DATA AUGMENTATION FOR NATURAL LANGUAGE PROCESSING

DOĞAL DİL İŞLEME İÇİN VERİ ARTIRMA

MUSTAFA ÇATALTAŞ

PROF. DR. İLYAS ÇİÇEKLİ

Supervisor

ASSOC. PROF. DR. NURDAN BAYKAN

Co-Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

July 2024

ABSTRACT

DATA AUGMENTATION FOR NATURAL LANGUAGE PROCESSING

Mustafa ÇATALTAŞ

Master of Science, Computer Engineering

Supervisor: Prof. Dr. İlyas ÇİÇEKLİ

Co-Supervisor: Assoc. Prof. Dr. Nurdan BAYKAN

July 2024, 88 pages

Advanced deep learning models have greatly improved various natural language processing tasks. While they perform best with abundant data, acquiring large datasets for each task is not always easy. Therefore, by using data augmentation techniques, comprehensive data sets can be obtained by creating synthetic samples from existing data. This thesis undertakes an examination of the efficacy of autoencoders as a textual data augmentation technique targeted at improving the performance of classification models in text classification tasks. The analysis encompasses the comparison of four distinct autoencoder types: Traditional Autoencoder (AE), Adversarial Autoencoder (AAE) Denoising Adversarial Autoencoder (DAAE) and Variational Autoencoder (VAE). Moreover, the study investigates the impact of different word embedding types, preprocessing methods, label-based filtering, and the number of epochs for training on the performance of autoencoders. Experimental evaluations are conducted using the SST-2 sentiment classification dataset, consisting of 7791 training instances. For data augmentation experiments, subsets of 100, 200, 400, and 1000 randomly selected

instances from this dataset were employed. Experimental evaluations involved augmenting data at ratios of 1:1, 1:2, 1:4, and 1:8 when working with small datasets. Comparative analysis with baseline models demonstrates the superiority of AE-based data augmentation methods at a 1:1 augmentation ratio. These findings underscore the effectiveness of using autoencoders as data augmentation methods for optimizing text classification performance in NLP applications.

Keywords: Natural Language Processing, Autoencoders, Data Augmentation, Text Classification

ÖZET

DOĞAL DİL İŞLEME İÇİN VERİ ARTIRMA

Mustafa ÇATALTAŞ

Yüksek Lisans, Bilgisayar Mühendisliği
Danışman: Prof. Dr. İlyas ÇİÇEKLİ
Eş Danışman: Doç. Dr. Nurdan BAYKAN
Temmuz 2024, 88 sayfa

Gelişmiş derin öğrenme modelleri, çeşitli doğal dil işleme (DDİ) görevlerinin etkinliğini büyük ölçüde artırmıştır. Bu modeller genellikle bol miktarda veriyle en iyi performansı gösterirken, her görev için büyük veri kümeleri elde etmek her zaman kolay olmamaktadır. Bu nedenle, veri artırma teknikleri kullanılarak, mevcut veriden sentetik örnekler oluşturarak kapsamlı veri kümelerinin elde edilmesi sağlanabilmektedir. Bu tez, metin sınıflandırma görevlerinde sınıflandırma modellerinin performansını artırmayı amaçlayan bir metinsel veri artırma tekniği olarak otokodlayıcıların etkililiğini incelemektedir. Analiz, Geleneksel Otokodlayıcı (GO), Değişimsel Otokodlayıcı (DO), Çekişmeli Otokodlayıcı (ÇO) ve Gürültü Önleyici Çekişmeli Otokodlayıcı (GÖÇO) olmak üzere dört farklı otokodlayıcı türünün karşılaştırılmasını kapsamaktadır. Ayrıca çalışma; farklı kelime gömme (temsil) türlerinin, ön işleme yöntemlerinin, etiket tabanlı filtrelemenin ve eğitme sayılarının otokodlayıcıların performansı üzerindeki etkisini araştırmaktadır. Deneysel çalışmalarda 7791 eğitim verisine sahip SST-2 duygu sınıflandırma veri seti kullanılmıştır. Veri arttırma çalışmaları için bu veri setinden rastgele seçilmiş 100, 200, 400 ve 1000 boyutundaki verilerle çalışılmıştır. Deneysel değerlendirmelerde, küçük veri setlerinde çalışırken 1:1, 1:2, 1:4 ve 1:8 oranlarında veri arttırma yapılmıştır. Temel modellerle karşılaştırmalı analizler, arttırma oranı 1:1'de GO tabanlı veri arttırma yöntemlerinin üstünlüğünü göstermektedir. Bu bulgular,

otokodlayıcıların, doğal dil işleme uygulamalarındaki metin sınıflandırma performansını optimize etmek için veri arttırma yöntemleri olarak kullanılmasının etkililiğini vurgulamaktadır.

Anahtar Kelimeler: Doğal Dil İşleme, Otokodlayıcılar, Veri Artırma, Metin Sınıflandırma

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisors, Prof. Dr. İlyas ÇİÇEKLİ and Assoc. Prof. Dr. Nurdan BAYKAN, for their invaluable guidance, and insightful feedback throughout the entirety of this research journey. Their expertise, encouragement, and patience have been instrumental in shaping this thesis and enhancing my academic growth.

I am also profoundly thankful to my wife, Gözde Düriye ÇATALTAŞ for her unwavering love, understanding, and endless encouragement during this challenging yet rewarding endeavor. Her unwavering support, sacrifices, and belief in my abilities have been my pillars of strength, motivating me to persevere through the obstacles and strive for excellence.

Finally, I would like to express my heartfelt gratitude to all my friends and family members who have supported me in many ways throughout this journey.

CONTENTS

FIGURES

TABLES

ABBREVIATIONS

NLP	:	**N**atural **L**anguage **P**rocessing
DA	:	**D**ata **A**ugmentation
DL	:	**D**eep **L**earning
ML	:	**M**achine **L**earning
LSTM	:	**L**ong **S**hort-**T**erm **M**emory
AE	:	**A**uto **E**ncoder
AAE	:	**A**dversarial **A**uto **E**ncoder
DAAE	:	**D**enoising **A**dversarial **A**uto **E**ncoder
VAE	:	**V**ariational **A**uto **E**ncoder
W2V	:	**W**ord**2V**ec
BoW	:	**B**ag-**o**f-**W**ords
Seq2Seq	:	**Seq**uence-to-**Seq**uence
TF-IDF	:	**T**erm **F**requency-**I**nverse **D**ocument **F**requency
RNN	:	**R**ecurrent **N**eural **N**etworks

1. INTRODUCTION

Over the past few decades, there have been a dramatic rise in the use of Deep Learning (DL) methodologies within the field of Natural Language Processing (NLP) which caused by remarkable efficacy of DL techniques in handling complex linguistic structures [1]. Thanks to emergence of advanced DL architectures such as Attention-based models [2] and Long Short-Term Memory (LSTM) [3], which have exhibited superior capabilities in capturing semantic information and long-range dependencies inherent in natural language data [4], significant progress have been achieved in tasks such as text classification and language modeling. Moreover, advancements in computational power of hardware infrastructures and parallel processing [5] have contributed to larger DL models, enabling DL models to capture more linguistic nuances [6]. As a result, most NLP applications have utilized DL models to obtain improved performance such as question answering, and sentiment analysis [7-11].

One drawback of DL models is their requirement of large volumes of data which cannot be supplied in every scenario [12]. To overcome this problem, different approaches are employed, such as transfer learning and data augmentation (DA) [13]. While all of these have contributed to tackling the problem, DA is the only solution which can increase the volume of the data used without having a change on the DL model.

DA serves as a pivotal technique in Machine Learning (ML), providing the expansion and enrichment of training datasets by introducing diverse variations to existing data samples [14]. Widely employed across various ML domains, this method addresses limitations posed by limited or imbalanced data, enhancing model performance and robustness [15]. Tailored augmentation strategies, such as flipping and rotating for image classification or paraphrasing and word dropout for sentiment analysis, underscore its adaptability to specific tasks, optimizing model efficacy amidst real-world complexities. Particularly in NLP, where the demand for diverse and abundant data is paramount, augmentation techniques hold high significance, promoting advancements in model generalization and performance through the manipulation of linguistic structures and semantics.

Text classification, a crucial aspect of NLP, involves assigning labels to textual documents according to their content, including sentiment analysis, spam detection, and

topic categorization [16]. While traditional methods like Naive Bayes and decision trees have been effective, they struggle with capturing semantic nuances. Deep learning methods, including transformers and LSTMs, have become effective instruments for text classification by encoding semantic details with exceptional performance [17]. Similarly, sentiment classification, a subtask of text classification, focuses on identifying emotional tone in text [18], with deep learning techniques revolutionizing the field by capturing contextual information and sentiment nuances effectively.

This thesis explores the application of autoencoders for textual DA in the context of text classification. Various autoencoder architectures and data augmentation strategies were examined on the SST-2 dataset. Furthermore, the influence of augmented data on the performance and robustness of text classification models is evaluated. By leveraging the power of autoencoders for DA, it is aimed to enhance the scalability, efficiency, and effectiveness of text classification systems, particularly in situations where obtaining labeled data is limited or costly.

The structure of this thesis is as outlined: an introduction to the primary focus of the thesis is given in Section 1. Section 2 gives detailed information on fundamentals of the thesis. Section 3 examines existing literature on textual DA methods and AE-based DA methods. Section 4 introduces methods used in this thesis. Section 5 presents findings on dataset comparisons and method evaluations. Section 6 provides a summary of significant discoveries and suggests implications for future research endeavors.

2. BACKGROUND OVERVIEW

2.1. Data Augmentation

Data augmentation (DA) is a commonly employed method in machine learning (ML) to augment the size and diversity of a training dataset [15]. General methods for DA making changes or adjustments to the existing data, generating new samples that resemble original samples but are not exact replicas of the original samples. DA has proven to be an effective method for improving the performance and robustness of ML models [19], Particularly in scenarios where there is a shortage or imbalance in the training data.

Different ML tasks require tailored DA techniques to optimize model performance. For example, in computer vision tasks like image classification, common augmentations include flipping, rotating, and changing colors [20] along with deep learning approaches such as Adversarial Training [21] and Neural Style Transfer to create variations in the training images [22]. Similarly, in NLP tasks such as topic modelling, paraphrasing and word dropout can be used to augment textual data, improving the model's ability to understand language nuances. Likewise, in speech recognition tasks, augmentations like pitch shifting and introducing noise can enhance the model's robustness against variations in speech patterns and background noise [23-25]. Adapting augmentation methods to specific tasks is crucial for enhancing model performance and adaptability across different real-world scenarios.

2.2. Textual Data Augmentation

Recent years have seen notable progress in Natural Language Processing (NLP), propelled by the adoption of Machine Learning (ML) techniques and increased computational capacity. Given the data-intensive nature of these ML approaches, the significance of textual Data Augmentation (DA) has grown, as it produces varied and high-caliber data samples crucial for training and evaluating ML models. In this section, fundamental textual DA techniques that have been effective in improving the performance and generalization capabilities of NLP models will be explored.

By introducing variations in sentence structure, semantics, and syntax, textual DA techniques unlock new possibilities for tackling challenges such as data scarcity, domain

adaptation, and model robustness. Textual DA methods can be grouped into 3 categories based on the underlying techniques as proposed in [26]. These categories are Noising-based DA, Paraphrasing-based DA and Sampling-based DA. Each technique offers a unique approach to augmenting textual data, providing researchers and practitioners with valuable tools to tackle real-world language processing tasks.

Figure 2.1 illustrates the basics of each DA technique. Here, paraphrasing-based DA is demonstrated with a substitution with hyponym; noising-based DA is demonstrated with random insertion of plural form and random deletion and sampling-based DA is demonstrated with a new sentence which created by sampling new attributes obtained from the dataset.

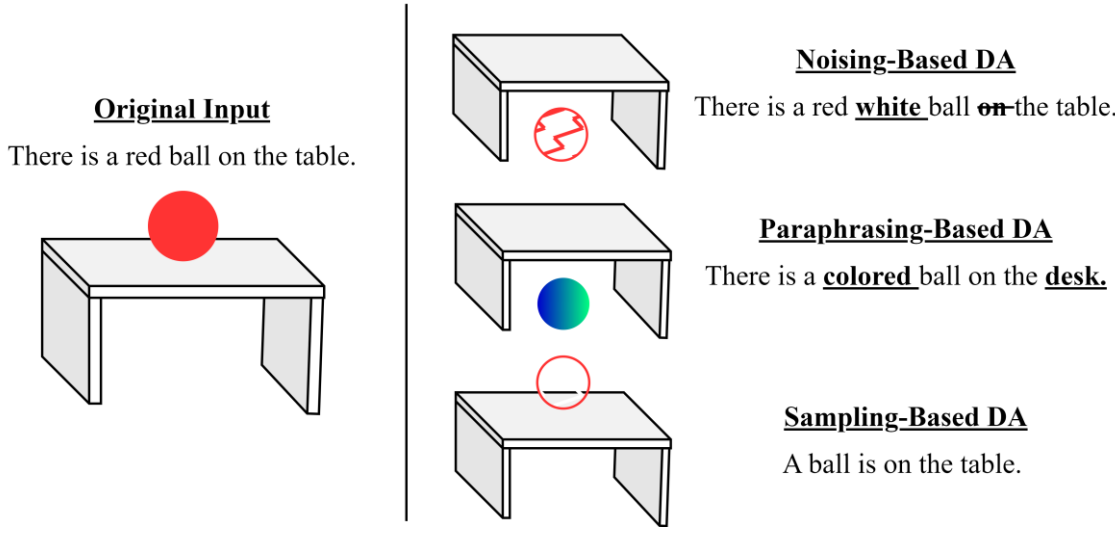


Figure 2.1. Example of paraphrasing-based, noising-based and sampling-based textual DA methods, adapted from [26].

2.2.1. Noise-Based DA Methods

Noise-based DA serves as a pivotal technique in NLP, aiming to simulate real-world complexities by introducing random variations into the original text [26]. Operating across character, sentence, and word levels, this method enriches the dataset, however, it does not ensure semantic coherence of the textual data unlike paraphrasing-based or sampling-based DA. The idea behind adding noise to textual data is to introduce new challenges to the DL models which cannot be found in natural text data.

At the character level noise, individual characters are manipulated through processes like random insertion [27]. At word level, random swapping, random insertion, random substitution, random deletion [28] of words are generally used to introduce noise to text by modifying tokens in sentences without affecting the structure of the sentence. Sentence level noising focuses on perturbing entire sentences to diversify structures and wording like combining [29] or sentence shuffling [30]. By exposing NLP models to diverse linguistic expressions encountered in real-world scenarios, noise-based augmentation enhances their robustness and generalization capabilities. In Figure 2.2, examples for noising-based DA methods are given.

DA Method	Examples	
	Original	Augmented
Random Swapping	The children played joyfully in the park	The children park joyfully in the played
Random Deletion	There is a tree behind them	There is a tree behind them
Random Insertion	Leaves are slowly falling	Leaves petals are slowly falling

Figure 2.2. Example of noising-based DA methods, adapted from [26].

2.2.2. Paraphrasing-Based DA Methods

Paraphrasing-based DA involves altering the initial sentence to create new sentences while preserving the original meaning, despite changes in structure and wording [26]. These methods operate across various levels including lexical, phrase, and sentence paraphrasing. Thesauruses like WordNet offer a straightforward approach by replacing words with synonyms or hypernyms to maintain semantic integrity [28].

Semantic embeddings offer an advancement over thesaurus-based methods by using pre-trained word embeddings to replace words with their closest neighbors in embedding space [31]. Language models, particularly pretrained models like BERT, introduce a context-aware approach to paraphrasing. The use of conditional BERT for sentence augmentation, masking and predicting words based on context is demonstrated in [32].

Another approach is Backtranslation [33], which employs Sequence-to-Sequence (Seq2Seq) language models. This method translates the original sentence into different target languages before translating it back to the source language. While the translated data may not precisely match the original, it maintains semantic similarity.

In Figure 2.3, examples from selected paraphrasing-based DA methods are given. Here, rule-based method paraphrases the sentence using predefined regular expression [34]. Another example given in Figure 2.3 is Backtranslation [33] where paraphrasing is done through translation. The last example utilizes word embeddings for paraphrasing [35].

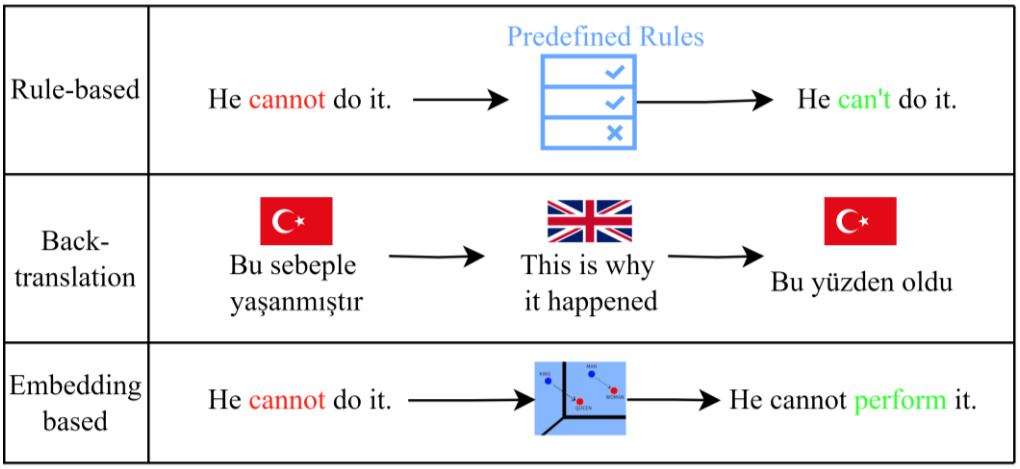


Figure 2.3. Example of paraphrasing-based DA methods, adapted from [26].

2.2.3. Sampling-Based DA Methods

Sampling-based DA entails the generation of novel instances from an existing corpus using various sampling techniques [26]. Unlike methodologies that might focus on individual instances during generation, this approach considers the entire corpus as a basis. Consequently, the resulting text manifests as a novel variant while upholding patterns consistent with the original corpus. Various sampling strategies, such as random sampling, stratified sampling, or oversampling of underrepresented classes, can be employed in this technique, tailored to the specific needs of the DA task. By leveraging sampling-based DA, the augmented dataset encompasses a wider array of variations, thereby increasing model robustness and generalization capabilities. Moreover, this technique proves particularly advantageous in scenarios characterized by limited original

dataset sizes, enabling the creation of additional training instances without necessitating manual data collection or annotation efforts.

2.3. Text Classification

Text classification in the field of NLP pertains to the fundamental task of assigning labels to text documents, playing a pivotal role in numerous applications ranging from sentiment analysis to document classification [16]. Over time, many methods have been devised to tackle the intricacies of text categorization. Traditionally, ML algorithms such as k-Nearest Neighbors, and decision trees were prominently employed for constructing classification models [16], where numerical representation methods were utilized to encode textual information. Despite their historical effectiveness, these conventional methods exhibit limitations in capturing semantic nuances at both the sentence and document levels, rendering them inadequate for addressing more intricate text classification tasks such as intent classification and irony detection [17]. Therefore, Deep learning models, such as recurrent neural networks (RNNs) and transformers [36-38], have emerged as powerful tools for text classification, leveraging their ability to learn intricate patterns and capture semantic relationships across large datasets [17].

Sentiment classification, a subtask of text classification within NLP, involves the automatic identification and categorization of the emotional tone or sentiment expressed within textual content [39]. Sentiment classification holds particular significance in applications such as customer feedback analysis and social media monitoring. Traditional machine learning methods have been widely employed for sentiment classification, often utilizing features like word embeddings and lexicon-based approaches [40]. However, deep learning techniques have revolutionized sentiment classification in recent years, offering enhanced capabilities in capturing contextual information and nuances of sentiment expression [18], as explained for text classification.

2.4. Autoencoders

An autoencoder represents a neural network structure extensively employed in unsupervised learning and dimensionality reduction endeavors [41]. Comprising two primary components, namely the encoder and decoder, autoencoders are trained to

reconstruct input data. The encoder condenses the input into a latent representation, while the decoder strives to reconstruct the initial input from this representation. Through the process of learning to reconstruct the input, the autoencoder effectively acquires the capability to generate a condensed representation of the data [42].

