A Geometric and Topological View on Deep Learning Language Models

Sherry (Jia Hui) Feng

Supervisor: Prof. Edmund M-K. Lai

Dr. Weihua Li

School of Engineering, Computer and Mathematical Sciences

Auckland University of Technology

A thesis submitted in fulfilment of the requirement for the degree of

Doctor of Philosophy

2024

To my Mother, who has been my number one supporter throughout my whole life. I would not be who I am today without her.

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Sherry (Jia Hui) Feng
2024

Acknowledgements

What better way to handle a life crisis than by starting a PhD? This journey would not have been possible without the support and guidance of many.

My heartfelt gratitude goes to my supervisor, Prof. Edmund Lai, who believed in me when I was struggling at the beginning and took me on as a student. His initial assistance, continuous support, and unwavering guidance throughout this journey have been invaluable. I am also immensely grateful to my secondary supervisor, Dr. Weihua Li, for his insightful discussions and valuable advice. Thank you both – I truly could not have asked for a better supervisory team.

Thank you to my family, who encouraged me as the first person in our family to go to university, and to my friends, who were there for me when I needed them most. You all know who you are.

Matthew, this chapter of my life would not have been possible without you. I am incredibly lucky to have you by my side.

To everyone who has been a part of this journey, thank you. Your support, encouragement, and love have made all the difference.

Abstract

As Natural Language Processing (NLP) models evolve, challenges such as interpretability, misinformation, and high computational demands persist, primarily due to limited understanding of their decision-making processes. This thesis aims to uncover crucial insights by examining NLP models through the frameworks of topology and geometry, addressing both the understanding and practical aspects of these challenges.

Through our topological framework analysis, we demonstrate, for the first time, the occurrence of neural collapse in NLP models for text classification tasks, revealing how features within each class collapse towards their mean while maintaining separation between different classes. Our geometric analysis, using convex hull computations and Delaunay triangulation, uncovered that high-performing language models maintain distinct semantic boundaries and demonstrate consistent geometric properties across different scales of analysis.

Building on these insights, we developed two novel algorithms: GATFilter and GATA. GATFilter improved augmented data quality, achieving performance gains of up to 8% across various NLP datasets (SST2, SNIPS, TREC, Question Topic) and multiple augmentation strategies (EDA, Backtranslation, Contextual Augmentation using BERT). GATA demonstrated exceptional performance on similar datasets while maintaining significant computational efficiency. GATA showed non-linear scaling in execution time (54.25 to 167.58 seconds from 20 to 80 samples) and avoided the large initial memory loads (>3900MB) required by methods like Backtranslation and ContextualBERT.

These findings advance NLP theory by establishing new geometric frameworks for analyzing word embeddings and model behavior, while offering practical solutions for resource-efficient text augmentation. Our work demonstrates that geometric principles can effectively balance model performance with computational efficiency.

Publications

- Feng, J.H. (2022). *Can Ready-to-Use RNNs Generate "Good" Text Training Data? International Journal of Advanced Computer Science and Applications*, Vol. 13, No. 3, pp. 53-61. DOI: 10.14569/IJACSA.2022.0130306
- Feng, J.H., Lai, E.M-K., Li, W. (2023). *A Study of Neural Collapse for Text Classification.* In: Conte, D., Fred, A., Gusikhin, O., Sansone, C. (eds) *Deep Learning Theory and Applications. DeLTA 2023.* Communications in Computer and Information Science, Vol. 1875, pp. 149-169. Springer, Cham. DOI: 10.1007/978-3-031-39059-3_9
- Feng, J.H., Lai, E.M-K., Li, W. (2024). *A Geometric Approach to Textual Augmented Data Filtering. Journal of Physics: Conference Series*, Vol. 2833, No. 1, Article 012007. IOP Publishing. DOI: 10.1088/1742-6596/2833/1/012007
- Feng, J.H., Lai, E.M-K., Li, W. (2024). *Geometry of Textual Data Augmentation: Insights from Large Language Models. Electronics*, Vol. 13, No. 18, Article 3781. DOI: 10.3390/electronics13183781

