It can be assumed that the utility loss is particularly large in this data set because of the nature of this task. Therefore, further investigations on various aspects are necessary to ensure consistent utility preservation in differential privacy applications for NLP tasks.

**Additional insights on budget choice and stop-word filtering** Via the results shown in table 5.3, we can note that the effect of improving privacy remains even when the stop-word filtering option of the mechanisms are disabled. Similarly to the result using default settings of 1-Diffractor and DP-MLM, the privacy scores of the result using the stop-word filtering setting are also decreased when the epsilon distribution was applied.

Another notable point is that both mechanisms showed better information protection (low accuracy) when no stop word filtering was applied. For the data presented in table 5.3, privacy was measured higher when data was modified with the stop-word filtering turned off (Trustpilot (Stop)), compared to when the data was disturbed by the default setting (Trustpilot (ref.)). There may be several interpretations of these results. One possible argument is that when stop-words filtering is applied, the budgets are already distributed to the stop-words. Still, due to the filtering, the modifications of those words are ignored. Because the budget is wasted, less will be assigned to words other than the stop-words, causing more disturbance and higher privacy levels.

Another finding is related to the limitation of the total budget. It was found that the more limited the budget, the more difference there was in improving privacy. Comparing the last two rows from table 5.3, the privacy improvement was $-0.016$ and $-0.003$ with 1-Diffractor and DP-MLM, respectively. However, when the budget was cut in half, the difference increased to $-0.019$ and $-0.017$ respectively. Especially for DP-MLM, the margin of improvement has increased by more than five times. This result shows that applying the epsilon distribution can achieve a high degree of privacy improvement when the budget is limited in the same environment. The reason for this result can be explained as follows: tightening the budget can result in the wasteful application of less budgeting to non-important words and, therefore, forcing higher privacy protection to those words, which is likely to disrupt all words more than necessary. Thus, distributing fewer budgets can be more critical to the privacy outcome.

Finally, the results of the sub-experiment conducted in section 5.2.3 indicate how budget selection affects privacy evaluation. Table 5.4 shows the privacy scores resulting from perturbed data with 1-Diffractor with different budget settings. Unlike the fixed budget approach, where the privacy budget is a fixed value for the whole dataset and which was used as the default in this experiment, in the individual budget approach, the budget applied to each data point in the dataset varies. The personal budget for text depends on the number of characters in the text. Although it can be difficult to directly compare the result values of the two approaches due to differences in the selection of training and evaluation data within the dataset, these results show that the overall privacy improvement was more significant in the individual budget approach than the fixed budget approach. In both Trustpilot and Yelp data sets, the degree of improvement in the individual budget setting was larger ($-0.053$ and $-0.108$ respectively for Trustpilot and Yelp) than in the fixed budget environment ($-0.023$ and $-0.038$). These results can be interpreted as privacy improvement through the epsilon distributor, which can be greater when an appropriate budget is selected for each data. In the individual budget setting, the budget is determined and allocated to each data according to its length. Therefore, it could mean that applying the epsilon distribution significantly improves privacy, which means that setting the privacy budget according to the data can be an important factor in improving the results of using distribution.

## 6.2 Implications and Contributions

This work makes several contributions to the field of privacy-preserving NLP. Firstly, it introduces an approach to distributing the privacy budget within sentences, potentially enhancing privacy protection in NLP applications. By incorporating linguistic methods for estimating the amount of information in words, this approach provides a more detailed understanding of word importance, allowing more accurate allocation of privacy budgets at the sentence level.

Additionally, the prototype developed in this study is designed to be extendable and customizable, allowing the users to adapt it to their specific needs and requirements. As a framework that provides privacy budget distribution, which is directly applicable to differential privacy mechanisms, this prototype offers flexibility and scalability for privacy NLP applications.

As a result, this study brings forth several significant implications:

**Exploration of New Approaches** By pioneering a novel approach to distributing privacy budgets at the sentence level and quantifying informativeness, this research opens new avenues in NLP differential privacy. Departing from the conventional equal distribution of privacy budgets across all words, it proposes a fresh alternative and validates its efficacy. This advancement in theoretical understanding, particularly the attempt to adapt DP at sentence-level NLP, contributes to a broader discourse on privacy preservation.