The autoencoder architecture consists of three primary components: an encoder, a bottleneck, and a decoder. Both the encoder and decoder may be constructed with multiple layers, such as convolutional layers or LSTM layers. The main goal of training an autoencoder is to minimize the reconstruction error, usually assessed through a loss function [43]. The autoencoder learns to encapsulate the most significant characteristics of the data in the latent representation by reducing the discrepancy between the input and output.

Autoencoders find applications across various domains. They are used for dimensionality reduction [44], where they learn condensed representations of data with high dimensions, which can be beneficial for tasks like feature extraction and data visualization. Additionally, autoencoders are employed in anomaly detection [45], where they learn the normal patterns of a dataset to detect anomalies or outliers. Variants of autoencoders [46], such as variational autoencoders (VAEs) [47] and generative adversarial networks (GANs) [48], are used for generative modeling. In Figure 2.4, a basic representation of architecture of a typical autoencoder is shown.

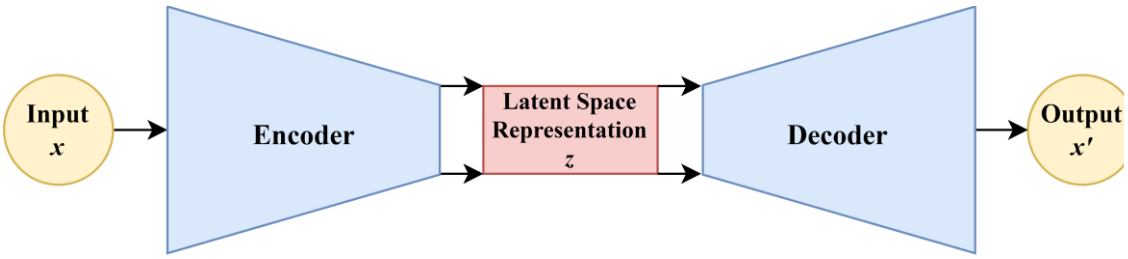


Figure 2.4. A basic representation of architecture of a typical autoencoder.

3. RELATED WORK ON DATA AUGMENTATION

As stated in Section 2, textual DA techniques have been categorized into three categories which are Noising-based DA, Paraphrasing-based DA and Sampling-based. Thus, previous works in literature are analyzed under these categories.

3.1. Noising-based DA Methods

The approach of noise-induced DA aims to enrich datasets by introducing random perturbations into textual data. One prominent application in the realm of textual DA is Easy Data Augmentation (EDA) [28] which combined various rule-based transformations to enhance the text classification performance of ML models. These methods include both noising-based and paraphrasing-based DA methods, so EDA could be considered under both categories. DA techniques employed in the study by Wei and Zou [28] unfolds as follows:

- Synonym Replacement: This method involves selecting a specified quantity of non-stop words from the sentence and then substituting them with equivalent synonyms.
- Random Insertion: This approach involves randomly selecting a non-stop word from the sentence, then inserting a random synonym of that word into a randomly chosen position within the sentence.
- Random Swap: This operation entails randomly selecting two words from the sentence and then exchanging their positions.
- Random Deletion: Words from the sentence are subject to random deletion with a certain probability p.

In the study conducted by [27], an innovative technique known as An Easier Data Augmentation (AEDA) aimed at augmenting textual data solely through the insertion of random punctuation marks was introduced. This operation involves determining the quantity of punctuation for each sentence, selecting insertion points, and punctuation randomly. In comparison with the widely used method of EDA [28], AEDA offers a simpler approach and avoids the loss of information associated with random deletion operations inherent in EDA. Empirical evaluations indicate that AEDA consistently outperforms EDA across various datasets in the context of text classification tasks [27].

3.2. Paraphrasing-based DA Methods

Paraphrasing-based DA methods aim to generate new samples based on original samples, while preserving the semantic coherence of original samples. A fine example of paraphrasing-based DA is proposed in [49] where the data is augmented via cropping and rotating the sentence with the help of dependency parsing [50]. While performing rotations and crops on sentences, the dependency parser algorithm determines the interrelated words and ensures the connectivity of such words is preserved throughout the transformational process. Furthermore, the parser identifies the essential components of the sentence and refrains from cropping them out of context. This method requires a dependency parser to work with any language. In experiments, this method is applied for different low-resource languages such as Lithuanian, Turkish for PoS-tagging task. The results showed that both cropping and rotating are valid DA methods that increase the accuracy of PoS-taggers. Figure 3.1 shows an example of methods proposed in [49].

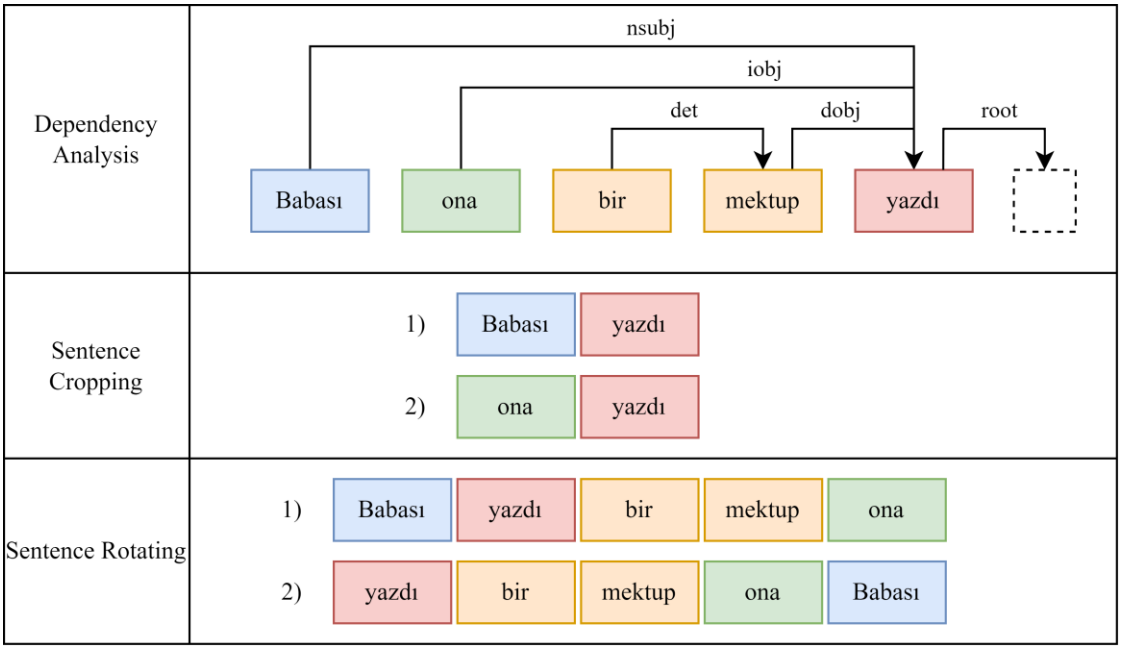


Figure 3.1. Example crop and rotation transformation proposed in [49].

Some of the studies in the literature propose precise augmentation techniques for more specific tasks to improve the performance gain attained from the augmentation process. In [51], a DA pipeline specifically tailored for aspect-based sentiment analysis was

proposed. Aspect-based sentiment analysis is used to determine sentiment orientation towards an entity or aspect within the text. The research in this area has accelerated since the arrival of advanced DL models since DL models can encode aspect-based sentiment information without manually extracted features unlike previous ML algorithms. DA techniques used in [51] were:

- Part-of-speech (PoS) wise synonym substitution (PWSS): After PoS-tagging the original sentence, synonyms with the same PoS-tags are identified. Then, the synonym with the highest semantic similarity score to the original word is selected and replaced within the sentence. An example of PWSS is given in Figure 3.2. In Figure 3.2, all words except for "quality" are candidates for synonym substitution. That is because "quality" is considered as an aspect term for this sentence.

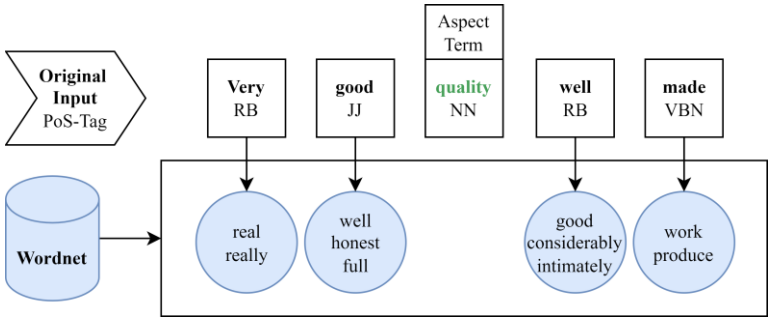


Figure 3.2. PWSS example, adapted from [51].

- Dependency relation-based word swap (DRAWS): A dependency syntax tree, representing a sentence's structure with a root and dependency arcs between words, is utilized. This tree serves as the basis for a DA method where synonym words with the same arc to the root node are exchanged between sentences to generate new sentences. An example of this DRAWS is given in Figure 3.3.

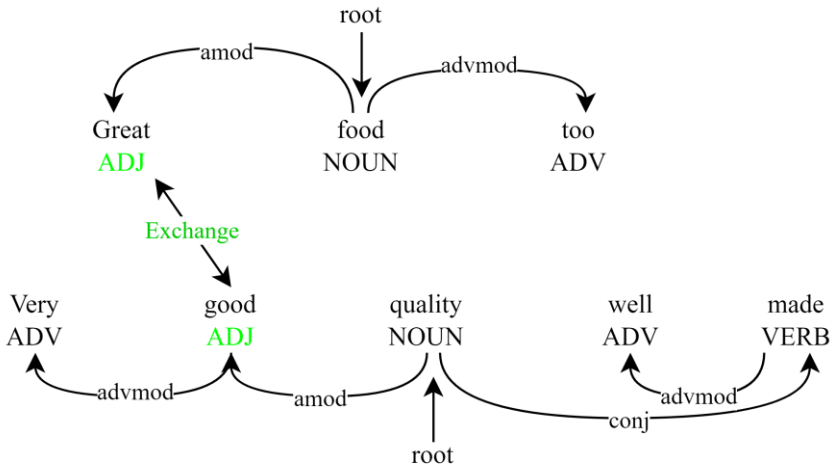


Figure 3.3. DRAWS example, adapted from [51].

3.3. Sampling-based DA Methods

Since large language models have been successful at understanding natural language and generating text within the boundaries of natural language, they are well-suited for generating new data samples upon original data samples. There have been numerous works that utilizes LLM's for DA [4].

In [52], a novel approach for leveraging prompt-based Large Language Model (LLM) architectures for DA was introduced. Prior to this investigation, existing studies utilized LLMs fine-tuned solely on text data, demonstrating their effectiveness. However, this study extended the scope by incorporating additional contextual information in the form of class labels, thereby enhancing the conditionality of the DA process. Specifically, employing a structured prompt format comprising text tags, class tags, and corresponding class labels for three distinct LLM architectures, namely the Bidirectional Encoder Representations from Transformers (BERT) [53], the Generative Pre-trained Transformer 2 (GPT-2) [54] and the Bidirectional and Auto-Regressive Transformers (BART) [55] were fine-tuned for DA. Experimental evaluations conducted for text classification tasks revealed superior performance of models trained with datasets tailored for BART. Although BERT exhibits comparable results to BART, it was observed that the GPT-2 model fails to effectively retain the target class information embedded within the text prompts, resulting in suboptimal outcomes.

In [56], a novel method employing GPT-3 for Data Augmentation (DA) was introduced. The authors critiqued the prompt-based approach, highlighting limitations such as the constraint on the size of in-context augmentation and challenges posed by real-world issues like memory constraints. This study addressed these concerns by fine-tuning GPT-3 using a structured prompt format, wherein the job description was presented initially, followed by the text and target classes. This structured prompt guided GPT-3 to generate text in a format that includes both the text itself and its associated probability of belonging to a specific class. This inclusion of class probabilities proved advantageous for DA, as it enables the filtering of generated data samples that may not confidently belong to a particular class based on their probabilities.

In [57], a pioneering approach to DA for NLP was presented, where a versatile language model was trained to satisfy the DA requirements across various NLP tasks, ranging from text classification to text generation. This approach introduced the Knowledge Mixture Training (KoMT) strategy, designed to train a pre-existing encoder-decoder generative language model named KnowDA. The KoMT strategy was structured around four fundamental components. Firstly, a diverse collection of datasets spanning a broad spectrum of NLP tasks was assembled, facilitating the training of the multi-task DA model, KnowDA. Secondly, a standardized format was devised for all tasks, adopting a key-value structure that encompasses features, feature descriptions, and actual samples. This format ensures uniformity across tasks. Thirdly, the training process involved denoising objectives, wherein key-value pairs within each sample were randomly masked to enhance the robustness of KnowDA. The primary aim of KnowDA was to predict these masked fields, thereby improving its ability to generate coherent outputs. Lastly, to address challenges stemming from rare or unseen NLP tasks during training, a demonstration component is integrated into the samples. Leveraging its pretraining and instruction-following capabilities, KnowDA adapted to previously unseen NLP tasks through demonstration exemplars. In Figure 3.4, an example of KoMT is given.

Figure 3.4. An example of KoMT method, adapted from [57].

3.4. Autoencoders for Data Augmentation

Autoencoders are used for data augmentation on a variety of data types such as image, text, and tabular data. In [58], Self-Supervised Manifold Based Data Augmentation (SSMBA) was introduced which resembles Denoising Autoencoders [59] by corrupting the input by a predefined corruption function and then reconstructing the original input from corrupted input using a reconstruction function. The motivation behind SSMBA was to create a DA model that could work with any supervised NLP task regardless of domain.

In [60], the effectiveness of sparse, undercomplete, deep, and variational autoencoders for augmenting and generating synthetic data, focusing on financial datasets was explored. Autoencoder augmentation significantly enhances predictive performance, with a notable average model score improvement. Variational autoencoders notably capture non-linear correlations more effectively.

In [61], DA for binary text classification is done through using Variational Autoencoder (VAE) as a generative model. In their approach, they train a VAE with original dataset and use the decoder of VAE with random sampling on latent vector of VAE. To provide conditionality, they trained one VAE per class which makes the output of DA specific to each class.

In [62], Variational Hierarchical Dialog Autoencoder (VHDA), which is tailored to capture linguistic features and structured annotations, was introduced. By leveraging interconnected latent variables, VHDA produces cohesive purposeful dialogs while

addressing training intricacies linked with variational models. Empirical findings across various dialog datasets highlight VHDA's effectiveness in enhancing subsequent dialog trackers through generative data augmentation. Furthermore, our integrated methodology surpasses prior benchmarks in tasks related to dialog response generation and user simulation.

4. PROPOSED DATA AUGMENTATION METHOD

This thesis investigates the efficacy of data augmentation techniques for enhancing natural language processing tasks. The proposed methods involve a series of steps: initial acquisition of a dataset, optional preprocessing of the text, selection of tokenization methods including Bag-of-Words or Word2Vec, followed by the training of various autoencoder (AE) types such as Traditional AE, Variational AE (VAE), Adversarial AE (AAE) and Denoising Adversarial AE (DAAE). All autoencoders utilize LSTM networks for both encoder and decoders. Subsequently, synthetic samples are generated by reconstructing original samples using the trained AEs, with the flexibility to apply different augmentation ratios using the proposed augmentation strategy. Optionally, a filtering pipeline may be employed to refine the generated samples. Figure 4.1 shows the general overview of the proposed methods in this thesis.

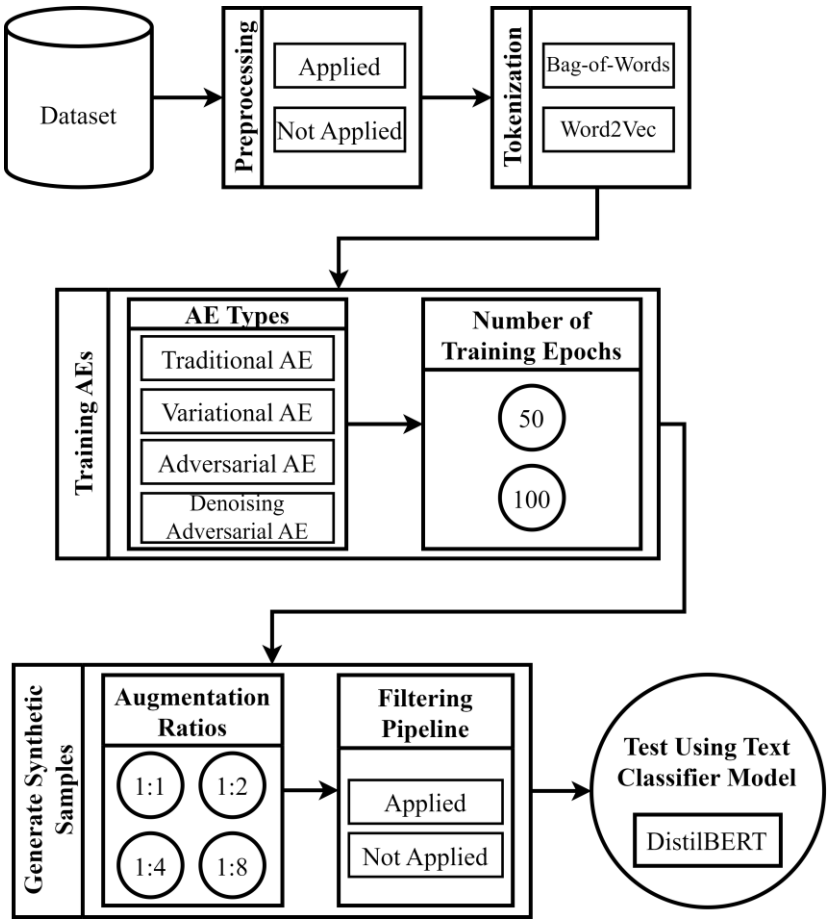


Figure 4.1. General overview of the proposed method in this thesis.

4.1. Tokenization and Vectorization

Tokenization is the process of dividing text into its atomic parts, usually words or subwords, in order to make textual data processable by machine learning models [63]. With the help of tokenization, text data is converted into numerical representations that ML algorithms can understand. There are various techniques that involve tokenization, such as n-gram tokenization [64] and tokenization using pre-trained transformer models [65]. These techniques ensure that textual data is broken down into manageable units, enabling the application of further text processing and analysis methods.

Vectorization refers to the methods that transform tokenized text into numerical vectors, which can be used as input features for machine learning models. Common vectorization techniques include Bag-of-Words (BoW) [66] and Word2Vec word embeddings [67]. Bag-of-Words creates vectors based on word frequencies within a text, while Word2Vec generates dense word embeddings that capture semantic relationships between words. In this study, both Bag-of-Words and Word2Vec were employed to represent text data, allowing for a comprehensive analysis of the textual information.

4.1.1. Bag-of-Words (BoW)

Bag-of-Words is a fundamental method for vectorization in NLP [68]. In Bag-of-Words, sentence level embeddings are created based on frequencies of words. Conceptually, the corpus containing all documents is considered as a “bag” where words are kept without any word order or grammar rule.

Bag-of-Words method begins with vocabulary creation from the corpus [68]:

- Let $D = \{d_1, d_2, \ldots, d_N\}$ be the corpus which contains N documents, where d represents document from the dataset.
- Create Vocabulary $V = \{w_1, w_2, \ldots, w_M\}$ where M is the total unique words count in the dataset and w represents a word from the dataset.

In the next step where documents are vectorized [68]:

- Each document d_i from the dataset is depicted in the form of a one-dimensional vector x_i with M elements.

- The j-th element of x_i, which is denoted as x_{ij}, shows the frequency of the word w_j in d_i. So, d_i can be expressed as $x_i = [x_{i1}, x_{i2}, \dots, x_{iM}]$ where $x_{ij} = count(w_j, d_i)$ which denotes how many occurrences of the word w_j in d_i.

An example of the creation of BoW vector representation is shown in Figure 4.2.