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Natural Language Processing (NLP) is a multidisciplinary field at the intersection of linguistics, computer science, and artificial intelligence. Its focus is on the processing of human languages by computers. By enabling machines to process and generate natural language, NLP opens up a wide range of possibilities for enhancing human-computer interaction and automating language-related tasks. This capability is increasingly crucial in our information-driven society. The importance of NLP is evident across multiple sectors. In business, it powers chatbots and virtual assistants for customer service (Hoy, 2018). Healthcare applications include the processing of clinical notes and medical literature (Demner-Fushman et al., 2009), while in education, NLP enables personalized learning systems (McNamara et al., 2013). Key applications include machine translation (Wu et al., 2018), improving global communication; information retrieval for search engines (Manning et al., 2008); and sentiment analysis to gauge public opinion (Liu and Zhang, 2012). Emerging applications extend to natural language interfaces for autonomous vehicles, voice control for smart home systems, and contract analysis and research for the legal sector (Ghosh et al., 2023).

Recent advancements in applying deep learning techniques to NLP are enabled by the development of models that embed text as numerical vectors in a high-dimensional space. Models like Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014) are two of the more popularly used ones, capturing semantic relationships and meanings at the word level. These word embedding models are grounded in distributional semantics theory, which posits that words appearing in similar contexts tend to have similar meanings (Harris, 1954). This theoretical underpinning allows us to understand and predict how these models capture word similarities, analogies, and other semantic properties.

However, as NLP evolves to tackle more complex tasks such as context-aware understanding, sentiment analysis, and conversational AI, the models have also grown in sophistication along with arising issues.

1.1 Challenges

1.1.1 Understanding Model Behaviour and Learning Dynamics

As NLP advances to more complex language understanding tasks using sophisticated neural architectures, the theoretical foundation of NLP models becomes increasingly opaque (Doshi, 2018; Guidotti et al., 2018). While models like BERT (Devlin et al., 2019), GPT (Radford et al., 2019), and other large language models (LLMs) demonstrate impressive performance, our understanding of why and how these models function still lags behind their capabilities. Firstly, there is an issue with *model opacity*. The inner workings of complex neural network-based NLP models often remain opaque, leading to issues such as the black box problem (Doshi-Velez and Kim, 2017), difficulties in bias detection and mitigation, and a trade-off between model performance and interpretability (Bolukbasi et al., 2016). This lack of understanding hampers our ability to fully trust and deploy these models in critical applications where understanding the reasoning behind decisions is crucial.

This leads to another issue – *ad hoc development practices*. The pace of advancement has led to a reliance on empirical, trial-and-error approaches rather than theoretically grounded methods. This results in limited generalizability (McCoy et al., 2019), difficulties in predicting performance across different domains or tasks (Jia and Liang, 2017), and a lack of principled improvement strategies (Linzen, 2020). The absence of a strong framework in analyzing model behaviour often leads to incremental improvements based on empirical results rather than fundamental breakthroughs in our understanding of language processing.

Such challenges underscore the critical need for a more robust analytical framework in NLP. We require approaches that can address multiple facets of the current limitations in the field. Specifically, we need methods that can elucidate the inner workings of complex models, providing insights into their decision-making processes and internal representations. For tasks where there is a lack of training data, these approaches should offer insight into effective data augmentation techniques, allowing us to leverage them more effectively and predictably. Furthermore, the

framework should provide principled strategies for model improvement, guiding researchers and practitioners in enhancing model performance while maintaining efficiency and interpretability.

1.1.2 High Resource Requirements

While LLMs have demonstrated remarkable capabilities in generating high-quality, contextually relevant data for augmentation, their use comes with significant computational and resource costs (Strubell et al., 2019). This challenge manifested a couple of critical issues.

The first one is the issue of *resource requirements and accessibility*. The size and complexity of these models often necessitate specialized, high-end hardware, such as graphical processing units (GPUs) or tensor processing units (TPUs), which are both expensive and not universally available (Schwartz et al., 2020). This creates a disparity in the field, potentially limiting full participation in cutting-edge NLP research and development (Costa et al., 2024). Such resource-intensive requirements could lead to an uneven distribution of AI capabilities, potentially impacting the diversity of approaches and innovations in the field.