**Practical Insights** The practical evaluation of the proposed framework provides valuable insights into its effectiveness in safeguarding privacy while preserving the utility of textual data across various NLP applications. This research bridges the gap between theoretical advancements and pragmatic implementations in privacy-preserving NLP by advancing a practical solution of applying DP in textual data tailored to real-world scenarios with finite privacy budgets.

## 6.3 Challenges and Future Improvements

Although the proposed approach has been shown to be promising for addressing privacy issues at the sentence level within NLP, it has faced several challenges during this research. These challenges highlight areas of improvement in the present thesis and suggest future research directions to improve the effectiveness and applicability of the proposed methodology.

One of the primary challenges faced during the research is the exploration of the linguistic method of estimating the amount of information in words. There have been many studies on information analysis or vectorization of textual data. Still, finding a process or study to quantify this information absolutely or relatively in linguistic terms was challenging. Furthermore, the scoring methods in the suggested prototype mostly rely on statistical approaches because of the lack of research on informativeness or word significance using a semantic approach. This could cause an imbalance in measuring informativeness since exploring semantic features may provide deeper insights into the importance of words in context. Further research could, therefore, investigate the use of linguistic methods for calculating word informativeness, such as analyzing the information content of different parts of speech. Understanding which parts of speech convey more information linguistically could lead to more accurate scoring methods within the prototype.

Expanding this prototype is also a challenge left for the future. Adding additional scoring methods and adjusting more reasonable weights assigned to each technique could enhance the overall effectiveness of the prototype. Conducting individual evaluations of each scoring method to determine their impact on privacy- and utility metrics could help optimize the scoring process.

Another limitation of this study was determining the budget. The privacy budget used in the experiment of this study cannot be said to be the most appropriate since each mechanism has a difference in scale, and it is difficult to estimate the degree of its impact on the data perturbation. For the uniformity of time constraints and experimental environment settings, this study applied one criterion; the basic privacy budget was set for each mechanism (1 for 1-Diffractor and 100 for DP-MLM), and the privacy budget for each dataset was fixed with the average number of words in the text data in the dataset. Testing the prototypes with different privacy budgets could give more insight into the impact of varying epsilon values on the effectiveness of the suggested approach. This exploration may uncover optimal privacy budget settings for different datasets and applications.

Additionally, testing the effectiveness of the proposed method in other environments or settings not performed in this study could be one of the future work to be done. For example, experimenting with DP mechanisms beyond those used in the current study could offer insights into the comparative effectiveness of different approaches.

Conducting additional tests under different settings and conditions, such as choosing different pretrained models or using various parameters in the fine-tuning and evaluation steps, could help identify factors that influence the robustness and stability of the proposed approach.

Finally, as computational resources are often limited, considering computational efficiency improvements in the prototype could enhance its practicality and scalability.

# 7 Conclusion

Ensuring individual confidentiality is a critical concern, particularly as the volume of text-based data is rapidly increasing across various natural language processing (NLP) applications. Differential privacy (DP) emerges as a promising framework to address these issues, providing strong guarantees through statistical methods and enabling meaningful data analysis while protecting sensitive information within datasets. Recently, research on the practical use of DP has been active in the field of NLP. Nevertheless, since most approaches focus on the word level, there is a need to explore DP at the sentence level. Additionally, research on privacy budget, an essential element in DP, remains relatively unexplored from an NLP perspective.

This thesis aimed to bridge this gap by proposing a new approach. To address this goal, we suggested to apply DP at the sentence level in the context of NLP by distributing the privacy budget to individual tokens within the sentence. The main idea of this approach is to calculate an individualized budget based on the quantified informativeness of each token in the sentence.

To investigate this idea, we developed a prototype that, given a text, calculates the relative informativeness of tokens within it and distributes a given privacy budget based on this. This prototype includes five linguistic base scoring methods and allows users to select a method or transform detailed weights. The evaluation of the proposed method comprises *i*) the application of the prototype to an actual DP mechanism to convert the data and *ii*) fine-tuning and evaluating a pre-trained language model on the perturbed data.

The evaluation's overall result demonstrates that applying the epsilon distribution generally increases privacy preservation while maintaining the utility of the text data. Our evaluation results consistently showed that using the epsilon distribution method led to lower privacy accuracy scores. This indicates improved privacy protection compared to a conventional naive approach, which distributes the given privacy budget equally to every token in the text. Importantly, this privacy enhancement was achieved without significant loss of utility across various datasets and differential privacy mechanisms.