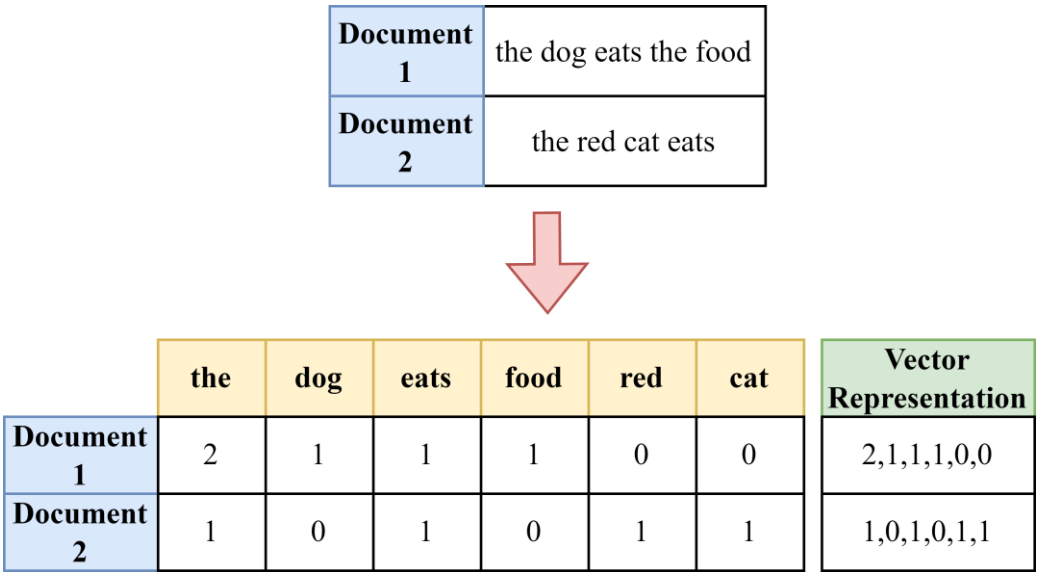


Figure 4.2. An example of BoW vector representation.

4.1.2. Word2Vec Word Embeddings (w2v)

Word2Vec is a well-known method when generating vector representations of words, developed by Google [69]. Two main algorithms play a role in Word2Vec which are Continuous Bag-of-Words (CBOW) and Continuous Skip-Gram.

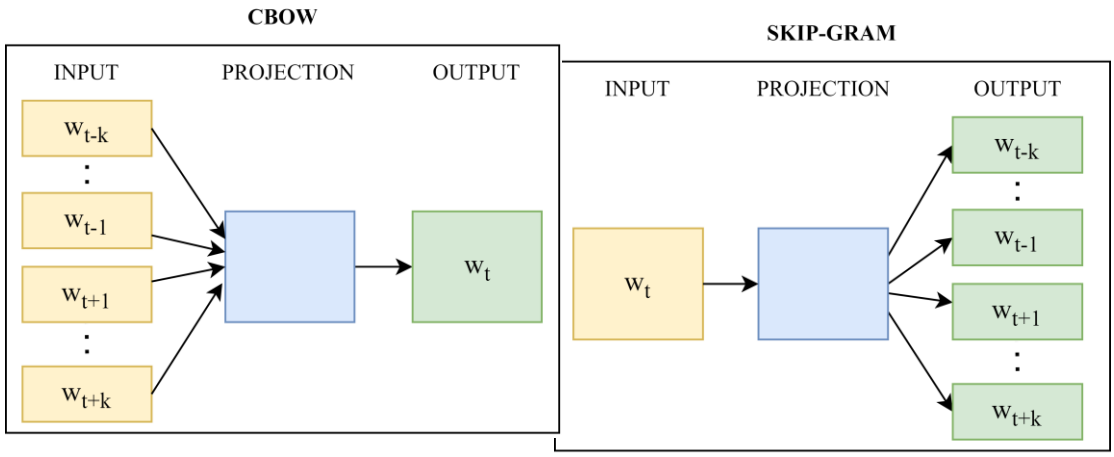
CBOW [67] is used to forecast the target word by analyzing its neighboring words. Mathematically, w_t is predicted using the list of its surrounding context words where k is window size. A mathematical representation of this process is given in Eq. 4.1 [67] for a vocabulary with size T.

$$\frac{1}{T}\sum_{t=k+1}^{T-k} \log P\left(w_t \mid w_{t-k}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+k}\right) \tag{4.1}$$

The Skip-Gram Model [70] is employed to forecast the context words provided a target word. Consequently, the objective of the Skip-Gram Model is to maximize the log probability of the context words given the target word, as illustrated in Eq. 4.2 [70].

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-k\leq j\leq k, j\neq 0}\log P(w_{t+j}|w_t) \tag{4.2}$$

Figure 4.3 shows block diagrams explaining how CBOW and Skip-Gram models work.



(a) (b)

Figure 4.3. Block diagrams for (a) CBOW model and (b) Skip-gram model.

In this thesis, a pretrained Word2Vec model [71] is used by restricting the vocabulary size to 100,000 words. Embeddings from Pretrained Language Models like BERT Embeddings [53] are not considered, because the proposed method is meant to be less dependent on language models.

4.2. Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNNs) [41] are a class of neural networks designed to process sequential data by retaining hidden state (h_t) information over multiple time steps, allowing them to capture temporal dependencies. A graphical representation of RNNs is given in Figure 4.4. Nevertheless, classic RNNs face the vanishing gradient

problem, causing gradients to decrease exponentially as they move back in time, limiting the network's capacity to learn long-term relationships. [72].

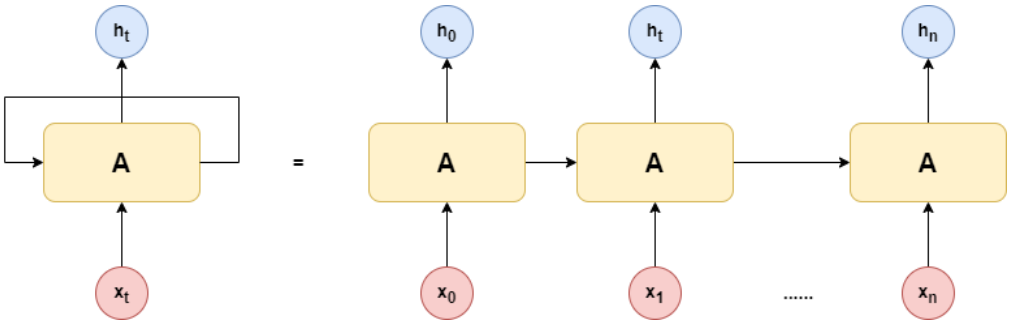


Figure 4.4. Architecture of an RNN model [73].

To address this limitation, Long Short-Term Memory (LSTM) networks [3] were introduced. LSTMs are a specialized form of RNNs that include gated mechanisms to control the flow of information through memory cells, effectively mitigating the vanishing gradient problem. By using input, forget, and output gates, LSTM networks can selectively update and retain relevant information over extended sequences. This capability allows LSTM networks to capture long-range dependencies more effectively than RNNs. This architectural improvement renders LSTMs particularly effective for tasks that involve modeling sequential data with complex temporal dynamics.

LSTM networks are characterized by their unique architecture, which consists of memory cells and gating mechanisms. Figure 4.5 depicts the structure of a single LSTM unit. The memory cell is shown as a horizontal line running through the unit, symbolizing the flow of information over time.

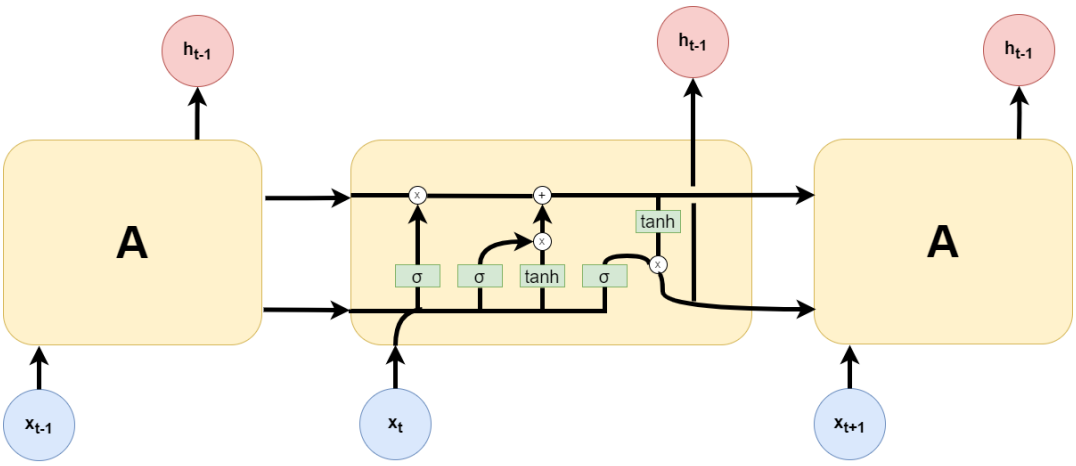


Figure 4.5. Architecture of an LSTM model [73].

The input gate is responsible for the information entering the memory cell, the forget gate is responsible for the information exiting the memory cell, and the output gate manages the information remaining within the memory cell.

The input gate in an LSTM network regulates the flow of new information that is incorporated into the memory cell. At each time step, the input gate computes a sigmoid activation function over the input x_t and the previous hidden state h_{t-1}. This activation determines which information from the current input and the previous hidden state is pertinent for updating the memory cell. The output of the input gate, which is represented as i_t, ranges between 0 and 1, with values close to 1 indicating that the corresponding input is important for updating the cell state. Mathematically, the operations performed by input gate are shown in Eq. 4.3 [74], where W represents weights and b represents biases.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{4.3}$$

The forget gate is tasked with deciding which information stored in the memory cell will be retained or discarded. Like the input gate, the forget gate computes a sigmoid activation function over the input x_t and the previous hidden state h_{t-1}. The value of f_t determines how much of the previous cell state $C_{t-1}$is maintained for the current time step. Mathematically, the operations performed by forget gate is shown in Eq. 4.4 [74], where W represents weights and b represents biases.

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (4.4)$$

In an LSTM network, the output gate regulates the passage of information from the memory cell to the current hidden state h_t, which functions as the output of the LSTM unit. The output gate applies a sigmoid activation function to both the input x_t and the previous hidden state h_{t-1}. Additionally, it calculates a hyperbolic tangent activation function over the candidate cell state $\tilde{C}_t$, which contains the information proposed for inclusion in the updated cell state C_t. The output gate activation, which is represented as o_t, decides the extent to which the updated cell state should be revealed to the rest of the network. Mathematically, the operations performed by output gate are shown in Eq. 4.5 [74], where W represents weights, b represents biases and σ represents sigmoid activation function.

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (4.5)$$

The cell state is the core component of an LSTM unit, responsible for retaining and updating information over time. At each time step, the cell state is updated based on the inputs from the input and forget gates. The updated cell state, indicated as C_t, results from a fusion of the previous cell state C_{t-1} and the new information suggested by the input gate. It acts as a long-term memory storage unit, enabling the LSTM network to capture dependencies across extended sequences of data. Mathematically, the operations performed to update cell state is shown in Eq. 4.6 and Eq. 4.7 [74], where W represents weights, b represents biases and $tanh$ represents hyperbolic tangent activation function. In Eq 4.7, $\odot$ represents element-wise multiplication.

$$\tilde{C}_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (4.6)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (4.7)$$

The hidden state within an LSTM network embodies the network's memory at a specific time step, containing the information distilled from the input sequence up to that point. It serves as the LSTM unit's output and acts as a condensed representation of the input sequence. The formula to calculate the hidden state h_t at time step t is represented mathematically in Equation 4.8 [74].

$$h_t = o_t \odot \tanh(C_t) \tag{4.8}$$

The hidden state captures dependencies for short-term as well as long-term in the input sequence, allowing the network to make predictions or perform tasks based on the context accumulated over multiple time steps. The hidden state is frequently employed as the input to subsequent layers in the neural network or as the final output for various tasks such as sequence prediction, classification, or generation. By updating the hidden state iteratively across time steps, an LSTM network can effectively capture complex patterns and relationships in sequential data.

In this thesis, LSTM networks serve as pivotal components within the autoencoder models, facilitating the learning and preservation of linguistic patterns inherent in the dataset which is a fundamental aspect of data augmentation tasks. By employing LSTM networks with 1024 hidden layer units for both encoders and decoders, the models are equipped to comprehend and encode intricate linguistic structures. The chosen sequence lengths of 512 for the encoder's LSTM and 128 for the decoder's LSTM are tailored to enhance the networks' capability to accurately capture and replicate these patterns. This strategic integration of LSTM networks underscores their crucial role in enabling the autoencoder models to effectively learn and replicate linguistic nuances, thereby enhancing the richness and diversity of the augmented dataset.

4.3. Autoencoders

An autoencoder is a class of unsupervised neural network which is designed primarily to replicate the input as accurately as possible [75]. In the middle of an autoencoder, there is a unique architectural feature known as the bottleneck layer. Contrary to conventional network architectures, where the number of neurons typically increases through successive layers, the bottleneck layer imposes a restriction by decreasing the dimensionality of the input data. This reduction creates a compression effect, compelling the model to capture the most useful features while discarding redundant or less informative aspects of the input.

Autoencoders comprise two components: the encoder and decoder. In the flow of an autoencoder, first, a representation of input data in a lower-dimensional vector space is created by the encoder block. The resulting vector is called the latent vector, which encapsulates the crucial features of the data. After creating this vector, the vector is taken by the decoder block and used to reconstruct the input given to the encoder. Formally, let x denote the input data, z represent the latent vector, and $\hat{x}$ denote the reconstructed output. The encoder f_{enc} maps the input x to the latent vector z, and the decoder f_{dec} attempts to reconstruct the original data from z. Mathematically, this process can be expressed as in Eq. 4.9 and Eq. 4.10:

$$z = f_{enc}(x) \tag{4.9}$$

$$\hat{x} = f_{dec}(z) \tag{4.10}$$

A schematic diagram of the autoencoder used in this thesis is shown in Figure 4.6.

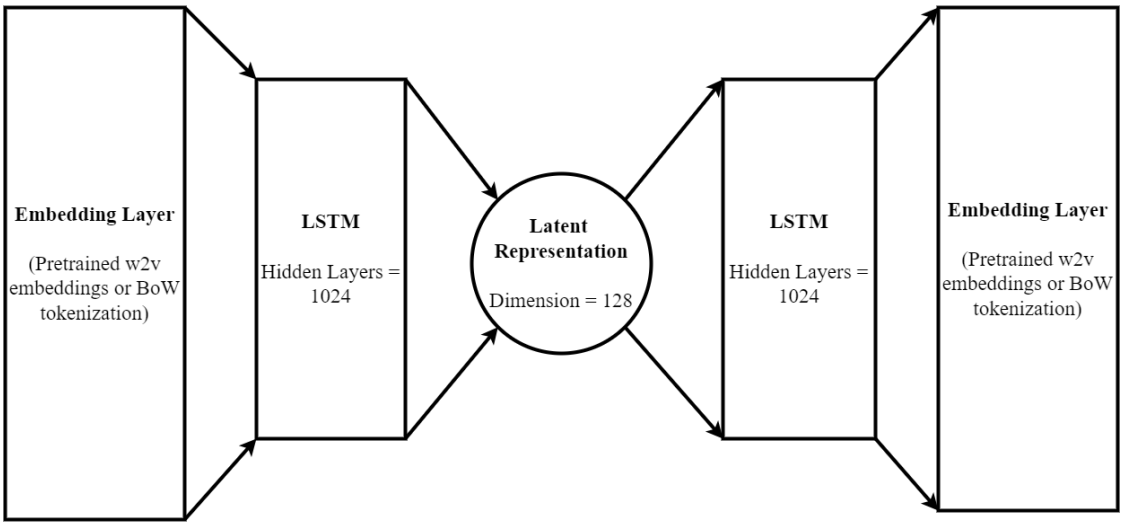


Figure 4.6. A diagram of autoencoder used in this thesis.

4.3.1. Variational Autoencoders (VAE)

Variational Autoencoders (VAEs) [47] represent a sophisticated advancement of conventional autoencoders, leveraging a probabilistic framework to enhance their DA capabilities. VAEs depart from deterministic representations by encoding input data into probability distributions within the latent space. Mathematically, given an input x, a VAE encoder $q(z|x)$ approximates the posterior distribution $p(z|x)$ over latent variables z, which captures the underlying structure of the data. This probabilistic perspective enables VAEs to model the inherent uncertainty in the data, offering a more nuanced and adaptable representation.

The training of VAEs revolves around optimizing two key objectives: (1) a reconstruction loss, L_{rec}, which ensures fidelity to the input data, and (2) a Kullback-Leibler (KL) divergence term, L_{KL}, that regularizes the latent space distribution. The overall loss function is thus formulated as in Eq. 4.11 [76]:

$$L_{VAE} = L_{rec} + \beta . L_{KL} \tag{4.11}$$

where β corresponds to a hyperparameter governing the balance between reconstruction accuracy and regularization of the latent space. This dual objective empowers VAEs to

not only produce faithful reconstructions but also to generate a variety of samples by sampling from the learned latent space distributions. Such stochasticity introduces controlled variability, rendering VAEs particularly effective for tasks requiring the generation of novel and diverse instances, such as DA. The inherent probabilistic nature of VAEs renders them a robust choice for learning rich representations and generating augmented data with meaningful diversity, thereby contributing to improved performance in downstream tasks.

In VAE, the encoder maps the input x to the parameter of a variational posterior $q(z|x)$, which is typically a gaussian distribution of mean vector $\mu(z|x)$ and a standard deviation vector $\sigma(z|x)$.

Figure 4.7 shows the VAE architecture used in this thesis. Here, σ represents standard deviation vector and μ represents mean vector.

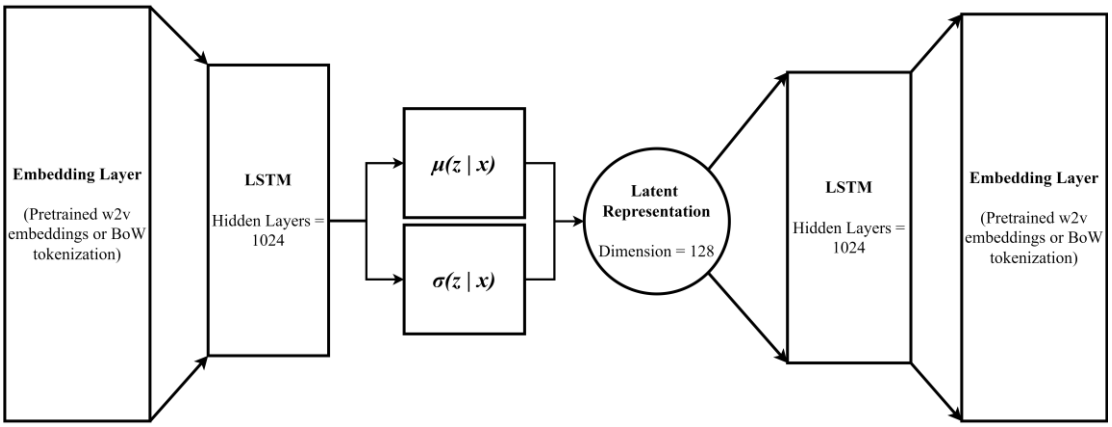


Figure 4.7. Variational Autoencoder model used in the experiments.

4.3.2. Adversarial Autoencoder (AAE)

Adversarial Autoencoders (AAEs) [77] integrate the fundamental concepts of autoencoders with the adversarial framework of generative adversarial networks (GANs). In contrast to traditional autoencoders, which solely focus on minimizing reconstruction loss, AAEs incorporate a discriminator network alongside the encoder-decoder architecture. This discriminator operates in tandem with the autoencoder, distinguishing between the encoded latent representations and a predefined prior distribution. This prior distribution is typically a normal distribution N, defined in Eq. 4.12 [77] :

$$z \sim N(\mu, \sigma^2 I) \tag{4.12}$$

where $\mu = 0$ is the mean vector and $\sigma^2 I$ is the covariance matrix with $\sigma = 1$, making the covariance matrix I the identity matrix. The adversarial training mechanism prompts the encoder to generate latent representations that align with this normal distribution, thereby improving the diversity and semantic meaningfulness of the representations. Mathematically, the objective function of AAEs combines the reconstruction loss term (L_{rec}) with the adversarial loss term (L_{adv}), defined as in Eq. 4.13 [77]:

$$L_{AAE} = L_{rec} + \lambda . L_{adv} \tag{4.13}$$

where λ controls the balance between reconstruction fidelity and adversarial training. This adversarial component facilitates the discovery of more diverse and semantically meaningful latent representations, thereby enhancing both data generation and representation learning capabilities of AAEs.

The motivation for using AAEs in data augmentation stems from the need to address the lack of variability and overfitting observed with traditional autoencoders. By leveraging the adversarial framework, AAEs achieve more diverse and robust augmented data through a unique combination of reconstruction loss and adversarial training. The adversarial component ensures that the latent representations generated by the encoder align with a predefined normal distribution, promoting diversity and preventing overfitting. This alignment forces the model to explore a broader range of semantic variations, resulting in more varied and meaningful augmented data. Consequently, this enhanced diversity in the latent space translates to more effective data augmentation, significantly improving the generalization performance of text classification models.