The second issue is *energy consumption*. The energy consumption associated with training and running LLMs is substantial, raising concerns about their environmental impact (Strubell et al., 2019). This high energy footprint not only increases operational costs but also contributes to the carbon footprint of NLP research and applications.

Given these resource-related challenges, there is a pressing need for high-performing NLP models that can operate with resource efficiency. These models should aim to achieve competitive performance with significantly reduced computational requirements, bridging the gap between resource-intensive state-of-the-art models and practical, widely deployable solutions. Developing such resource-efficient alternatives is crucial for democratizing access to advanced NLP capabilities, enabling their use in computationally constrained environments, fostering innovation across a broader spectrum of researchers and practitioners, and reducing the environmental impact of NLP technologies.

In this thesis, *resource efficiency* refers to models that can achieve competitive performance with significantly reduced computational requirements. Specifically:

1. **Reduced computing power requirements**: Models that can be trained and deployed on less powerful hardware, making them accessible to a broader range of users and organizations.

2. **Reduced data requirements**: Models that perform well with smaller datasets, reducing the need for extensive data collection and annotation.

1.2 Research Approach

The challenges outlined above are deeply intertwined – the lack of a robust theoretical foundation contributes to the black box nature of NLP models, often leading to ad hoc solutions and resource-intensive approaches. By developing a stronger basic understanding, this research aims to address multiple challenges simultaneously, contributing to more explainable and efficient NLP systems.

The main aim of this research is to gain insights and understanding into the effects of training data in the learning process of text classification models in a low-resource context. This is approached from a geometric and topological perspective.

Topological methods can reveal global structural properties of the high-dimensional spaces in which NLP models operate, potentially uncovering hidden patterns and relationships. In particular, persistent homology methods have been used in various machine learning contexts to study data manifolds and have shown promise in understanding the structure of word embeddings (Carlsson, 2009; Draganov and Skiena, 2024). In this thesis, it is used to gain insights that lead to more efficient model architectures and improved generalization capabilities.

Computational geometric analyses provide insights into the local relationships between words and concepts within these spaces, offering a more nuanced understanding of how models process language. It is supported by research that applies geometric perspectives to word embeddings, enabling a deeper understanding of their internal structure and the relationships they encode (Nickel and Kiela, 2017; Pennington et al., 2014).

The challenging task of data augmentation for text classification has been chosen as a vehicle through which the research objective stated above is to be acheived. This choice is motivated by three key factors. Firstly, it is to *highlight contextual understanding*. Data augmentation techniques, particularly those involving word or phrase substitutions, highlight how machines interpret meaning contextually. For instance, consider the sentence:

This is good

is a positive intent sentence. However, swapping two words around, it becomes:

is This good

which is questioning if "this" is of any good, a completely different meaning. Putting the same words in different orders, demonstrate how subtle changes can dramatically alter meaning. By studying how models handle such augmentations, we can gain insights into their understanding of language structure and semantics.

The second factor is that it helps to *bridge theory and practise*. Data augmentation serves as an excellent bridge between theoretical concepts and practical model improvements. By systematically altering input data and observing the effects on model performance, theories about language representation and processing in NLP models can be developed and tested.

Lastly, it *addresses resource intensity*. Effective data augmentation techniques have the potential to improve model performance without necessarily increasing model size or computational requirements. This aligns with objective (in 1.3.2), which is to develop more efficient NLP systems that maintain high performance while reducing resource intensity.

This research aims to iterate between theory and practice, using the performance of developed algorithms to refine and validate the theoretical framework. By doing so, we hope to contribute to the development of NLP systems that are not only powerful but also explainable, efficient, and sustainable. A detailed review of the related techniques and concepts applied in NLP are provided in Chapter 2.

1.3 Research Questions and Objectives

This research is guided by two main objectives, each addressing a specific challenge identified earlier.

1.3.1 Objective 1

The first objective is to establish geometric and topological frameworks for analyzing NLP model behaviours. It aims to examine the inner workings of NLP models through:

- **Physical Structure Analysis:** Investigating the architectural properties of neural networks processing textual data.
- **Learning Process and Training Data Examination:** Focusing on data augmentation techniques and their impact on model behaviour.