In terms of contributions, our research advances the theoretical understanding of DP in the context of NLP, particularly at the sentence level, contributing to broader discussions on privacy preservation. Furthermore, our practical evaluation of the proposed framework provides insights into its effectiveness for protecting sensitive information in diverse NLP applications. Our work addresses critical challenges in privacy-preserving NLP by offering a usable solution for practical use cases with finite privacy budgets.

Future research may explore ways to further improve our approach, incorporating semantic methods for quantifying token informativity and exploring alternative mechanisms for privacy. Moreover, extending the scope of the analysis to a variety of datasets and NLP tasks will provide deeper insights into the generalizability and robustness of our approach.

# List of Figures

# List of Tables

# Bibliography

[1] C. Dwork. “Differential Privacy”. In: *Automata, Languages and Programming*. Ed. by M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12. ISBN: 978-3-540-35908-1.

[2] 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*. 2016, pp. 308–318.

[3] 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.

[4] A. Bkakria, A. Tasidou, N. Cuppens-Boulahia, F. Cuppens, F. Bouattour, and F. Ben Fredj. “Optimal Distribution of Privacy Budget in Differential Privacy”. In: *Risks and Security of Internet and Systems*. Ed. by A. Zemmari, M. Mosbah, N. Cuppens-Boulahia, and F. Cuppens. Cham: Springer International Publishing, 2019, pp. 222–236. ISBN: 978-3-030-12143-3.

[5] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi. “Differential Privacy: On the Trade-Off between Utility and Information Leakage”. In: *Formal Aspects of Security and Trust*. Ed. by G. Barthe, A. Datta, and S. Etalle. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 39–54. ISBN: 978-3-642-29420-4.

[6] O. Klymenko, S. Meisenbacher, and F. Matthes. “Differential Privacy in Natural Language Processing The Story So Far”. In: *Proceedings of the Fourth Workshop on Privacy in Natural Language Processing*. Association for Computational Linguistics, 2022. DOI: `10.18653/v1/2022.privatenlp-1.1`. URL: `http://dx.doi.org/10.18653/v1/2022.privatenlp-1.1`.

[7] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O’Reilly Media, Inc.", 2009.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems* 26 (2013).

[9] D. Cer, Y. Yang, S. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y. Sung, B. Strope, and R. Kurzweil. “Universal Sentence Encoder”. In: *CoRR* abs/1803.11175 (2018). arXiv: `1803.11175`. URL: `http://arxiv.org/abs/1803.11175`.

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018). arXiv: `1810.04805`. URL: `http://arxiv.org/abs/1810.04805`.

[11] A. Radford and K. Narasimhan. "Improving Language Understanding by Generative Pre-Training". In: 2018. URL: `https://api.semanticscholar.org/CorpusID:49313245`.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[13] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Ed. by Q. Liu and D. Schlangen. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: `10.18653/v1/2020.emnlp-demos.6`. URL: `https://aclanthology.org/2020.emnlp-demos.6`.

[14] P. Resnik. "Using information content to evaluate semantic similarity in a taxonomy". In: *arXiv preprint cmp-lg/9511007* (1995).

[15] C. Fellbaum. *WordNet: An electronic lexical database*. MIT press, 1998.

[16] R. Arumugam and R. Shanmugamani. *Hands-On Natural Language Processing with Python: A practical guide to applying deep learning architectures to your NLP applications*. Packt Publishing, 2018. ISBN: 9781789135916. URL: `https://books.google.de/books?id=ipplDwAAQBAJ`.

[17] S. Petrov, D. Das, and R. McDonald. "A Universal Part-of-Speech Tagset". In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Ed. by N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012, pp. 2089–2096. URL: `http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf`.

[18] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". In: *Proceedings of the Eighteenth International Conference on Machine Learning*. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289. ISBN: 1-55860-778-1. URL: `http://dl.acm.org/citation.cfm?id=645530.655813`.

[19] D. Nadeau and S. Sekine. "A survey of named entity recognition and classification". In: *Lingvisticae Investigationes* 30 (2007), pp. 3–26. URL: `https://api.semanticscholar.org/CorpusID:8310135`.

[20] Y. Vasiliev. *Natural language processing with Python and spaCy: A practical introduction*. No Starch Press, 2020.