Figure 4.8 shows the AAE architecture used in this thesis.

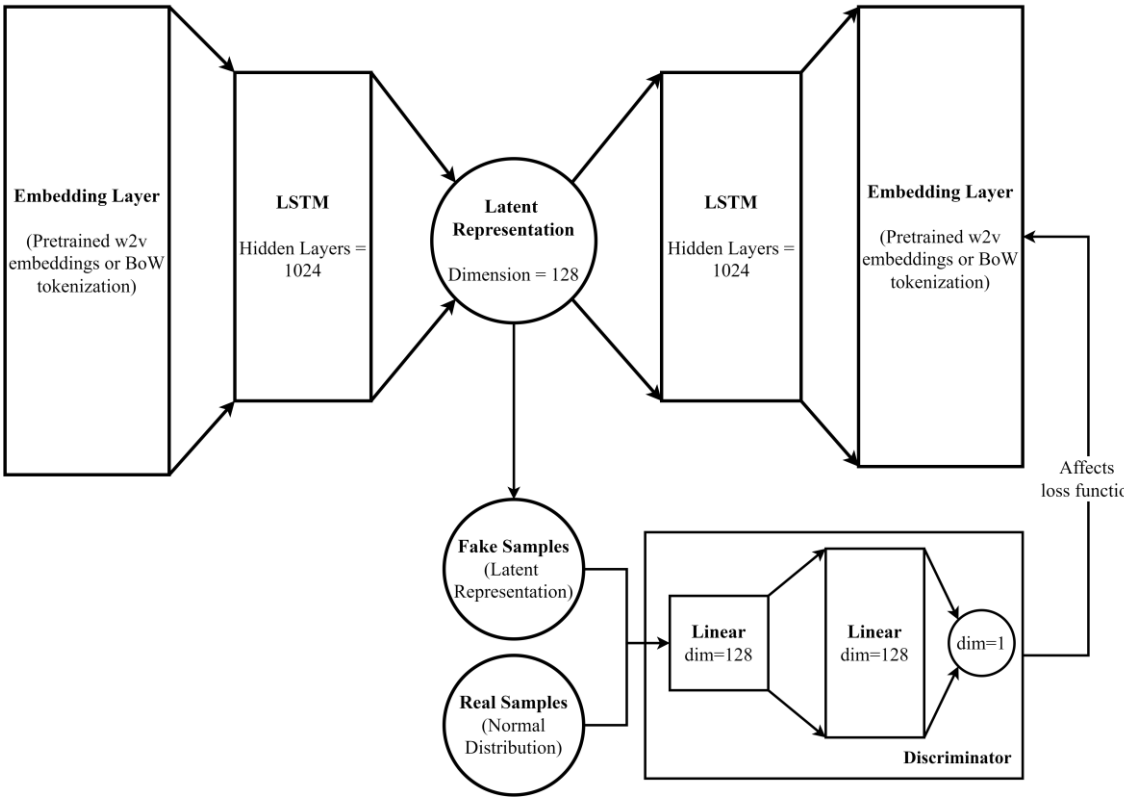


Figure 4.8. Adversarial Autoencoder model used in the experiments.

4.3.3. Denoising Adversarial Autoencoders (DAAE)

Denoising Adversarial Autoencoders (DAAEs) [78] amalgamate the denoising mechanism inherent in Denoising Autoencoders (DAEs) [59] with the adversarial training paradigm of Adversarial Autoencoders (AAEs). The integration of these components equips DAAEs with the ability to reconstruct original sentences from perturbed versions, thereby enhancing the robustness of the learned latent representations. This denoising process serves to refine the geometry of the latent space, ensuring that semantically similar texts correspond to proximate latent representations. Mathematically, the objective function of DAAEs combines the reconstruction loss L_{rec} with adversarial loss L_{adv} as shown in Eq. 4.12. Figure 4.9 shows DAAE architecture used in this thesis.

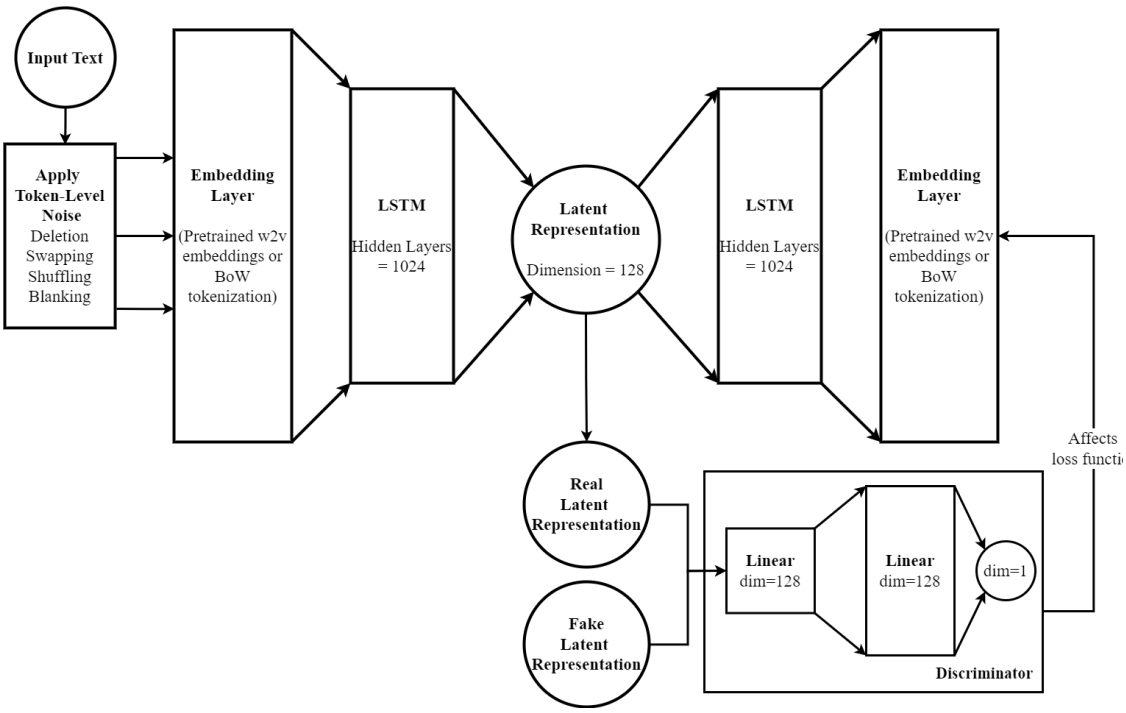


Figure 4.9. Denoising Adversarial Autoencoder model used in the experiments.

4.3.4. Autoencoder's Utilization for DA

In this thesis, AEs are employed as an approach for data augmentation for textual data. The primary objective is to explore capabilities AE's in generating synthetic textual data based on only original data samples without using any other resource. By generating such synthetic data, it is aimed to enhance robustness and generalization of text classification models.

The autoencoder architecture comprised an encoder and a decoder, each made up of LSTM networks with 1024 hidden layer units. The encoder's role was to acquire a condensed representation of the input text data, while the decoder's objective was to reconstruct the original input from this latent space. However, by restricting the dimension of latent space to 128, autoencoders are aimed to have some losses during reconstruction process, leading to new data samples that resemble original samples.

In this study, four types of autoencoders (AEs) are employed to explore their performance and efficacy in text data augmentation. The three main variants utilized are Variational Autoencoder (VAE) [47], Adversarial Autoencoder (AAE) [77] and Denoising Adversarial Autoencoder (DAAE) [78] along with traditional AE. Each variant of AE

offers unique advantages. VAE incorporates probabilistic modeling, enabling the generation of diverse and semantically meaningful variations of input data by sampling from a learned latent space. This makes it particularly well-suited for generating novel text samples and enhancing the diversity of augmented datasets. On the other hand, AAE incorporates an adversarial component, leveraging a discriminator network to encourage the learned latent space to match a specified prior distribution Finally, DAAE is chosen for its simplicity and effectiveness in reconstructing clean input from noisy data, making it suitable for tasks where robustness to input perturbations is crucial. Within the scope of the thesis, it is aimed to gain insights into their respective strengths and limitations by comparing the performance of these different AE variants, ultimately informing the selection of the most suitable model for text data augmentation tasks.

4.4. DistilBERT

Text classification models are computational algorithms that are designed to categorize text data into predefined classes based on the content of data [79]. These models traditionally range from rule-based models to statistical approaches like Naïve Bayes or ML approaches like Support Vector Machines (SVM) [79]. Modern text classification often utilizes deep learning, particularly with neural network architectures like LSTMs and transformers such as BERT [53].

BERT is a transformative model that is designed to grasp the contextual meaning of words within a sentence, taking into account both preceding and subsequent contexts, thus making it bidirectional [53]. his is achieved through a transformer architecture utilizing self-attention mechanisms [80]. Unlike previous models that read text sequentially, BERT processes text in both directions simultaneously, capturing richer contextual information. Having been pre-trained on extensive volumes of text data, BERT can be further refined for particular tasks, such as text classification.

When evaluating DA methods, it is essential to use an unbiased classifier model. Throughout this thesis, all experiments utilize DistilBERT (a distilled version of BERT) [81]. DistilBERT leverages knowledge distillation [82], where it learns from a larger 'teacher' model, which is BERT [53] for DistilBERT, to replicate its behavior, retaining around 97% of BERT's language understanding while being 40% smaller and 60% faster [81]. This model is particularly beneficial in real-time applications or environments with

constrained computational resources. Like BERT, DistilBERT is pretrained on extensive text data, enabling it to deliver strong performance even when fine-tuned on smaller datasets. Figure 4.10 shows the knowledge distillation process of DistilBERT.

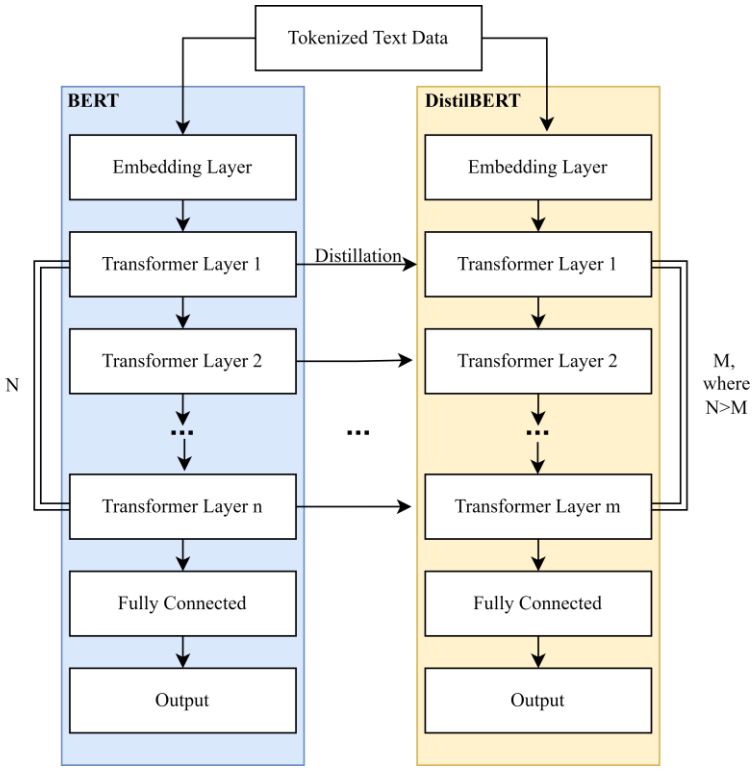


Figure 4.10. Knowledge distillation on BERT and DistilBERT.

4.5. Data Augmentation Pipeline

In typical applications, autoencoders are principally employed to accurately reconstruct input data while simultaneously reducing the dimensionality of the input space, making them suitable for compression tasks. However, in this study, autoencoders are repurposed for DA purposes where their primary objective shifts from precise input reconstruction to the generation of comparable versions of the input data. Achieving this entails a divergence from the autoencoder's conventional goal. This modification can be facilitated by constraining the dimensionality of the latent representation vector, thereby compelling the autoencoder to prioritize the generation of synthetic data that closely resembles the original input.

The methodology employed in this thesis is delineated in Figure 4.11. The initial phase of this process involves training an autoencoder (AE) with the original dataset, constituting the most resource-intensive aspect of the pipeline. During this phase, the AE tries to learn the intricacies of reconstructing the input data. Subsequently, the second phase entails employing the trained AE to reconstruct the input data, thereby generating an augmentation ratio of 1:1 relative to the original dataset. To achieve higher augmentation ratios, such as 1:2 or 1:4, successive iterations of reconstruction are performed. Specifically, to generate a 1:2 augmentation ratio, the output of the AE from the 1:1 augmentation ratio is reconstructed once more. Similarly, to attain a 1:4 augmentation ratio, the output from the 1:2 augmentation ratio is subjected to further reconstruction.

An iteration of reconstruction involves passing the output data from the previous iteration through the AE again. Each iteration effectively doubles the number of augmented samples. For example:

- Iteration 1 (1:1 augmentation ratio): The AE takes the original dataset as input and reconstructs each sample, resulting in twice the number of samples as the original dataset, including original samples.
- Iteration 2 (1:2 augmentation ratio): The AE takes the 1:1 augmented dataset as input and reconstructs each sample, resulting in three times the number of samples as the original dataset, including original samples.
- Iteration 3 (1:4 augmentation ratio): The AE takes the 1:2 augmented dataset as input and reconstructs each sample, resulting in five times the number of samples as the original dataset, including original samples.

This strategy yields promising outcomes in scenarios characterized by limited dataset sizes. However, this reconstruction pipeline may lack diversity due to the intrinsic nature of autoencoders, which primarily aim to accurately reproduce the input data without introducing substantial variations. Data augmentation examples are detailed in Appendix A.

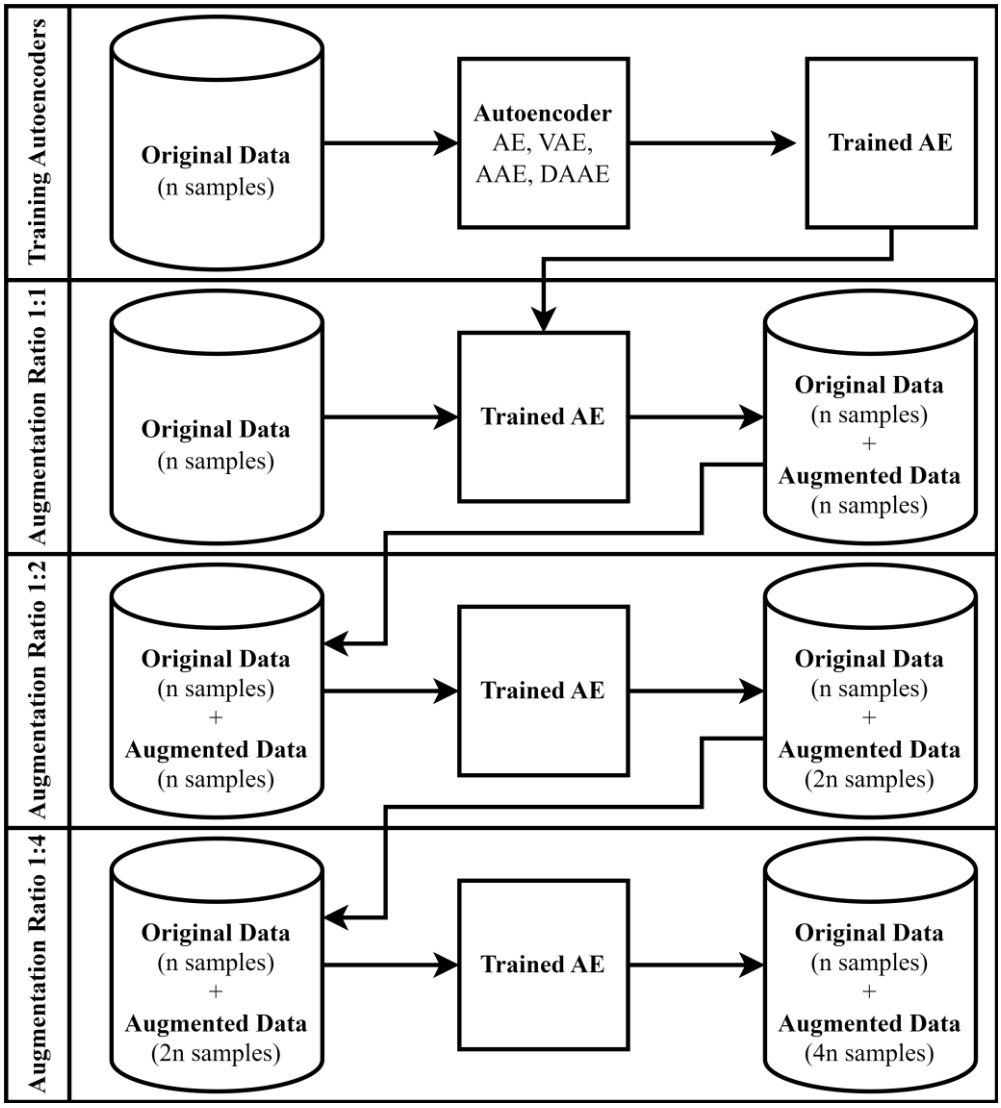


Figure 4.11. Augmentation by reconstruction pipeline

4.6. Data Augmentation with Filtering

As the outputs of autoencoders do not always perfectly replicate the original text, they can introduce variations in the information they encapsulate. This may result in reconstructed samples exhibiting characteristics reminiscent of other classes in the dataset, potentially harming classifier performance. Hence, there arises a need for a method to weed out falsely labeled reconstructed samples. One approach involves training a classifier solely on the original data and then employing it to classify reconstructed samples. If the classifier's prediction contradicts the label of the original sample, the reconstructed sample is discarded. This strategy ensures the reliability of the

reconstructed data. Figure 4.12 illustrates this process in a schematic diagram. The filtering mechanism with using a classifier trained with original data is adapted from [83].

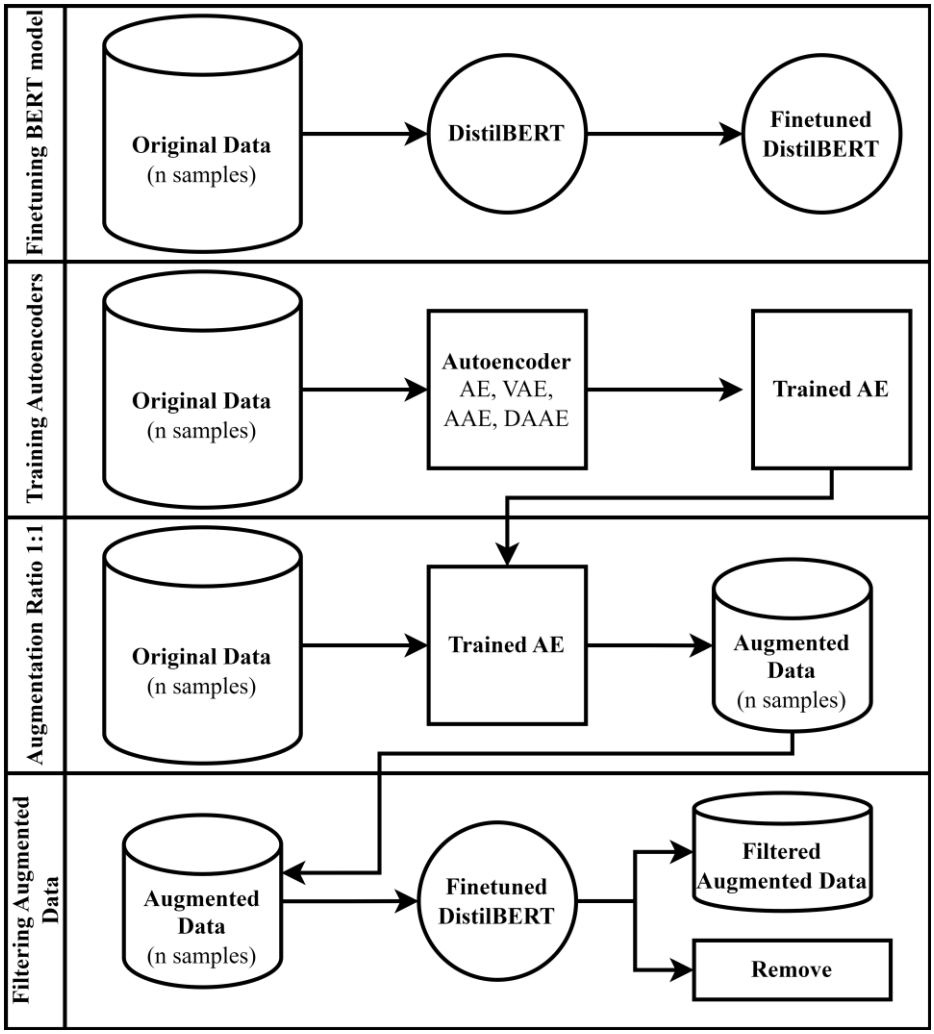


Figure 4.12. Filtering pipeline for augmentation with reconstruction

5. EXPERIMENTAL RESULTS

5.1. Dataset

The Stanford Sentiment Treebank (SST-2) dataset [84] is a common dataset which is used as a baseline for sentiment analysis tasks in NLP. It comprises movie reviews labeled with either positive or negative sentiments. The SST-2 dataset provides a valuable benchmark for evaluating the performance of sentiment classification models. Table 5.1 gives the numbers of positive-negative samples in the training, validation, and test sets of SST-2 dataset.