The research questions associated with this objective are as follows.

RQ1 What insights do topological approaches provide into the inner workings and physical structure of deep neural networks processing textual data?

RQ2 What do topological and geometric analyses reveal about the mechanisms by which NLP models interpret meaning and learn from training data?

RQ3 What are the key differences in topological and geometric properties between high-performing LLMs and poor-performing models in data augmentation tasks?

1.3.2 Objective 2

Making use of the insights gained from Objective 1, the second objective aims to develop NLP algorithms that are both computational and resource efficient for data augmentation. It seeks to answer the following research questions.

RQ4 To what extent do the data augmentation algorithms developed based on geometric principles improve model performance while maintaining computational efficiency?

RQ5 What quantifiable performance differences exist between the developed algorithms and existing methods across standard NLP benchmarks?

RQ6 In what ways do the computational resource requirements of the developed algorithms differ from those of resource-intensive LLMs, particularly in terms of time and memory?

1.4 Thesis Structure and Summary

Figure 1.1 shows the structure of this thesis and relates the contents of the chapters to the research questions.

Chapter 2 provides a review of related literature. It also contains the results of a preliminary experiment using Recurrent Neural Networks for generating augmentation data. Together, they highlight the need for a better understanding of NLP models to develop more robust and effective data augmentation methods.

Chapter 3 addresses **RQ1** by investigating the phenomenon of neural collapse in text classification models using Topological Data Analysis. This chapter makes a significant contribution by showing that, for the first time, neural collapse occurs in NLP models. The findings reveal how the convergence of features to a simplified structure provides insights into the physical behaviour of neural networks processing textual data, and how topological approaches can uncover hidden

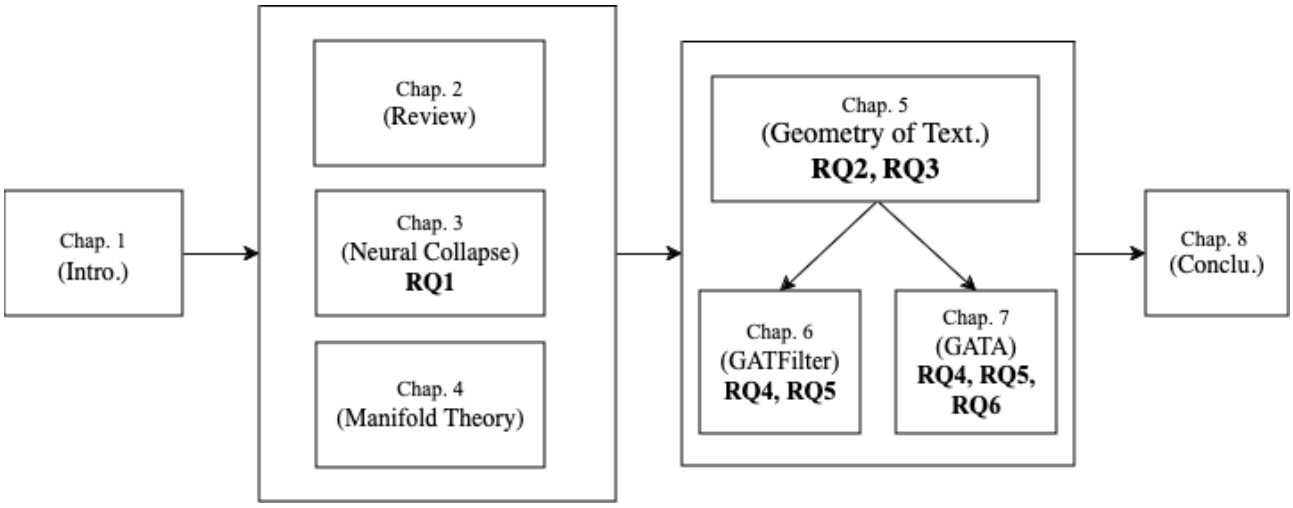


Fig. 1.1 Thesis Structure

structures within the data that are not apparent from the labels alone. These insights lay the foundation for developing more interpretable NLP models.