[21] N. Reimers and I. Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks". In: *arXiv preprint arXiv:1908.10084* (2019).

[22] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5776–5788.

[23] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[24] L. Hu, I. Habernal, L. Shen, and D. Wang. “Differentially private natural language models: Recent advances and future directions”. In: *arXiv preprint arXiv:2301.09112* (2023).

[25] T. Diethe, O. Feyisetan, B. Balle, and T. Drake. “Preserving privacy in analyses of textual data”. In: (2020).

[26] W. Shi, A. Cui, E. Li, R. Jia, and Z. Yu. “Selective differential privacy for language modeling”. In: *arXiv preprint arXiv:2108.12944* (2021).

[27] J. Mattern, B. Weggenmann, and F. Kerschbaum. “The limits of word level differential privacy”. In: *arXiv preprint arXiv:2205.02130* (2022).

[28] C. Meehan, K. Mrini, and K. Chaudhuri. *Sentence-level Privacy for Document Embeddings*. 2022. arXiv: `2205.04605 [cs.LG]`.

[29] L. Rosenblatt, J. Allen, and J. Stoyanovich. “Spending Privacy Budget Fairly and Wisely”. In: *arXiv preprint arXiv:2204.12903* (2022).

[30] A. Bkakria, A. Tasidou, N. Cuppens-Boulahia, F. Cuppens, F. Bouattour, and F. Ben Fredj. “Optimal distribution of privacy budget in differential privacy”. In: *Risks and Security of Internet and Systems: 13th International Conference, CRiSIS 2018, Arcachon, France, October 16–18, 2018, Revised Selected Papers 13*. Springer. 2019, pp. 222–236.

[31] A. M. Schakel and B. J. Wilson. “Measuring word significance using distributed representations of words”. In: *arXiv preprint arXiv:1508.02297* (2015).

[32] K. Kireyev. “Semantic-based estimation of term informativeness”. In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2009, pp. 530–538.

[33] Z. Wu and C. L. Giles. “Measuring term informativeness in context”. In: *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies*. 2013, pp. 259–269.

[34] A. Madsen, N. Meade, V. Adlakha, and S. Reddy. “Evaluating the faithfulness of importance measures in nlp by recursively masking allegedly important tokens and retraining”. In: *arXiv preprint arXiv:2110.08412* (2021).

[35] Q. Liu, Z.-H. Ling, H. Jiang, and Y. Hu. “Part-of-Speech Relevance Weights for Learning Word Embeddings”. In: *arXiv e-prints*, arXiv:1603.07695 (Mar. 2016), arXiv:1603.07695. DOI: `10.48550/arXiv.1603.07695`. arXiv: `1603.07695 [cs.CL]`.

[36] C. Lioma and R. Blanco. *Part of Speech Based Term Weighting for Information Retrieval*. 2017. arXiv: `1704.01617 [cs.IR]`.

[37] T. Klammer, M. Schulz, and A. Volpe. *Analyzing English Grammar*. Longman, 2010. ISBN: 9780205685943. URL: `https://books.google.de/books?id=1dbRPgAACAAJ`.

[38] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. Smith. "Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments". In: vol. 2. Jan. 2011, pp. 42–47.

[39] A. Felippo, N. Roman, T. Pardo, and L. Moura. *THE DANTESTOCKS CORPUS: AN ANALYSIS OF THE DISTRIBUTION OF UNIVERSAL DEPENDENCIES-BASED PART OF SPEECH TAGS*. Nov. 2022. DOI: `10.36227/techrxiv.21594384.v1`.

[40] D. Hovy, A. Johannsen, and A. Søgaard. "User Review Sites as a Resource for Large-Scale Sociolinguistic Studies". In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15. Florence, Italy: International World Wide Web Conferences Steering Committee, 2015, pp. 452–461. ISBN: 9781450334693. DOI: `10.1145/2736277.2741141`. URL: `https://doi.org/10.1145/2736277.2741141`.

[41] X. Zhang, J. Zhao, and Y. LeCun. "Character-level Convolutional Networks for Text Classification". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc., 2015. URL: `https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf`.

[42] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: In the Proceedings of ICLR. 2019.

[43] P. He, J. Gao, and W. Chen. *DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing*. 2021. arXiv: `2111.09543 [cs.CL]`.