Table 5.1. Number of samples in each class of SST-2 dataset.

Class	Number of Samples		
	Training Set	Validation Set	Test Set
Positive	4054	900	909
Negative	3737	900	912
Total	7791	1800	1821

5.2. General Parameters for Data Augmentation

To evaluate the effectiveness of Data Augmentation (DA) methods across different scenarios, two parameters are chosen: $augmentation\ ratio$ and $dataset\ size$.

- $augmentation\ ratio$: This parameter refers to the ratio of synthetic sentences generated for each original sentence in the training set.
- $dataset\ size$: This parameter indicates the size of the training dataset. The original training dataset is divided into smaller portions to evaluate DA methods' performance in scenarios with limited data.

For each experiment, the final training set size ($\#_{training\ data}$) is determined using Eq. 5.1.

$$\#_{training\ data_t} = (augmentation\ ratio \times dataset\ size) + dataset\ size \quad (5.1)$$

5.3. Classifier

Throughout this thesis, all experiments utilize DistilBERT [81] which is explained in Section 4.4. The parameters employed in training DistilBERT across all experiments are outlined in Table 5.2.

Table 5.2. Parameters used for DistilBERT model.

Parameter	**Value**
Optimizer	AdamW [85]
Training Epochs	2
Batch Size	8
Learning Rate	5e-5
Activation Function	Gaussian Error Linear Units (GELUs)
Pretrain checkpoint	distilbert-base-uncased
Dropout	0.2
Loss Function	Cross entropy

5.4. Evaluation Metrics

When assessing machine learning models in supervised learning scenarios, commonly utilized evaluation metrics include accuracy, precision and recall [86]. However, due to the balanced nature of the SST-2 dataset, as observed from Table 5.1, the sole evaluation metric employed to gauge the model's performance in classification tasks is accuracy [86]. Accuracy is a metric that quantifies the proportion of correct classifications relative to the total number of classifications made. It is often favored for classification evaluations as it offers a straightforward and intuitive measure of a model's overall capability to classify instances accurately. Eq. 5.2 illustrates the calculation of accuracy.

$$Accuracy = \frac{Correct\ prediction}{total\ number\ of\ predictions} \tag{5.2}$$

5.5. Number of Training Epochs Comparison for Autoencoders

In this study, the influence of the number of training epochs on the efficacy of autoencoder-based textual DA is investigated. By selecting the duration of training epochs as 50 or 100, it is aimed to understand how the autoencoder's ability to reconstruct input texts evolves over time and its consequent impact on the quality and diversity of augmented data. Through qualitative and quantitative evaluations, it is analyzed how different numbers of epochs affect the model's capacity to capture underlying textual patterns and generate meaningful augmentations. In Figure 5.1, it is observed that when the number of epochs for autoencoders is 100, the classifier performs significantly better in dataset size 100. However, as the dataset size grows, this effect diminishes.

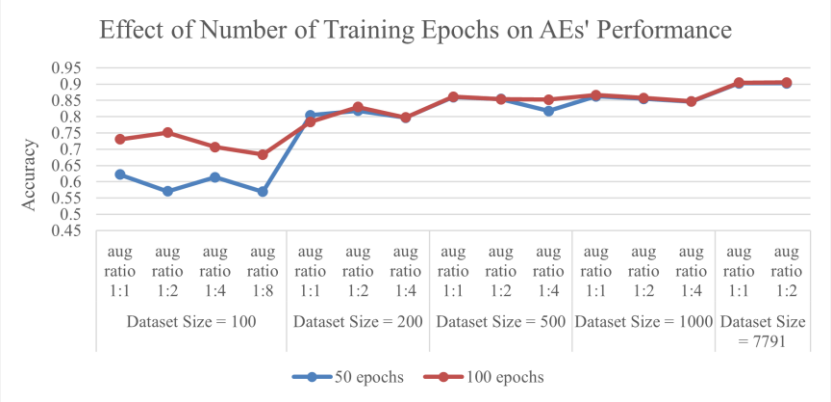
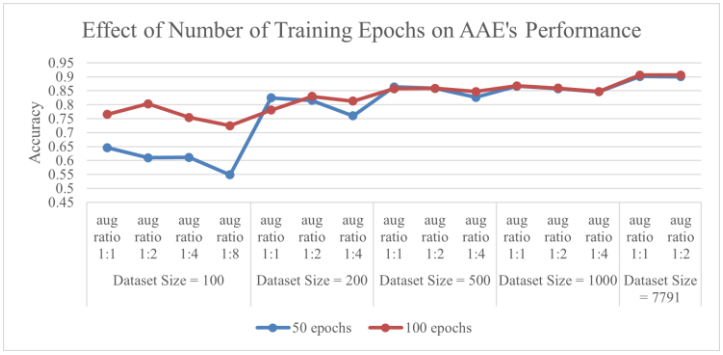
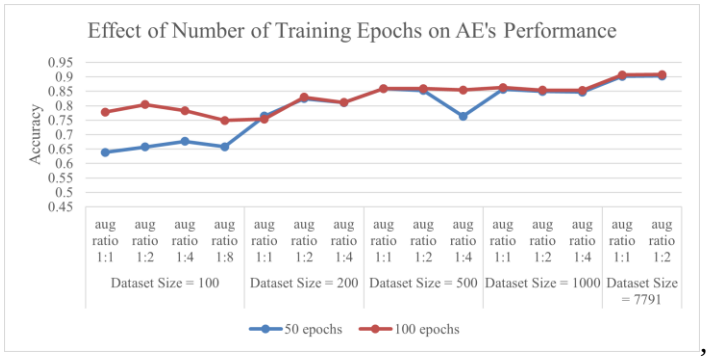


Figure 5.1. Average accuracies of DistilBERT classifier when the training epochs of autoencoders is 50 and 100, considering different dataset sizes and augmentation ratios.

Figure 5.2 presents 4 different line graphs illustrating the impact of the training epochs on different AEs' performance. For dataset sizes of 200, 500, 1000 and 7791, no notable effect is observed on any type of AE. For dataset size of 100, all AE types are boosted with increasing training epochs to 100. While the other three AE types boosted significantly, the impact of increasing training epochs on DAAE is relatively low.

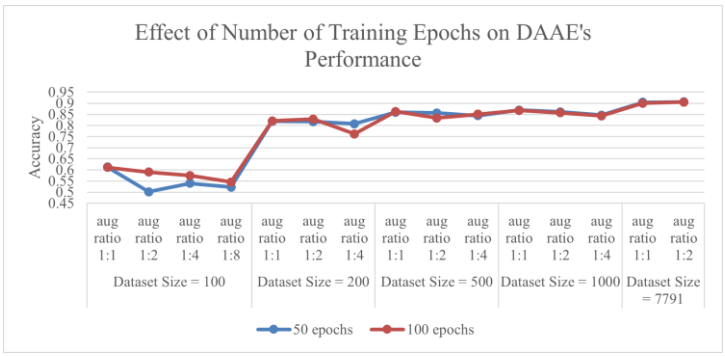


(a)

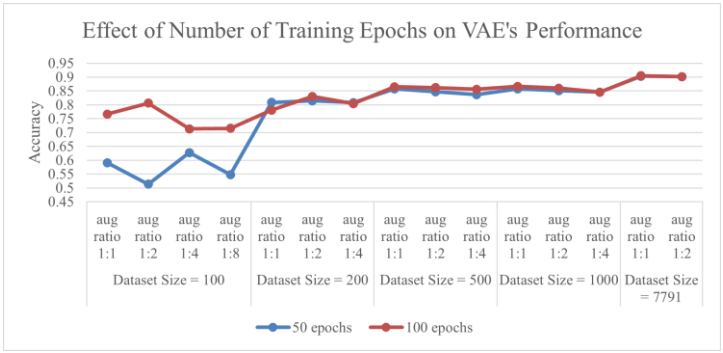


,

(b)



(c)



(d)

Figure 5.2. Average accuracies of DistilBERT classifier when the training epochs of autoencoders is 50 and 100, for (a) AAE, (b) AE, (c) DAAE and (d) VAE.

5.6. Effect of Filtering Pipeline

As outlined in Section 4.5, applying a filter on reconstructed samples could be useful. To gauge the efficacy of our filtering pipeline, a comparative analysis is conducted between the filtered and unfiltered pipelines, focusing on the average accuracies attained by classifiers trained on datasets augmented by each respective method, which is shown in Figure 5.3. The results of this comparison underscore a significant impact of the filtering process, particularly pronounced in datasets of smaller sizes, where the filtering mechanism yields discernible improvements in classification accuracy. However, as dataset size increases, the influence of filtering diminishes.

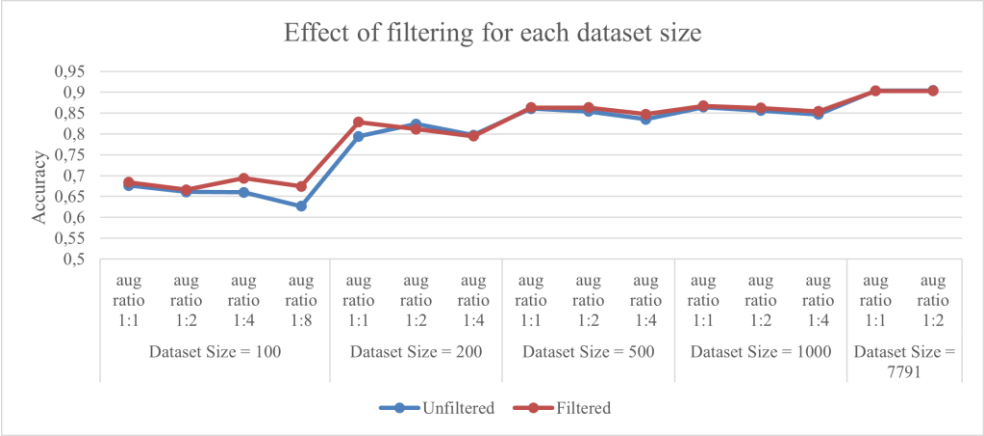
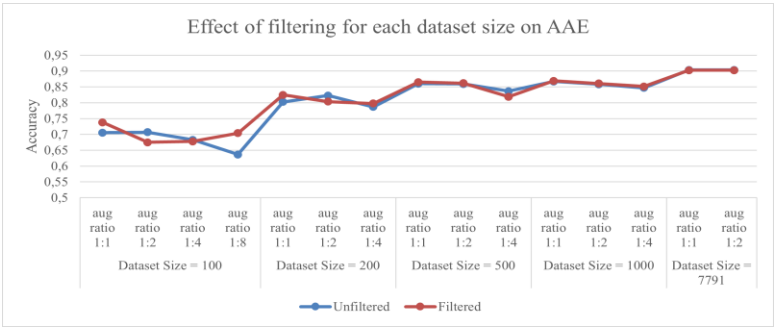
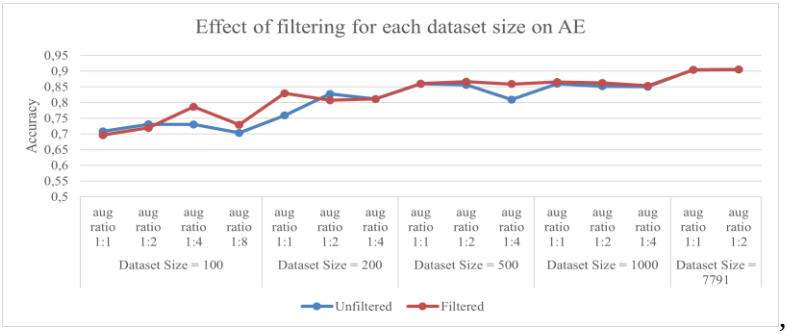


Figure 5.3. Average performances filtered pipeline and unfiltered pipeline of AE's for different dataset sizes and augmentation ratios.

Figure 5.4 presents 4 different line graphs illustrating the impact of filtering pipeline on different AEs' performance. For dataset sizes of 500, 1000 and 7791, no notable effect of filtering pipeline is observed on any type of AE. For dataset size of 100, all AE types are boosted with filtering pipeline, except traditional AE when augmentation ratio is 1:1 and AAE when augmentation ratio is 1:2. For the dataset size of 200, traditional AE and VAE boosted with filtering pipeline when augmentation ratio is 1:1.

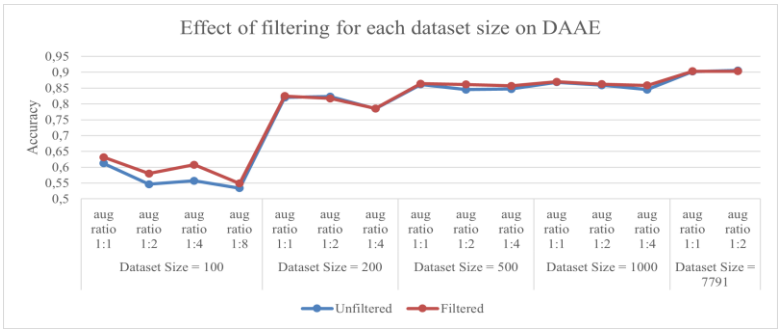


(a)

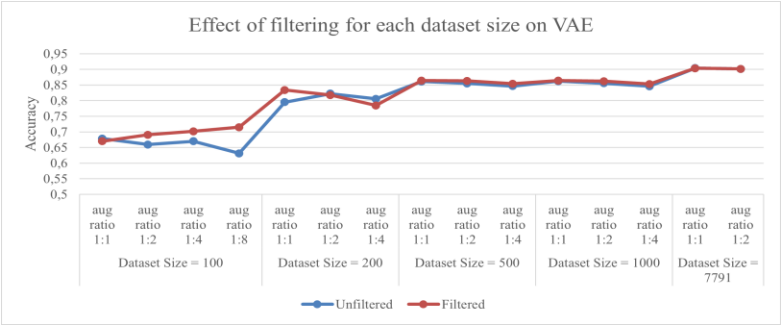


,

(b)



(c)



(d)

Figure 5.4. Average performances filtered pipeline and unfiltered pipeline of AE's for different dataset sizes and augmentation ratios, considering different dataset sizes and augmentation ratios for (a) AAE, (b) AE, (c) DAAE and (d) VAE.

5.7. Comparison of Different Autoencoders in Data Augmentation

The line chart depicted in Figure 5.5 presents the average accuracies of classifiers trained on datasets augmented with four distinct types of AEs. Observing Figure 5.5, it becomes evident that across dataset sizes of 200, 500, 1000, and 7791, classifiers exhibit similar performances when the augmentation ratio is 1:1. However, with an increase in the augmentation ratio within these dataset sizes, a notable decline in classifier accuracy is observed for AAE and VAE, whereas the drop is comparatively less pronounced for traditional AEs and DAAE. Notably, in the case of a dataset size of 100, DAAEs demonstrate poor performance across all augmentation ratios, with traditional AEs surpassing other AE types except at a 1:1 augmentation ratio. Moreover, in the dataset size of 100, both traditional AEs and VAEs exhibit a positive impact stemming from an increase in the augmentation ratio.

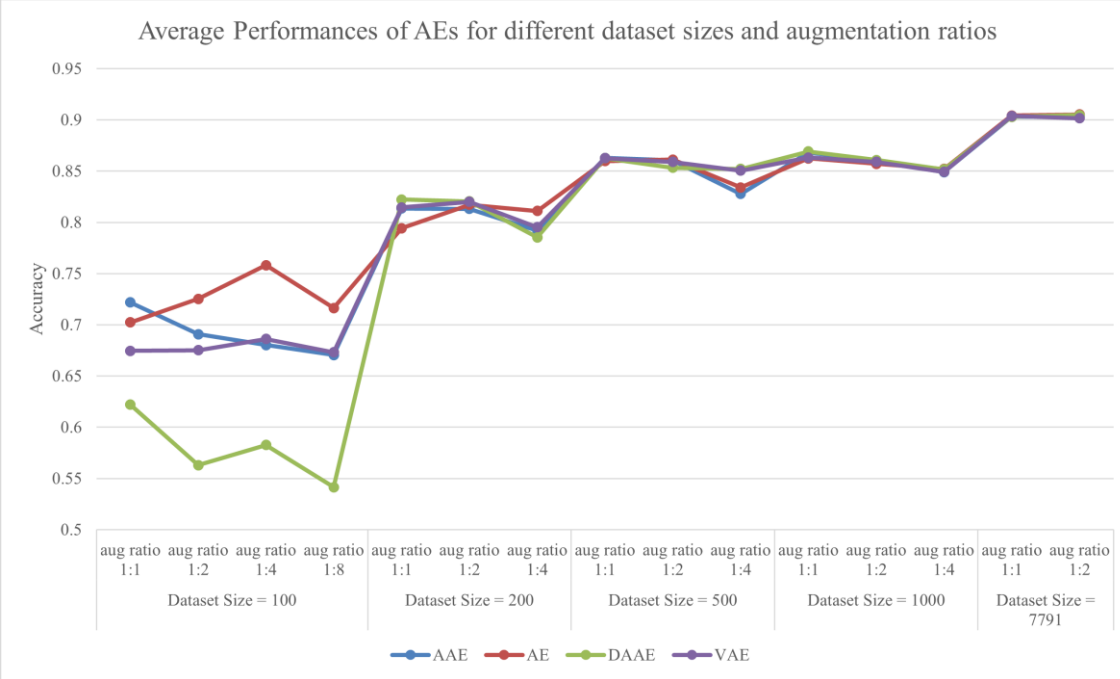


Figure 5.5. Average performances different types of AE's for different dataset sizes and augmentation ratios.

5.8. Embedding Comparison

In experiments, the effect of integrating pre-trained Word2Vec embeddings into the embedding layer of AEs is analyzed. Word2Vec enriches vocabulary variability and provides a contextually richer representation which comes with an increase in parameters and training time. The comparison is done between Word2Vec and bag-of-words (BoW) tokenization which simplifies text representation by counting word frequencies. Figure 5.6 shows the average accuracies of the classifier that uses data augmented with Word2Vec embedding layered AEs and BoW embedding layered AEs while Figure 5.7 shows the same analysis for each AE type.

As seen from Figure 5.6, using pretrained Word2Vec as embedding layer of AE's does not make much difference on dataset sizes of 200, 500, 1000 and 7791. Furthermore, it even decreases the performance of the classifier on augmentation ratios 1:2 and 1:8 of dataset size of 100. When the cost of using Word2Vec embeddings is considered, it becomes an irrelevant choice.