In Chapter 4, it is shown that reducing high-dimensional word embeddings to just two principal components can preserve essential meaning. This finding not only answers a fundamental question about language representation but also establishes the dimensionality reduction approach used in Chapters 5, 6, and 7. These chapters present an exploration of the geometric and topological properties of text representation.

Chapter 5 addresses **RQ2** and **RQ3** by conducting a comprehensive geometric analysis of effective augmentation data. This chapter contributes novel insights into the geometric characteristics of high-quality augmented text, revealing that effective augmentations maintain specific geometric relationships with the original data. It demonstrates that high-performing LLMs generate augmentations that remain within the convex hull of the original data, while poor-performing models often produce points outside these boundaries. These findings provide a geometric explanation for the success of certain augmentation methods over others.

Building on these insights, Chapter 6 introduces GATFilter, a practical algorithm designed to filter out augmented text data that is not potentially useful for training text classifiers. This chapter addresses **RQ4** and **RQ5** by demonstrating how geometric principles can be applied to improve the quality of augmented datasets. GATFilter achieves significant performance improvements (2-8%) across various NLP datasets and augmentation strategies, while maintaining computational efficiency compared to existing methods.

Lastly, Chapter 7 presents GATA, an algorithm that extends GATFilter to generate effective augmented text data. This chapter fully addresses **RQ4**, **RQ5**, and **RQ6** by demonstrating how

geometry-based approaches can not only improve classification accuracy but also significantly reduce computational requirements compared to state-of-the-art methods, including resource-intensive language models. GATA achieves competitive performance on benchmark datasets while requiring substantially less memory and computational resources than methods like TinyBERT, addressing the critical need for resource-efficient NLP systems.

Chapter 8 concludes the research presented in this thesis and offers some suggestions for future work.

Chapter 2

From Words to Spaces: A Review of NLP Models and Mathematical Approaches

In Chapter 1, we identified two significant challenges in the field of NLP: **limited analytical frameworks for model behavior and the high resource intensity of current models**. These issues not only hinder the interpretability and generalizability of NLP models but also limit their accessibility due to the significant computational resources required. In this chapter[1], we provide a literature review of the current status and challenges. We then explore how integrating topological and geometric methods into NLP can offer novel solutions to these challenges, providing both systematic mathematical frameworks for analysis and potential pathways for more efficient model development.

2.1 Natural Language Processing

The roots of NLP can be traced back to the 1950s, with early work focusing on machine translation (Weaver, 1952). Over the decades, NLP has evolved through several paradigms:

Rule-based systems (1950s-1980s)

These systems relied on hand-crafted linguistic rules and were the dominant approach in the early days of NLP (Chomsky, 1953). Chomsky's work on formal language theory and generative

[1]The preliminary experiment contents of this chapter are based on the author's publication *Can RNNs Generate Good Textual Training Data?* cited on Page-*iv*.

grammar provided a foundation for many rule-based systems. Notable examples include ELIZA, one of the first chatbots, which used pattern matching and predetermined scripts to simulate conversation (Weizenbaum, 1966), and the LUNAR system, developed for querying a database about moon rocks, which demonstrated the potential of natural language interfaces (Woods, 1972). Rule-based machine translation systems like SYSTRAN were among the first attempts at automated translation (Toma, 1977). While powerful for specific domains, these systems ultimately struggled with the ambiguity and complexity of natural language (Hayes, 1981), leading researchers to explore alternative approaches.

Statistical NLP (1980s-2000s)

This paradigm introduced probabilistic models and machine learning techniques, marking a significant shift from rule-based approaches (Manning, 1999). IBM's work on statistical machine translation in the late 1980s was groundbreaking, using large parallel corpora to learn translation probabilities (Brown et al., 1990). Hidden Markov Models (HMMs) became popular for tasks like part-of-speech tagging and speech recognition (Rabiner, 1989). The development of large annotated corpora, such as the Penn Treebank, enabled supervised learning approaches (Marcus et al., 1993). Maximum Entropy models provided a flexible framework for incorporating diverse features in NLP tasks (Berger et al., 1996), while techniques like the Expectation-Maximization (EM) algorithm allowed for unsupervised learning from raw text (Dempster et al., 1977). Statistical approaches proved more robust to the variability of natural language compared to rule-based systems, setting the stage for further advancements in the field.