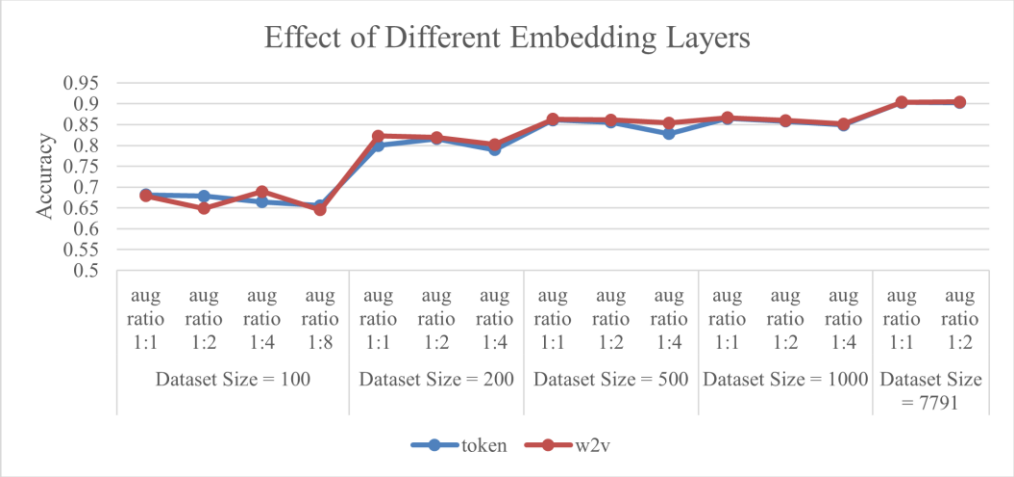
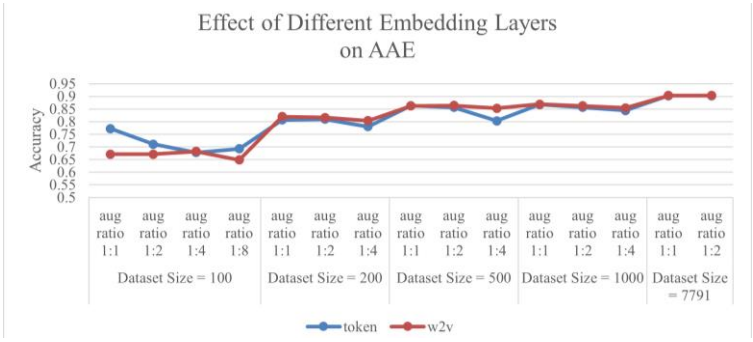
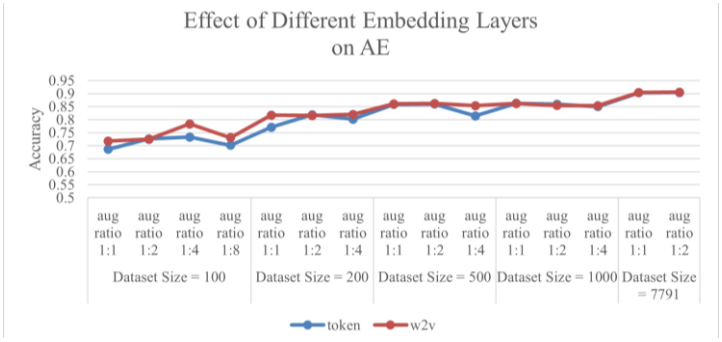


Figure 5.6. Average performances of different types of embedding layers of AE's for different dataset sizes and augmentation ratios.

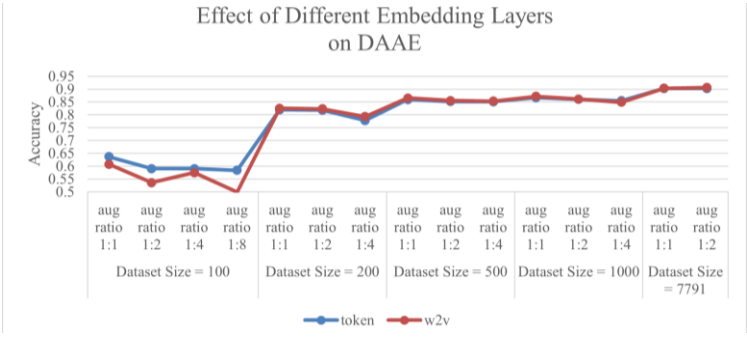
Figure 5.7 presents 4 different line graphs illustrating the impact of different embedding layers on different AEs performance. For dataset sizes of 200, 500, 1000 and 7791, no notable effect of using w2v as embedding layer is observed on any type of AE. For dataset size of 100, AAE and DAAE are affected adversely from using w2v as embedding layer for all augmentation ratios, while traditional AE and VAE's performances are relatively higher with w2v embeddings.



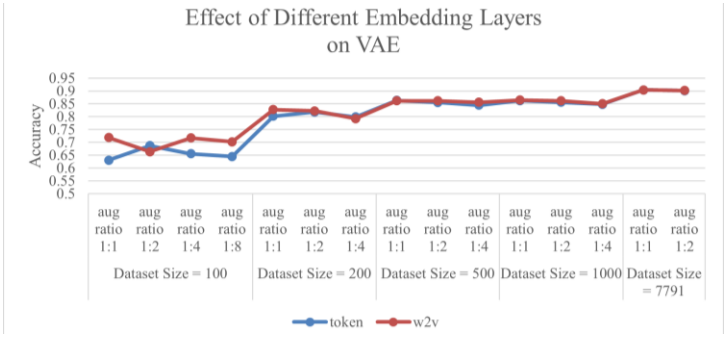
(a)



(b)



(c)



(d)

Figure 5.7. Average performances of different types of embedding layers of AE's, considering different dataset sizes and augmentation ratios for (a) AAE, (b) AE, (c) DAAE and (d) VAE.

5.9. Effect of Preprocessing on AE Input

Textual datasets are prone to being noisy which might cause ambiguity and confusion to AE models. Thus, a series of preprocessing steps is applied to the dataset used in this thesis. These preprocessing steps involve lowercasing, removal of non-alphanumeric characters and extra spaces and changing numeric characters with the tag $< number >$.

Figure 5.8 and Figure 5.9 present line graphs illustrating the impact of preprocessing on AE performance. As depicted in Figure 5.8, which displays the average accuracies for all AEs, significant enhancements are observed in smaller dataset sizes with preprocessing, resulting in approximately a 5% increase in accuracy for a dataset size of 100 and approximately a 2% increase in accuracy for a dataset size of 200.

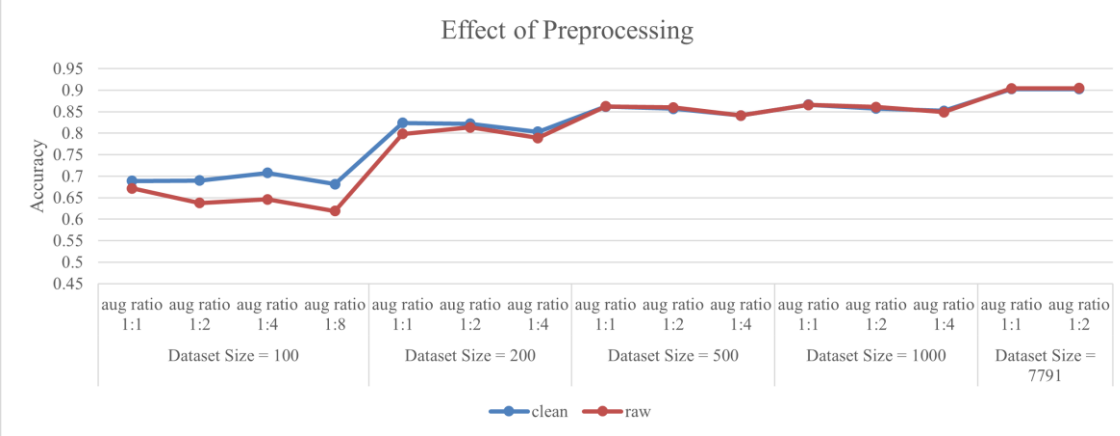
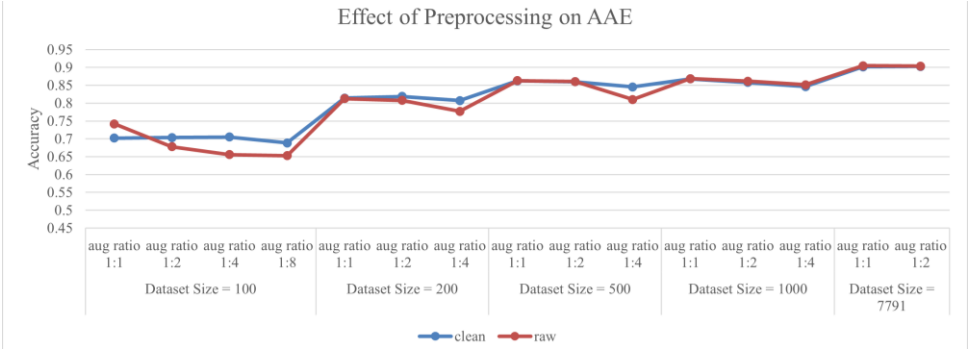
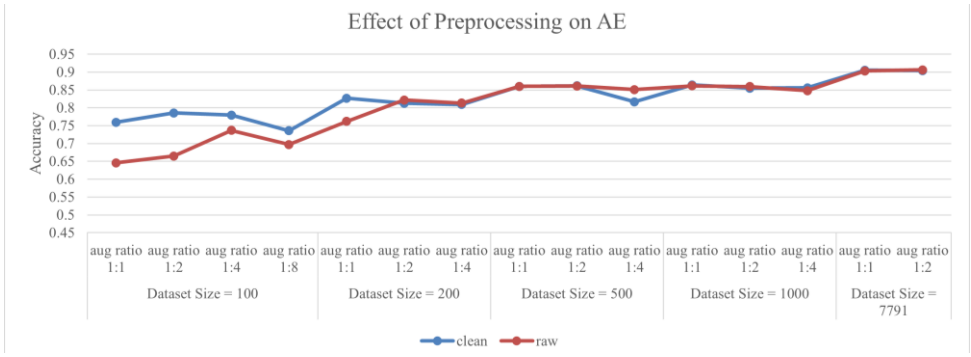


Figure 5.8. Average performances of AE's when preprocessing applied or not applied for different dataset sizes and augmentation ratios.

Figure 5.9 presents 4 different line graphs illustrating the impact of preprocessing on different AEs performance. For dataset sizes of 500, 1000 and 7791, no notable effect of applying preprocessing is observed on any type of AE. For dataset size of 100, only AAE and VAE are affected adversely from preprocessing when augmentation ratio is 1:1, while traditional AE's performance increased significantly. For other augmentation ratios of dataset size of 100, preprocessing boosted all AE's performance.

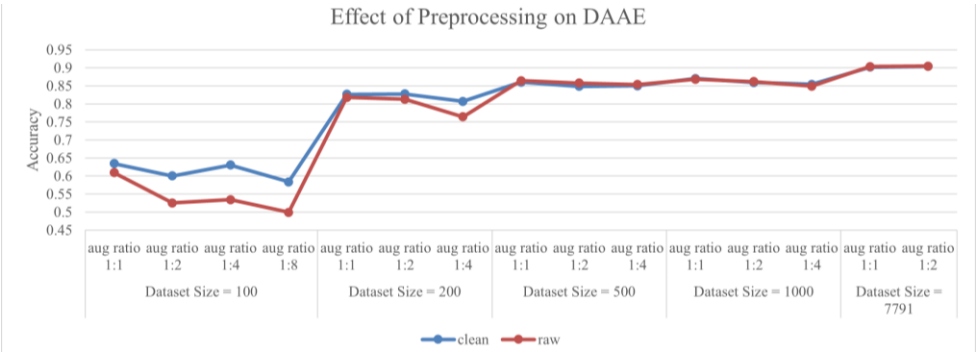


(a)

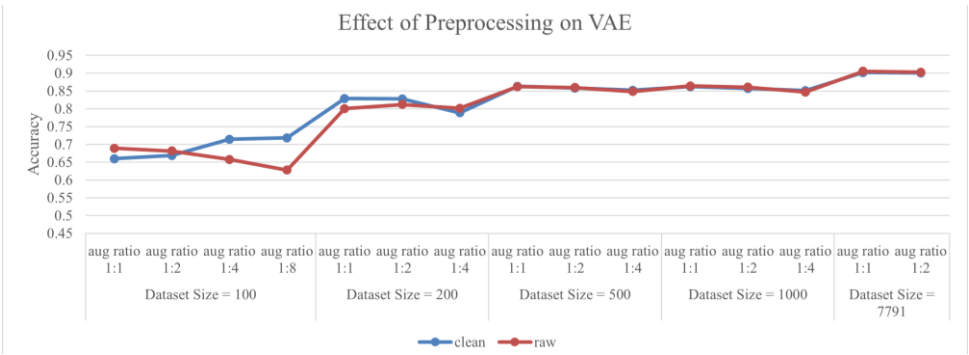


,

(b)



(c)



(d)

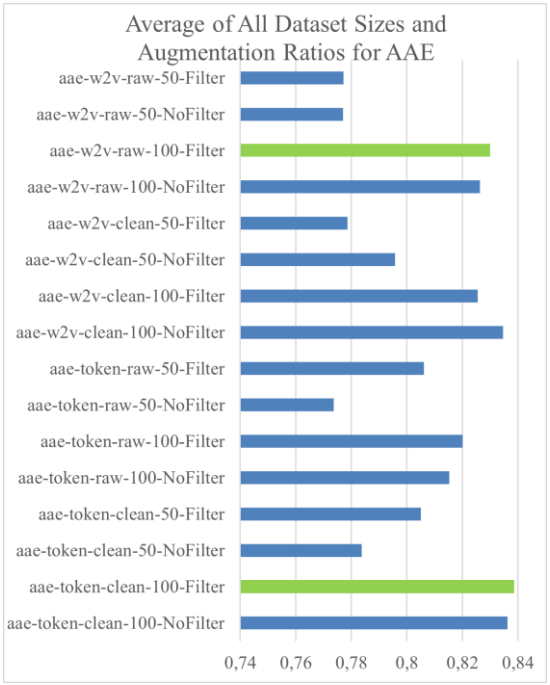
Figure 5.9. Average performances when preprocessing applied or not applied, considering different dataset sizes and augmentation ratios for (a) AAE, (b) AE, (c) DAAE and (d) VAE.

5.10. Comparison of Results with Baseline Methods

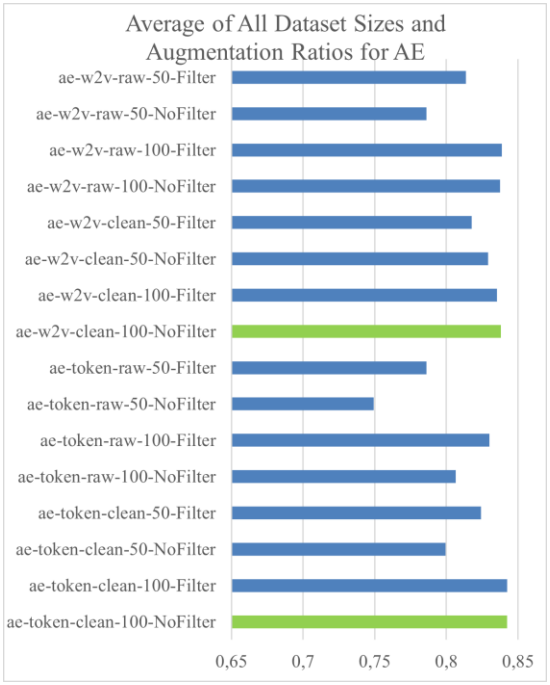
In the exploration of autoencoder-based text DA method, a thorough examination of numerous parameters was conducted to optimize its performance. These parameters included the type of AE, ranging from traditional AE to advanced variations which are AAE, VAE, and DAAE. Additionally, consideration was given to the preprocessing of data, whether specific preprocessing techniques were applied or not. Furthermore, different embedding layer types, including Word2Vec (w2v) and token embeddings, were investigated to evaluate their impact on augmentation quality. The number of training epochs for the autoencoders selected as 50 or 100 epochs, and the effectiveness of applying filtering techniques during augmentation was also explored. In total, these parameter permutations yielded 32 distinct variations of autoencoders.

The naming convention for the AE-based DA methods consists of abbreviations representing key parameters. These include the type of autoencoder (“ae” for traditional AE, "aae" for AAE, “vae” for VAE, “daae” for DAAE), the type of embeddings used ("token" for Bag-of-Words or "w2v" for Word2Vec), data preprocessing ("clean" or "raw"), the training epochs (50 or 100 epochs), and whether filtering techniques were applied during augmentation ("Filter" or "NoFilter"). As an example, “aae-token-clean-100-Filter” refers to AAE-based DA method with Bag-of-Words tokenization, preprocessing applied, trained for 100 epochs and filtering pipeline applied. This naming system enables straightforward identification and comparison of different DA methods based on their key characteristics.

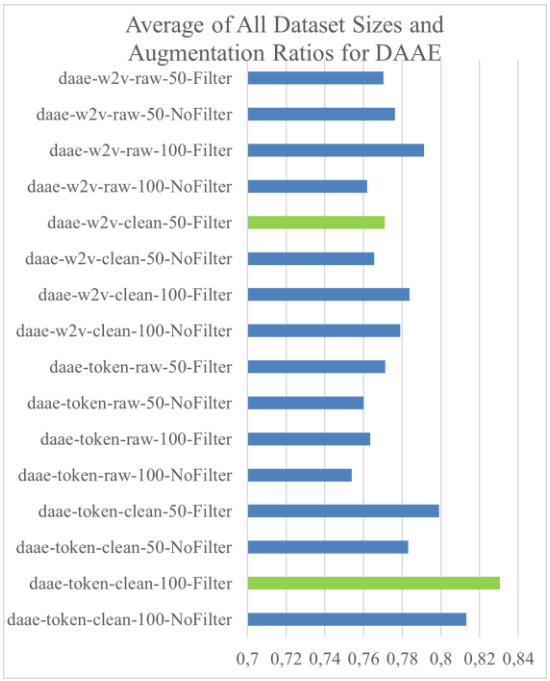
To facilitate comparison with existing literature, two variants from each autoencoder type were selected for detailed analysis in this section. When selecting these variants, the parameters considered were their overall performance on all dataset sizes and augmentation ratios and their performances on each dataset size separately. Figure 5.10 shows the average accuracies of each AE type’s all variants. The selected variants are highlighted with green. As can be seen from Figure 5.10, most selected variants are selected because they have the best average performance on all dataset sizes and augmentation ratios. There are a few exceptions that do not follow this rule such as “daae-w2v-clean-50-Filter”. The reason for such selections is that these variants perform better on most dataset sizes compared to the others from the same AE type, but their average accuracy is affected adversely due to these variants’ poor performance on particular dataset size.



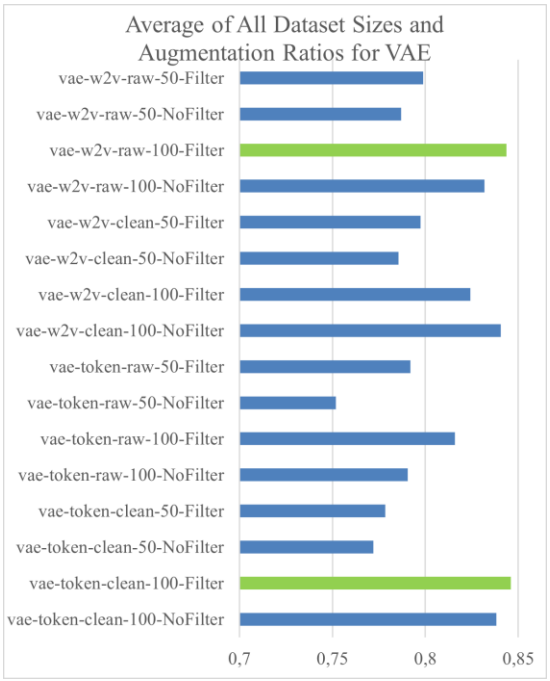
(a) AAE



(b) AE



(c) DAAE



(d) VAE

Figure 5.10. Average of accuracies for each variant of (a) AAE, (b) AE, (c) DAAE, and (d) VAE.

The selected baseline models from literature include An Easier Data Augmentation (AEDA) [27] as noising-based DA method, Easy Data Augmentation (EDA) [28] as paraphrasing-based DA method, and Language Model Based Data Augmentation (LAMBADA) [83] as sampling-based DA method. These models were chosen to provide a comparison across diverse data augmentation strategies. EDA is a popular DA method commonly used as a baseline, providing a benchmark for evaluating other data augmentation techniques. AEDA is a completely noise-based and simple method, which helps to understand the impact of random noise on data augmentation. LAMBADA is the best-performing model among the baselines and is powered by a pretrained large language model, demonstrating the effectiveness of leveraging advanced pretrained models for data augmentation. However, the augmentation strategy proposed in this thesis, based on autoencoders, is distinct in that it does not depend on any other pretrained models. By comparing our autoencoder-based augmentation strategy with these well-established techniques, it is aimed to highlight its strengths and potential advantages, as well as identify areas for further improvement.

Tables 5.2 through 5.6 present the effect of each DA method on accuracy of DistilBERT classifier for various augmentation ratios corresponding to dataset sizes of 100, 200, 500, 1000, and 7791 (Full set), respectively.

Table 5.3 presents the accuracy results of DistilBERT on datasets augmented using various DA methods, with a dataset size of 100. Initially, without any augmentation, the DistilBERT classifier achieves an accuracy of 49.9%, resembling the performance of a random classifier for a two-class dataset. When DA methods are applied with a 1:1 augmentation ratio, all methods exhibit significant performance improvements, ranging from a gain of 19.6% to 33.9%, except for daae-w2v-clean-50-NoFilter, which struggles to effectively generate textual data with this dataset size. For augmentation ratios of 1:2, 1:4, and 1:8, LAMBADA consistently outperforms other DA methods, as expected due to its sampling-based approach powered by a pretrained LLM. AE-based DA methods yield the best results for a 1:1 augmentation ratio. Although performance seems to decrease as the augmentation ratio increases, the results achieved in data augmentation using AE are higher than those achieved without DA. This shows that AE-based DA can be used on low data set sizes.

Table 5.3. Comparison of the selected AE-based DA methods with baseline models from the literature for the dataset size of 100 on SST-2 dataset.

	Dataset Size = 100	Accuracy				
		No Aug.	Aug. ratio 1:1	Aug. ratio 1:2	Aug. ratio 1:4	Aug. ratio 1:8
	No Augmentation	0.499	-	-	-	-
DA methods	LAMBADA (GPT-2)	-	0.836	**0.839**	**0.839**	**0.844**
	EDA	-	0.793	0.817	0.755	0.768
	AEDA	-	0.799	0.806	0.772	0.757
	aae-token-clean-100-Filter	-	0.835	0.821	0.76	0.745
	aae-w2v-raw-100-Filter	-	0.711	0.822	0.704	0.762
	ae-token-clean-100-NoFilter	-	0.833	0.796	0.795	0.771
	ae-w2v-clean-100-NoFilter	-	0.831	0.8	0.813	0.76
	daae-token-clean-100-Filter	-	0.695	0.74	0.798	0.812
	daae-w2v-clean-50-NoFilter	-	0.508	0.499	0.499	0.499
	vae-token-clean-100-Filter	-	0.787	0.821	0.818	0.792
	vae-w2v-raw-100-Filter	-	**0.838**	0.822	0.814	0.722

Table 5.4 displays the accuracy results of DistilBERT on datasets augmented using various DA methods, with a dataset size of 200. Initially, without any augmentation, the DistilBERT classifier achieves an accuracy of 82.8%, a significant improvement compared to the dataset size of 100. When applying DA methods with a 1:1 augmentation ratio, EDA, AEDA, and two AE-based methods show a slight decrease in classifier performance, approximately 1%. Conversely, VAE-based DA methods and LAMBADA exhibit an increase of approximately 2% in classifier performance. For augmentation ratios of 1:2 and 1:4, LAMBADA consistently outperforms other DA methods, like the dataset size of 100. Additionally, AE-based DA methods demonstrate the best results for augmentation ratio 1:1. Although the performance seems to decrease compared to that of augmentation ratio 1:1, the results obtained in higher augmentation ratios are at a level that can compete with the methods from the literature.

Table 5.4. Comparison of the selected AE-based DA methods with baseline models from the literature for the dataset size of 200 on SST-2 dataset.

Dataset Size = 200		Accuracy			
		No Aug.	Aug. ratio 1:1	Aug. ratio 1:2	Aug. ratio 1:4
	No Augmentation	0.828	-	-	-
DA methods	LAMBADA (GPT-2)	-	0.844	**0.845**	**0.853**
	EDA	-	0.812	0.824	0.813
	AEDA	-	0.818	0.796	0.805
	aae-token-clean-100-Filter	-	0.834	0.806	0.824
	aae-w2v-raw-100-Filter	-	0.824	0.825	0.807
	ae-token-clean-100-NoFilter	-	0.841	0.828	0.812
	ae-w2v-clean-100-NoFilter	-	0.824	0.834	0.802
	daae-token-clean-100-Filter	-	0.815	0.807	0.811
	daae-w2v-clean-50-NoFilter	-	0.839	0.831	0.829
	vae-token-clean-100-Filter	-	0.844	0.823	0.802
	vae-w2v-raw-100-Filter	-	**0.846**	0.83	0.796

Table 5.5 presents the accuracy results of DistilBERT on datasets augmented using various DA methods, with a dataset size of 500. Initially, without any augmentation, the DistilBERT classifier achieves an accuracy of 85%, indicating a 2.2% increase compared to the dataset size of 200, as expected. When DA methods are applied with augmentation ratio 1:1, all methods show an improvement in classifier performance. Like other dataset sizes, the VAE-based DA method demonstrates the highest increase, approximately 2.3%. For augmentation ratio of 1:2, the "aae-w2v-raw-100-Filter" DA method outperforms other methods, unlike any other AE-based DA method. For a 1:4 augmentation ratio, LAMBADA outperforms other methods, consistent with dataset sizes of 100 and 200 In addition, for dataset sizes of 100 and 200, the performance of AE-based DA methods does not decrease as the augmentation ratio increases. In fact, in some cases, it increases for a 1:2 augmentation ratio.

Table 5.5. Comparison of the selected AE-based DA methods with baseline models from the literature for the dataset size of 500 on SST-2 dataset.

	Dataset Size = 500	Accuracy			
		No Aug.	Aug. ratio 1:1	Aug. ratio 1:2	Aug. ratio 1:4
	No Augmentation	0.850	-	-	-
DA methods	LAMBADA (GPT-2)	-	0.858	0.864	**0.87**
	EDA	-	0.853	0.863	0.838
	AEDA	-	0.855	0.85	0.85
	aae-token-clean-100-Filter	-	0.866	0.865	0.863
	aae-w2v-raw-100-Filter	-	0.866	**0.877**	0.85
	ae-token-clean-100-NoFilter	-	0.856	0.857	0.849
	ae-w2v-clean-100-NoFilter	-	0.872	0.863	0.853
	daae-token-clean-100-Filter	-	0.863	0.866	0.861
	daae-w2v-clean-50-NoFilter	-	0.866	0.864	0.855
	vae-token-clean-100-Filter	-	**0.873**	0.868	0.864
	vae-w2v-raw-100-Filter	-	0.867	0.863	0.865

Table 5.6 displays the accuracy results of DistilBERT when applied to datasets augmented using different DA methods, each with a dataset size of 1000. Initially, without any augmentation, the DistilBERT classifier achieves an accuracy of 87.5%, marking a 2.5% increase compared to the dataset size of 500. When DA methods are employed with a 1:1 augmentation ratio, none of them manage to surpass the result without augmentation, with some experiencing a decrease in the classifier's performance of up to 1.9%. Notably, DAAE-based DA methods exhibit improved performance as the dataset size increases, being the only method capable of achieving the same accuracy as the scenario with no augmentation. For a 1:2 augmentation ratio, EDA emerges as the top performer, matching the accuracy of the scenario without augmentation, while most AE-based DA methods and LAMBADA show similar performance. Conversely, with a 1:4 augmentation ratio, LAMBADA exhibits the highest performance, while the accuracy of other DA methods remained the same.

Table 5.6. Comparison of the selected AE-based DA methods with baseline models from the literature for the dataset size of 1000 on SST-2 dataset.

Dataset Size = 1000		Accuracy			
		No Aug.	Aug. ratio 1:1	Aug. ratio 1:2	Aug. ratio 1:4
No Augmentation		0.875	-	-	-
DA methods	LAMBADA (GPT-2)	-	0.874	0.866	**0.879**
	EDA	-	0.86	**0.875**	0.853
	AEDA	-	0.874	0.872	0.859
	aae-token-clean-100-Filter	-	0.863	0.862	0.83
	aae-w2v-raw-100-Filter	-	0.87	0.86	0.861
	ae-token-clean-100-NoFilter	-	0.874	0.859	0.856
	ae-w2v-clean-100-NoFilter	-	0.856	0.846	0.855
	daae-token-clean-100-Filter	-	**0.875**	0.85	0.862
	daae-w2v-clean-50-NoFilter	-	0.87	0.858	0.854
	vae-token-clean-100-Filter	-	0.866	0.864	0.863
	vae-w2v-raw-100-Filter	-	0.862	0.872	0.854

Table 5.7 presents the accuracy outcomes of DistilBERT applied to datasets augmented using various DA methods, each with a dataset size of 7791, which constitutes the full set size. Without any augmentation, the DistilBERT classifier achieves an accuracy of 90.3%, indicating a 2.8% increase compared to the dataset size of 1000. Upon analyzing the results of the DA methods, it is observed that none of them significantly enhance the performance of DistilBERT. On the other hand, there is no DA method that affects the results too adversely unlike other dataset sizes. Table 5.7 excludes results for an augmentation ratio of 1:4 due to resource limitations. Although the success rate without DA is 90.3%, it is seen that DA achieves the same success in all other methods.

Table 5.7. Comparison of the selected AE-based DA methods with baseline models from the literature for the dataset size of 7791 (Full Set) on SST-2 dataset.

Dataset Size = 7791 (Full Set)		Accuracy		
		No Aug.	Aug. ratio 1:1	Aug. ratio 1:2
No Augmentation		0.903	-	-
DA methods	LAMBADA (GPT-2)	-	**0.909**	0.902
	EDA	-	0.90	0.906
	AEDA	-	0.906	0.906
	aae-token-clean-100-Filter	-	0.901	0.907
	aae-w2v-raw-100-Filter	-	**0.909**	0.905
	ae-token-clean-100-NoFilter	-	0.907	0.906
	ae-w2v-clean-100-NoFilter	-	0.907	**0.912**
	daae-token-clean-100-Filter	-	0.901	0.903
	daae-w2v-clean-50-NoFilter	-	0.904	0.909
	vae-token-clean-100-Filter	-	0.907	0.900
	vae-w2v-raw-100-Filter	-	0.906	0.902

* No results available for augmentation ratio 1:4

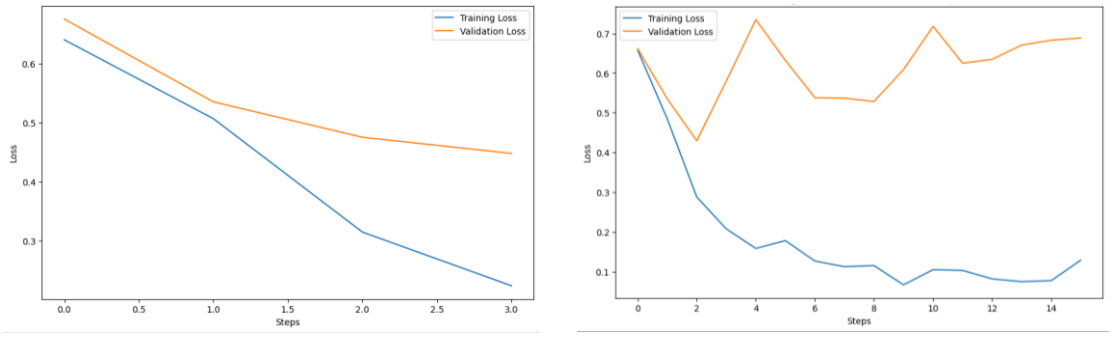
5.11. Interpretation of Generation of Synthetic Samples from An Original Sample

In this section, reconstructions of autoencoders on various aspects are interpreted through example outputs provided in Appendix A. When outputs in Appendix A are analyzed, the observations on different aspects of this thesis are as follows:

- Preprocessing: For dataset sizes of 100, 200, 500, and 1000; the methods where preprocessing applied generally produce sentences that closely resemble the original sentence, demonstrating limited variability. In contrast, methods where preprocessing is not applied, particularly those using Word2Vec, introduce more variation but often include <unk> tokens.
- Embedding Types: Token-based methods tend to maintain the original sentence structure more faithfully with small changes in the sentences, while Word2Vec methods introduce more semantic diversity.
- Different AE Types: At small dataset sizes, Traditional AEs and AAE can generate synthetic sentences that are more consistent with the original sentence compared to DAAE and VAE. On the other hand, for higher dataset sizes,

Traditional AE and AAE are not able to generate different variations, mostly reconstructing the original sentence, while DAAE and VAE is more successful at generating sentences that varies from the original sentence.

As observed from some outputs from Appendix A (Table A.3-ae-token-clean-100 or Table A.4- aae-token-clean-100), some AEs learn to replicate the original sentences without introducing sufficient diversity. This replication leads to multiple entries of the same sample in the augmented dataset. This lack of variability in augmented text data can affect adversely text classification models, especially with higher augmentation ratios. Because, when generated sentences are too similar to the original, the model may fail to learn robust, generalized features, leading to overfitting—where the model performs well on training data but poorly on unseen data. This issue is worsened at higher augmentation ratios, where increased data quantity lacks diversity, causing the model to memorize rather than generalize. An example of this situation is seen on Figure 5.11 where the validation and training losses of two DistilBERT models are compared, one trained with data augmented by "aae-token-clean-100-NoFilter" with augmentation ratio 1:1 and the other with augmentation ratio 1:8. "aae-token-clean-100-NoFilter" fails to generate new variations of original text as can be seen in Appendix A. This situation resulted in a decrease on training loss while an increase on validation loss which implies overfitting for the model trained with data with augmentation ratio 1:8, which is not observed for the model trained with data with augmentation ratio 1:8 in Figure 5.11.



(a) aae-token-clean-100-NoFilter with augmentation ratio 1:1

(b) aae-token-clean-100-NoFilter with augmentation ratio 1:8

Figure 5.11. Training and validation losses for two DistilBERT classifier: (a) trained with aae-token-clean-100-NoFilter with augmentation ratio 1:1 dataset and (b) aae-token-clean-100-NoFilter with augmentation ratio 1:8.

6. CONCLUSION

In recent years, the emergence of Deep Learning (DL) techniques has transformed the field of Natural Language Processing (NLP), facilitating significant progress in various tasks such as named entity recognition, language modeling, and question answering. However, the success of DL models largely depends on the availability and quality of the training data. Given the complex and diverse nature of natural language, acquiring a sufficiently large and diverse dataset can be challenging. This situation creates the need for data augmentation (DA). DA techniques seek to artificially increase the size and diversity of the training data by applying various transformations. In the context of NLP, where annotated datasets are often limited and expensive to obtain, data augmentation emerges as a crucial tool for enhancing model performance and generalization.

This thesis focuses on exploring the effect of leveraging the reconstruction capabilities of autoencoders as data augmentation method for enhancing performance of text classification tasks. In the scope of the work, four distinct types of autoencoders are investigated, namely traditional Autoencoders (AE), Variational Autoencoders (VAE), Denoising Adversarial Autoencoders (DAAE), and Adversarial Autoencoders (AAE). Two primary embedding types, Bag-of-Words and Word2Vec, are considered for representing textual data, and their effect on the performance of autoencoders are analyzed. Additionally, the impact of preprocessing methods is examined. Furthermore, the training duration, represented by the number of epochs, selected from 50 or 100 epochs to assess its influence on model performance. Finally, the effectiveness of applying a specific filtering technique, determining whether to retain or discard augmented samples based on its consistency with the prediction of a classifier is evaluated.

To assess the effectiveness of leveraging autoencoders for data augmentation in text classification tasks, experiments were conducted using the SST-2 dataset and accuracy as the evaluation metric. The performance in different data availability scenarios was evaluated by changing the size of the dataset partitions as 100, 200, 500, 1000 randomly selected data from this dataset, including the full dataset containing 7791 samples. Additionally, augmentation ratios indicating the ratio of augmented samples to original samples ranging as 1:1, 1:2, 1:4 and 1:8 were experimented. The performance of autoencoder-based data augmentation methods was compared with three baseline models

which are Easy Data Augmentation (EDA), An Easy Data Augmentation (AEDA), and Language Model Based Data Augmentation (LAMBADA). These baseline models were compared with two selected variants from each AE type.

The contributions of the thesis are as follows:

- Autoencoder-based data augmentation methods for enhancing text classification performance were investigated.
- Four distinct types of autoencoders (AE, VAE, DAAE, AAE) in the context of data augmentation for text classification task were evaluated.
- The impact of embedding types (Bag-of-Words, Word2Vec) on the performance of autoencoder-based data augmentation was analyzed.
- Preprocessing's influence on model performance was examined.
- The effect of training epochs (50 vs. 100) on the efficacy of autoencoder-based data augmentation was evaluated.
- Augmentation ratios (1:1, 1:2, 1:4, 1:8) were analyzed to determine optimal augmentation strategies.
- The effectiveness of filtering pipeline in improving the quality of augmented data was evaluated.
- Varying dataset partition sizes to assess model performance under different data availability scenarios were analyzed.

The experiments conducted across varying dataset sizes revealed intriguing insights into the performance of AE-based DA methods for text classification tasks. Notably, for dataset sizes of 100 and 200, Traditional AE and VAE demonstrated superior performance compared to other AE types, highlighting their effectiveness in augmenting small datasets. Moreover, preprocessing had a considerable impact on performance for smaller dataset sizes, whereas its influence diminished for larger dataset sizes. Interestingly, the choice of embedding type did not significantly affect performance, with w2v offering no considerable advantage over BoW despite its higher computational cost. However, increasing the number of training epochs from 50 to 100 and implementing a filtering pipeline proved beneficial, particularly for poorly performing AEs on smaller dataset sizes. When compared with baseline models, AE-based DA methods exhibited

superior performance on augmentation ratio of 1:1. However, for higher augmentation ratios, LAMBADA, which requires a pretrained LLM, outperformed other DA methods. These findings highlight the importance of choosing augmentation strategies based on dataset size, preprocessing methods, and augmentation ratio to maximize performance gains in text classification tasks.

While this thesis primarily focuses on leveraging the reconstruction capabilities of autoencoders for data augmentation in text classification tasks, there exist promising avenues for further exploration in this domain. One potential direction is to explore alternative ways of utilizing autoencoders beyond direct reconstruction of input text. For instance, instead of generating augmented samples by reconstructing input data, autoencoders could be utilized to sample from the latent space or generate synthetic samples solely from the decoder component. This approach could offer greater flexibility in generating diverse and realistic augmented data. Moreover, considering the potential computational complexity of training autoencoders on large datasets, in future research, strategies for training autoencoders on a partition of the dataset and then utilizing the trained autoencoder to reconstruct unseen data samples could be investigated. Additionally, exploring novel architectures or variations of autoencoders tailored specifically for text data could lead to more effective data augmentation methods. Furthermore, integrating autoencoder-based data augmentation techniques with other augmentation strategies or ensemble methods could be explored to further enhance model robustness and generalization across diverse datasets and tasks.

REFERENCES

[1] P. Goyal, S. Pandey, K. Jain, Deep learning for natural language processing, New York: Apress, (**2018**).

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

[3] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation, 9 (**1997**) 1735-1780.

[4] J. Chen, D. Tam, C. Raffel, M. Bansal, D. Yang, An empirical survey of data augmentation for limited data learning in NLP, Transactions of the Association for Computational Linguistics, 11 (**2023**) 191-211.

[5] Y. Wang, G.-Y. Wei, D. Brooks, A systematic methodology for analysis of deep learning hardware and software platforms, Proceedings of Machine Learning and Systems, 2 (**2020**) 30-43.

[6] A. Torfi, R.A. Shirvani, Y. Keneshloo, N. Tavaf, E.A. Fox, Natural language processing advancements by deep learning: A survey, arXiv preprint arXiv:2003.01200, (**2020**).

[7] Q.T. Ain, M. Ali, A. Riaz, A. Noureen, M. Kamran, B. Hayat, A. Rehman, Sentiment analysis using deep learning techniques: a review, International Journal of Advanced Computer Science and Applications, 8 (**2017**).

[8] S. Ali, K. Masood, A. Riaz, A. Saud, Named entity recognition using deep learning: A review, 2022 International Conference on Business Analytics for Technology and Security (ICBATS), IEEE, **2022**, pp. 1-7.

[9] L. Deng, Y. Liu, Deep learning in natural language processing, Springer**2018**.

[10] T. Iqbal, S. Qureshi, The survey: Text generation models in deep learning, Journal of King Saud University-Computer and Information Sciences, 34 (**2022**) 2515-2528.

[11] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep learning--based text classification: a comprehensive review, ACM computing surveys (CSUR), 54 (**2021**) 1-40.

[12] G. Marcus, Deep learning: A critical appraisal, arXiv preprint arXiv:1801.00631, (**2018**).