Neural NLP (2010s)

This current paradigm leverages deep learning and neural networks, leading to unprecedented performance on various NLP tasks (Goldberg, 2022). Word embeddings, such as Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014), revolutionized word representation by capturing semantic relationships in dense vector spaces of neural networks. For instance, recurrent neural networks, particularly Long Short-Term Memory (LSTM) networks, became popular for sequence modeling tasks (Hochreiter and Schmidhuber, 1997).

Recent Advances (present)

The introduction of attention mechanisms (Bahdanau et al., 2015) and subsequently the Transformer architecture (Vaswani et al., 2017) led to significant improvements in machine translation and other NLP tasks. Transfer learning, exemplified by models like Embeddings from Large Models (ELMo) (Peters et al., 2018a), Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), and Generating Pre-training Transformer (GPT)

(Radford et al., 2019), allowed models to leverage knowledge from large-scale unsupervised pre-training. Recent LLMs like GPT-3 (Brown et al., 2020) have demonstrated remarkable few-shot learning capabilities across a wide range of NLP tasks. While neural approaches have consistently outperformed previous methods on benchmarks, sometimes achieving human-level performance (Wang et al., 2019). However, despite these remarkable advances, significant challenges remain.

2.2 Fundamental Principles of Word Representation

As discussed in Chapter 1, the theoretical underpinnings of modern NLP models, particularly LLMs, remain opaque. This lack of a clear framework which manifests into various issues such as the predominantly empirical approach to NLP models, where techniques are often developed and refined through trial and error. This section explores the background of word embeddings, demonstrating how mathematical approaches have already proven their worth in capturing the complexities of language. This exploration will not only provide context for our proposed methods but also strengthen the rationale behind applying topological and geometric analyses to data augmentation in NLP.

To fully grasp the power and limitations of LLMs in data augmentation, it is essential to understand how these models represent and process language at a fundamental level. At the core of LLMs lies the concept of word representations, which has evolved significantly over the years. By examining the journey from traditional approaches to modern embedding techniques, we can gain valuable insights into how LLMs capture and manipulate linguistic information.

In this section, we explore the evolution of word representation techniques, from simple one-hot encodings to the sophisticated embedding methods employed in state-of-the-art LLMs. This exploration will provide the foundation for our subsequent analysis of how topological and geometric approaches can enhance our understanding and utilization of LLM-based data augmentation.

2.2.1 Traditional Approaches

Early approaches to word representation in NLP relied on simple, discrete encoding methods. One of the most basic was the one-hot encoding, where each dimension corresponds to a unique word in the vocabulary, with the dimension for the specific word set to 1 and all other dimensions set to 0 (Jurafsky, 2000).

Consider the sentence: 'The cat chased the dog.' Assuming a vocabulary derived from this sentence, the one-hot encodings for "cat" and "dog" would be:

Vocabulary: [the, cat, chased, dog]

$$\mathbf{v}_{\text{cat}} = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}^\top$$

$$\mathbf{v}_{\text{dog}} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^\top$$

Here, the word 'cat' is represented as a vector where the second position (corresponding to 'cat') is set to 1, and all other positions are set to 0. Similarly, the word "dog" is represented with the fourth position set to 1.

Another traditional approach was the bag-of-words (BoW) model. The BoW model represents a text as an unordered collection of words, disregarding grammar and word order (Harris, 1954). In BoW, a document is represented as a vector where each dimension corresponds to a word in the vocabulary, and the value is typically the count of that word in the document. For instance:

Document: "The cat sat on the mat"

$$\mathbf{v}_{\text{doc}} = \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \end{bmatrix}^\top$$

where the vector represents the counts for [the, cat, sat, on, mat].

Building upon the BoW model, TF-IDF (Term Frequency-Inverse Document Frequency) weighs terms based on their importance in a document and corpus (Sparck Jones, 1972). The TF-IDF score for a term t in document d is calculated as:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

where:

$$\text{TF}(t, d) = \frac{\text{count of } t \text{ in } d}{\text{total words in } d}$$

$$\text{IDF}(t) = \log \frac{\text{total documents}}{\text{documents containing } t}$$

N-grams extend the idea of such traditional methods by considering sequences of N words. For example, bigrams ($N = 2$) for the sentence 'The cat sat on the mat' would be:

Bigrams: ['The cat', 'cat sat', 'sat on', 'on the', 'the mat']

This approach can capture some local context but still struggles with long-range dependencies (Brown et al., 1992).

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) applies singular value decomposition (SVD) to the term-document matrix to reduce its dimensionality. This technique can uncover latent semantic structures in the data, partially addressing the synonymy and polysemy problems of simpler methods.

The LSA algorithm involves the following steps:

1. Construct a term-document matrix A.
2. Apply SVD: $A = U\Sigma V^T$.
3. Reduce dimensionality by keeping only the k largest singular values.

These methods, while useful for certain tasks, are limited in capturing semantic relationships and dealing with high-dimensional, sparse representations (Bengio and Senécal, 2003). To address these issues, researchers have developed word embedding techniques that learn dense, continuous vector representations of words from large text corpora. In the next section, we will explore the current state of word embeddings.

Word Embeddings

The introduction of word embeddings marked a significant advancement in word representation. Word embeddings are dense vector representations of words in a continuous vector space, where semantically similar words are mapped to nearby points (Mikolov et al., 2013b).

Word2Vec

Word2Vec, introduced by Mikolov et al. (2013b), uses neural networks to learn word vectors. It comes in two architectures:

1. Continuous Bag-of-Words (CBOW): Predicts a target word from its context words.
2. Skip-gram: Predicts context words given a target word.

The key innovation of Word2Vec is its ability to capture semantic and syntactic relationships between words. For example, the famous analogy:

$$\text{vector(king)} - \text{vector(man)} + \text{vector(woman)} \approx \text{vector(queen)}$$

This demonstrates that the learned embeddings encode meaningful semantic relationships (Mikolov et al., 2013c). Word2Vec embeddings are typically 100-300 dimensions.

GloVe

GloVe, developed by Pennington et al. (2014), combines the advantages of global matrix factorization and local context window methods. Unlike Word2Vec, which is a predictive model, GloVe is a count-based model that works on co-occurrence statistics.

The GloVe model is formulated as a weighted least squares problem:

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(\mathbf{w}_i^T \tilde{\mathbf{w}}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2$$

Where X_{ij} is the co-occurrence count between words i and j, $\mathbf{w}_i$ and $\tilde{\mathbf{w}}_j$ are word vectors, and b_i and $\tilde{b}_j$ are bias terms.

GloVe embeddings typically range from 50 to 300 dimensions, balancing the trade-off between computational efficiency and capturing meaningful semantic relationships.

FastText

FastText, introduced by Bojanowski et al. (2017), extends the Word2Vec model by representing each word as a bag of character n-grams. This approach allows the model to generate embeddings for out-of-vocabulary words and often performs better for morphologically rich languages.

The FastText model represents a word w as:

$$\text{s}(w, c) = \sum_{g \in G_w} \mathbf{z}_g^T \mathbf{v}_c$$

Where G_w is the set of n-grams in w, $\mathbf{z}_g$ is the vector representation of n-gram g, and $\mathbf{v}_c$ is the context vector. FastText embeddings are typically 100-300 dimensions, similar to Word2Vec, allowing for a rich representation of words.

2.2.2 Neural network architectures for NLP

The field of NLP has seen decent advancements with the introduction of various neural network architectures. These architectures have been instrumental in capturing the complexities of language and achieving state-of-the-art performance on numerous NLP tasks.

Recurrent Neural Networks (RNNs): RNNs, particularly LSTM networks (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRUs) (Cho et al., 2014), were among the first neural architectures to achieve significant success in NLP tasks. These models are designed to process sequential data, making them well-suited for tasks like language modeling and machine translation. However, RNNs face challenges in capturing long-range dependencies due to the vanishing gradient problem (Bengio et al., 1994).