[13] L. Alzubaidi, J. Bai, A. Al-Sabaawi, J. Santamaría, A.S. Albahri, B.S.N. Al-dabbagh, M.A. Fadhel, M. Manoufali, J. Zhang, A.H. Al-Timemy, A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications, Journal of Big Data, 10 (**2023**) 46.

[14] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the royal statistical society: series B (methodological), 39 (**1977**) 1-22.

[15] K. Maharana, S. Mondal, B. Nemade, A review: Data pre-processing and data augmentation techniques, Global Transitions Proceedings, 3 (**2022**) 91-99.

[16] M. Razno, Machine learning text classification model with NLP approach, Computational Linguistics and Intelligent Systems, 2 (**2019**) 71-73.

[17] S. González-Carvajal, E.C. Garrido-Merchán, Comparing BERT against traditional machine learning text classification, arXiv preprint arXiv:2005.13012, (**2020**).

[18] S. Seo, C. Kim, H. Kim, K. Mo, P. Kang, Comparative study of deep learning-based sentiment classification, IEEE Access, 8 (**2020**) 6861-6875.

[19] S.-A. Rebuffi, S. Gowal, D.A. Calian, F. Stimberg, O. Wiles, T.A. Mann, Data augmentation can improve robustness, Advances in Neural Information Processing Systems, 34 (**2021**) 29935-29948.

[20] E.D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q.V. Le, Autoaugment: Learning augmentation strategies from data, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, **2019**, pp. 113-123.

[21] X. Yi, E. Walia, P. Babyn, Generative adversarial network in medical imaging: A review, Medical image analysis, 58 (**2019**) 101552.

[22] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, Journal of big data, 6 (**2019**) 1-48.

[23] T. Ko, V. Peddinti, D. Povey, S. Khudanpur, Audio augmentation for speech recognition, Interspeech, **2015**, pp. 3586.

[24] T. Ko, V. Peddinti, D. Povey, M.L. Seltzer, S. Khudanpur, A study on data augmentation of reverberant speech for robust speech recognition, 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, **2017**, pp. 5220-5224.

[25] D.S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E.D. Cubuk, Q.V. Le, Specaugment: A simple data augmentation method for automatic speech recognition, arXiv preprint arXiv:1904.08779, (**2019**).

[26] B. Li, Y. Hou, W. Che, Data augmentation approaches in natural language processing: A survey, Ai Open, 3 (**2022**) 71-90.

[27] A. Karimi, L. Rossi, A. Prati, AEDA: An Easier Data Augmentation Technique for Text Classification, Findings of the Association for Computational Linguistics: EMNLP 2021, **2021**, pp. 2748-2754.

[28] J. Wei, K. Zou, Eda: Easy data augmentation techniques for boosting performance on text classification tasks, arXiv preprint arXiv:1901.11196, (**2019**).

[29] F.M. Luque, Atalaya at TASS 2019: Data augmentation and robust embeddings for sentiment analysis, arXiv preprint arXiv:1909.11241, (**2019**).

[30] G. Yan, Y. Li, S. Zhang, Z. Chen, Data augmentation for deep learning of judgment documents, Intelligence Science and Big Data Engineering. Big Data

and Machine Learning: 9th International Conference, IScIDE 2019, Nanjing, China, October 17–20, 2019, Proceedings, Part II 9, Springer, **2019**, pp. 232-242.

[31] W.Y. Wang, D. Yang, That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets, Proceedings of the 2015 conference on empirical methods in natural language processing, **2015**, pp. 2557-2563.

[32] X. Wu, S. Lv, L. Zang, J. Han, S. Hu, Conditional bert contextual augmentation, Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19, Springer, **2019**, pp. 84-95.

[33] R. Sennrich, B. Haddow, A. Birch, Improving neural machine translation models with monolingual data, arXiv preprint arXiv:1511.06709, (**2015**).

[34] C. Coulombe, Text data augmentation made simple by leveraging nlp cloud apis, arXiv preprint arXiv:1812.04718, (**2018**).

[35] S. Kobayashi, Contextual augmentation: Data augmentation by words with paradigmatic relations, arXiv preprint arXiv:1805.06201, (**2018**).

[36] P. Liu, X. Qiu, X. Huang, Recurrent neural network for text classification with multi-task learning, arXiv preprint arXiv:1605.05101, (**2016**).

[37] Y. Luan, S. Lin, Research on text classification based on CNN and LSTM, 2019 IEEE international conference on artificial intelligence and computer applications (ICAICA), IEEE, **2019**, pp. 352-355.

[38] C.B. Do, A.Y. Ng, Transfer learning for text classification, Advances in neural information processing systems, 18 (**2005**).

[39] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up? Sentiment classification using machine learning techniques, arXiv preprint cs/0205070, (**2002**).

[40] B. Ohana, B. Tierney, Sentiment classification of reviews using SentiWordNet, (**2009**).

[41] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning Internal Representations by Error Propagation, Parallel Distributed Processing, Explorations in the Microstructure of Cognition, ed. DE Rumelhart and J. McClelland. Vol. 1. 1986, Biometrika, 71 (**1986**) 599-607.

[42] P. Baldi, Autoencoders, unsupervised learning, and deep architectures, Proceedings of ICML workshop on unsupervised and transfer learning, JMLR Workshop and Conference Proceedings, **2012**, pp. 37-49.

[43] W.H.L. Pinaya, S. Vieira, R. Garcia-Dias, A. Mechelli, Autoencoders, Machine learning, Elsevier**2020**, pp. 193-208.

[44] W. Wang, Y. Huang, Y. Wang, L. Wang, Generalized autoencoder: A neural network framework for dimensionality reduction, Proceedings of the IEEE conference on computer vision and pattern recognition workshops, **2014**, pp. 490-497.

[45] M. Sakurada, T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, **2014**, pp. 4-11.

[46] J. Zhai, S. Zhang, J. Chen, Q. He, Autoencoder and its various variants, 2018 IEEE international conference on systems, man, and cybernetics (SMC), IEEE, **2018**, pp. 415-419.

[47] D.P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114, (**2013**).

[48] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, Communications of the ACM, 63 (**2020**) 139-144.

[49] G.G. Sahin, M. Steedman, Data Augmentation via Dependency Tree Morphing for Low Resource Languages, 2018 Conference on Empirical Methods in Natural Language Processing, ACL Anthology, **2018**, pp. 5004-5009.

[50] G. Eryiğit, J. Nivre, K. Oflazer, Dependency parsing of Turkish, Computational Linguistics, 34 (**2008**) 357-389.

[51] G. Li, H. Wang, Y. Ding, K. Zhou, X. Yan, Data augmentation for aspect-based sentiment analysis, International Journal of Machine Learning and Cybernetics, 14 (**2023**) 125-133.

[52] V. Kumar, A. Choudhary, E. Cho, Data augmentation using pre-trained transformer models, arXiv preprint arXiv:2003.02245, (**2020**).

[53] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805, (**2018**).

[54] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, OpenAI blog, 1 (**2019**) 9.

[55] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension, Association for Computational Linguistics, Online, **2020**, pp. 7871-7880.

[56] K.M. Yoo, D. Park, J. Kang, S.-W. Lee, W. Park, GPT3Mix: Leveraging Large-scale Language Models for Text Augmentation, Findings of the Association for Computational Linguistics: EMNLP 2021, **2021**, pp. 2225-2239.

[57] Y. Wang, J. Zheng, C. Xu, X. Geng, T. Shen, C. Tao, D. Jiang, Knowda: All-in-one knowledge mixture model for data augmentation in few-shot nlp, arXiv preprint arXiv:2206.10265, (**2022**).

[58] N. Ng, K. Cho, M. Ghassemi, SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness, 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Association for Computational Linguistics (ACL), **2020**, pp. 1268-1283.

[59] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, Proceedings of the 25th international conference on Machine learning, **2008**, pp. 1096-1103.

[60] J.M.D. Delgado, L. Oyedele, Deep learning with small datasets: using autoencoders to address limited datasets in construction management, Applied Soft Computing, 112 (**2021**) 107836.

[61] F. Piedboeuf, P. Langlais, Effective data augmentation for sentence classification using one VAE per class, Proceedings of the 29th International Conference on Computational Linguistics, **2022**, pp. 3454-3464.

[62] K.M. Yoo, H. Lee, F. Dernoncourt, T. Bui, W. Chang, S.-g. Lee, Variational hierarchical dialog autoencoder for dialog state tracking data augmentation, arXiv preprint arXiv:2001.08604, (**2020**).

[63] S.J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Gallé, A. Raja, C. Si, W.Y. Lee, B. Sagot, Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP, arXiv preprint arXiv:2112.10508, (**2021**).

[64] M. Jimenez, C. Maxime, Y. Le Traon, M. Papadakis, On the impact of tokenizer and parameters on n-gram based code analysis, 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME), IEEE, **2018**, pp. 437-448.

[65] N. Rajaraman, J. Jiao, K. Ramchandran, Toward a Theory of Tokenization in LLMs, arXiv preprint arXiv:2404.08335, (**2024**).

[66] W.A. Qader, M.M. Ameen, B.I. Ahmed, An overview of bag of words; importance, implementation, applications, and challenges, 2019 international engineering conference (IEC), IEEE, **2019**, pp. 200-204.

[67] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781, (**2013**).

[68] S. Ma, X. Sun, Y. Wang, J. Lin, Bag-of-words as target for neural machine translation, arXiv preprint arXiv:1805.04871, (**2018**).

[69] J. Lilleberg, Y. Zhu, Y. Zhang, Support vector machines and word2vec for text classification with semantic features, 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC), IEEE, **2015**, pp. 136-140.

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

[71] Anonymous, Word2vec embeddings, https://radimrehurek.com/gensim/models/word2vec.html.(Access Date: **16.05.2024**)

[72] S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6 (**1998**) 107-116.

[73] Y. Yang, J. Wang, B. Wang, Prediction model of energy market by long short term memory with random system and complexity evaluation, Applied Soft Computing, 95 (**2020**) 106579.

[74] Y. Yu, X. Si, C. Hu, J. Zhang, A review of recurrent neural networks: LSTM cells and network architectures, Neural computation, 31 (**2019**) 1235-1270.

[75] M.A. Kramer, Nonlinear principal component analysis using autoassociative neural networks, AIChE journal, 37 (**1991**) 233-243.

[76] S. Odaibo, Tutorial: Deriving the standard variational autoencoder (vae) loss function, arXiv preprint arXiv:1907.08956, (**2019**).

[77] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, B. Frey, Adversarial autoencoders, arXiv preprint arXiv:1511.05644, (**2015**).

[78] T. Shen, J. Mueller, R. Barzilay, T. Jaakkola, Educating text autoencoders: Latent representation guidance via denoising, International conference on machine learning, PMLR, **2020**, pp. 8719-8729.

[79] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, D. Brown, Text classification algorithms: A survey, Information, 10 (**2019**) 150.

[80] B. Ghojogh, A. Ghodsi, Attention mechanism, transformers, BERT, and GPT: tutorial and survey, (**2020**).

[81] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108, (**2019**).

[82] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531, (**2015**).

[83] A. Anaby-Tavor, B. Carmeli, E. Goldbraich, A. Kantor, G. Kour, S. Shlomov, N. Tepper, N. Zwerdling, Do not have enough data? Deep learning to the rescue!, Proceedings of the AAAI conference on artificial intelligence, **2020**, pp. 7383-7390.

[84] R. Socher, A. Perelygin, J. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, Proceedings of the 2013 conference on empirical methods in natural language processing, **2013**, pp. 1631-1642.

[85] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv preprint arXiv:1711.05101, (**2017**).

[86] D.M. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, arXiv preprint arXiv:2010.16061, (**2020**).

APPENDIX A. EXAMPLE RECONSTRUCTIONS

Table A.1. Example sentence and its cleaned version.

Variant	Sentence	Predicted Label
Original Sentence	“this bond film goes off the beaten path , not necessarily for the better .”	0
Clean Sentence	“this bond film goes off the beaten path not necessarily for the better”	0

Note: In the subsequent tables (Table A.2, Table A.3, Table A.4, and Table A.5), the following conventions apply for the synthetic sample compositions based on augmentation ratios:

- The first element of each bullet list represents synthetic samples for an augmentation ratio of 1:1.
- The first two elements of each bullet list collectively represent synthetic samples for an augmentation ratio of 1:2.
- All four elements of each bullet list together represent synthetic samples for an augmentation ratio of 1:4.

Table A.2. Reconstructions from the selected DA methods, for example sentence in Table A.1 for dataset size of 100.

	Variant	Sentence
Dataset Size 100	aae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	aae-w2v-raw-100	• "it treats <unk> <unk> journey with honesty that is tragically rare in the depiction <unk> young women in film <unk>" • "<unk> be more he can cross swords with the best <unk> them <unk> helm <unk> more traditionally plotted popcorn thriller while surrendering little <unk> his intellectual rigor or creative composure <unk>" • "it treats <unk> <unk> journey with honesty that is tragically rare in the depiction <unk> young women in film <unk>" • "<unk> be more he can cross swords with the best <unk> them <unk> helm <unk> more traditionally plotted popcorn thriller while surrendering little <unk> his intellectual rigor or creative composure <unk>"
	ae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	ae-w2v-clean-100	• "i solemn pretension prevents off from beaten path not in the the the" • "i solemn pretension prevents us from sharing the awe awe which which holds" • "i solemn pretension prevents us from sharing the awe awe which which holds" • "i solemn pretension prevents us from sharing the awe awe which which holds holds"
	daae-token-clean-100	• "a fascinating curiosity piece fascinating that is for about ten minutes" • "a strong first act and absolutely inescapably gorgeous motion deliver the amazing" • "a strong first act and absolutely inescapably gorgeous motion deliver the amazing" • "this is a gorgeous film vivid that color music and life and again"
	daae-w2v-clean-50	• "<unk> proves <unk> lousy <unk> <unk> pic <unk> <unk> <unk><unk> have <unk> <unk> have dramatized the <unk> <unk> novel which itself felt like" • "<unk> <unk> excels <unk> lousy <unk> <unk> pic <unk> <unk> <unk> have <unk> <unk> have dramatized the <unk> <unk> novel which itself felt like" • "<unk> <unk> excels <unk> lousy <unk> <unk> pic <unk> <unk> <unk> have <unk> <unk> have dramatized the <unk> <unk> novel which itself felt like" • "<unk> <unk> excels <unk> lousy <unk> <unk> pic <unk> <unk> <unk> have <unk> <unk> have dramatized the <unk> <unk> novel which itself felt like"
	vae-token-clean-100	• "the bond film goes off the beaten path not necessarily for the better" • "the filmmakers know how to please the eye but not not not not the best the best the best" • "the filmmakers know how to please the eye but not not not not the best the best the best" • "the filmmakers know how to please the eye but not not not not the prettiest pictures that best the best"
	vae-w2v-raw-100	• "<unk> proves film goes off the beaten path <unk> not necessarily for the better <unk>" • "<unk> <unk> measured <unk> <unk> gently tedious in its comedy <unk> secret ballot is <unk> purposefully <unk> movie <unk> which may be why it <unk> so successful at lodging itself in the brain <unk>" • "<unk> <unk> measured <unk> <unk> gently tedious in its comedy <unk> secret ballot is <unk> purposefully <unk> movie <unk> which may be why it <unk> so successful at lodging itself in the brain <unk>" • "<unk> <unk> measured <unk> <unk> gently tedious in its comedy <unk> secret ballot is <unk> purposefully <unk> movie <unk> which may be why it <unk> so successful at lodging itself in the brain <unk>"

Table A.3. Reconstructions from the selected DA methods, for example sentence in Table A.1 for dataset size of 200.

	Variant	Sentence
Dataset Size 200	aae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	aae-w2v-raw-100	• "contradicts everything we they come <unk>" • "<unk> bond film goes off the beaten path <unk> not necessarily for the better <unk>" • "the pedestrian as they come <unk>" • "<unk> bond film goes off the beaten path <unk> not necessarily for the better <unk>"
	ae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	ae-w2v-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	daae-token-clean-100	• "playing only film goes off the beaten path not necessarily for the better" • "not only does deliver but i suspect it might deliver again and again" • "not only does deliver but i suspect it might deliver again and again" • "not only does deliver but i suspect it might deliver again and again"
	daae-w2v-clean-50	• "the <unk> never <unk> almost directly influenced this love story but <unk> <unk> <unk> making his directorial feature debut does strong measured work" • "the <unk> seems <unk> have directly influenced this love story but even more reassuring <unk> the" • "the <unk> seems <unk> have directly influenced this love story but even more reassuring <unk> the" • "the <unk> never <unk> have directly influenced this love story but even more reassuring <unk> can"
	vae-token-clean-100	• "the bond film goes off the beaten path not necessarily for the better" • "the bond film goes off the beaten path not necessarily for the better" • "the bond film goes off the beaten path not necessarily for the better" • "the bond film goes off the beaten path not necessarily for the better"
	vae-w2v-raw-100	• "contradicts everything best sports the best little ever <unk>" • "<unk> be more genial than ingenious <unk> but it gets the job done <unk>" • "the pathetic junk is movie best little ever seen <unk>" • "<unk> be more genial than ingenious <unk> but it gets the job done <unk>"

Table A.4. Reconstructions from the selected DA methods, for example sentence in Table A.1 for dataset size of 500.

	Variant	Sentence
Dataset Size 500	aae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	aae-w2v-raw-100	• "the bond film goes off the beaten path <unk> not necessarily for the better <unk>" • "the bond film goes off the beaten path <unk> not necessarily for the better <unk>" • "the bond film goes off the beaten path <unk> not necessarily for the better <unk>" • "the bond film goes off the beaten path <unk> not necessarily for the better <unk>"
	ae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	ae-w2v-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	daae-token-clean-100	• "this charming but slight tale has warmth wit and interesting characters compassionately portrayed" • "this charming but slight tale has warmth wit and interesting characters compassionately portrayed" • "this charming but slight tale has warmth wit and interesting characters compassionately portrayed" • "this charming but slight tale has warmth wit and interesting characters compassionately portrayed"
	daae-w2v-clean-50	• "the film <unk> pace is actually one <unk> its strengths" • "the <unk> the <unk> dynamite sticks built only controversy would recognize" • "the <unk> the <unk> dynamite sticks built only controversy would recognize" • "the <unk> the <unk> dynamite sticks built only controversy would recognize"
	vae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	vae-w2v-raw-100	• "the bodily function shines on all beaten path <unk> not necessarily for the better <unk>" • "the film brilliantly shines on all beaten path <unk> not necessarily for the better <unk>" • "the film brilliantly shines on all beaten path <unk> not necessarily for the better <unk>" • "the film brilliantly shines on all beaten characters <unk> as the for the better <unk>"

Table A.5. Reconstructions from the selected DA methods, for example sentence in Table A.1 for dataset size of 1000.

	Variant	Sentence
Dataset Size 1000	aae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	aae-w2v-raw-100	• "the movie is about the worst thing <unk> has done in the united states <unk>" • "<unk> has no affect the the their flaws <unk> <unk> heaven is one such beast <unk>" • "the result your sat scores are below slightly <unk> kids would quickly change the channel <unk>" • "<unk> has no affect the the energy <unk> <unk> but is one such beast <unk>"
	ae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	ae-w2v-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	daae-token-clean-100	• "the film is impressive for the sights and sounds of the wondrous beats the world has to offer" • "the film is impressive for the sights and sounds of the wondrous beats the world has to offer" • "the film is impressive for the sights and sounds of the wondrous beats the world has to offer" • "the film is impressive for the sights and sounds of the wondrous beats the world has to offer"
	daae-w2v-clean-50	• the <unk> <unk> <unk> fault <unk> flashy <unk> is <unk> <unk> relaxed <unk> displays" • "<unk> <unk> <unk> <unk> <unk> <unk> dim echo <unk> <unk> <unk>" • "<unk> <unk> <unk> <unk> <unk> <unk> dim echo <unk> <unk> <unk> • "<unk> <unk> <unk> make the oddest <unk> couples <unk> in this sense the movie <unk> <unk> <unk>
	vae-token-clean-100	• "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better" • "this bond film goes off the beaten path not necessarily for the better"
	vae-w2v-raw-100	• "the film is about the worst thing <unk> has done in the united states <unk>" • "<unk> film is not be by the flaws <unk> <unk> heaven is one such beast <unk>" • "the film is about the worst thing <unk> has done in the united states <unk>" • "<unk> film is not be by the flaws <unk> <unk> heaven is one such beast <unk>"