Convolutional Neural Networks (CNNs): While initially developed for computer vision tasks, CNNs have been successfully adapted for NLP (Kim, 2014). They excel at capturing local patterns in text, making them effective for tasks such as text classification and sentiment analysis. CNNs can efficiently process text in parallel, overcoming some of the sequential processing limitations of RNNs.

Transformer Architecture: The introduction of the Transformer architecture by Vaswani et al. (2017) marked a significant milestone in NLP. Transformers rely entirely on attention mechanisms, eliminating the need for recurrence and convolutions. This architecture has become the foundation for many state-of-the-art models, including BERT (Devlin et al., 2019) and GPT (Radford et al., 2019). Transformers excel at capturing long-range dependencies and can be efficiently trained on large datasets.

Graph Neural Networks (GNNs): GNNs have gained traction in NLP for tasks that involve structured data or require modeling relationships between entities (Wu et al., 2020). They have been applied to tasks such as semantic role labeling, machine translation, and question answering, where understanding the relational structure of language is crucial.

2.2.3 Learning Paradigms

The success of neural network architectures in NLP is closely tied to the learning paradigms employed. These paradigms determine how models acquire knowledge from data and how they generalize to new, unseen examples.

Supervised Learning: Supervised learning remains a dominant paradigm in NLP, where models are trained on labeled datasets to learn a mapping from inputs to outputs (Goodfellow

et al., 2016). This approach has been successful in tasks such as text classification, named entity recognition, and machine translation. However, the reliance on large labeled datasets can be a limitation, especially for low-resource languages or specialized domains.

Unsupervised Learning: Unsupervised learning techniques aim to discover hidden structures in unlabeled data (Bengio et al., 2013). In NLP, this paradigm has been particularly useful for tasks such as topic modeling (Blei et al., 2003) and word embedding learning (Mikolov et al., 2013b). Unsupervised learning allows models to leverage large amounts of unlabeled text data, which is often more readily available than labeled data.

Transfer Learning: Transfer learning has revolutionized NLP in recent years (Ruder, 2019). This paradigm involves pre-training models on large, general-purpose datasets and then fine-tuning them on specific downstream tasks. Models like BERT (Devlin et al., 2019) and GPT (Radford et al., 2019) have demonstrated the power of this approach, achieving state-of-the-art results across a wide range of NLP tasks with minimal task-specific fine-tuning.

Few-Shot and Zero-Shot Learning: As an extension of transfer learning, few-shot and zero-shot learning paradigms aim to perform well on new tasks with very few or no labeled examples (Brown et al., 2020). These approaches leverage the knowledge acquired during pre-training to generalize to new tasks, potentially reducing the need for large task-specific datasets.

Self-Supervised Learning: Self-supervised learning has emerged as a powerful paradigm in NLP (Liu et al., 2021). This approach involves creating supervised learning tasks from unlabeled data, often by masking or corrupting the input and training the model to reconstruct it. Models like BERT use masked language modeling, a self-supervised task, during pre-training. This paradigm allows models to learn rich representations from large amounts of unlabeled text data.

The combination of advanced neural architectures and these learning paradigms has significantly advanced NLP performance. However, the increasing complexity and computational demands of these models, particularly with the advent of LLMs, have also introduced new challenges. In scenarios with limited resources or specific domain constraints, these resource-intensive models may not always be the most practical solution. This challenge drives the need to explore alternative, more efficient approaches.

To address this, we turn our attention to RNNs, which, despite being simpler and less resource-intensive than LLMs, have shown potential in various NLP tasks. The question remains, however, whether these smaller models can effectively generate high-quality augmented

data. To investigate this, we conducted a preliminary experiment to assess the viability of using pre-configured RNNs for textual data augmentation.

2.3 Preliminary Study: Assessing Resource-Efficient Generative Neural Network Effectiveness

Given the resource-intensive nature of LLMs, it becomes imperative to explore more efficient alternatives for data augmentation in NLP. While LLMs offer state-of-the-art performance, their computational demands often outweigh their benefits, especially in scenarios with limited resources or time constraints. This necessity prompts us to investigate simpler, more lightweight models that could potentially offer a balance between efficiency and effectiveness in data augmentation tasks. RNNs, with their ability to process sequential data and generate text, present themselves as a promising candidate for this purpose. RNNs are generally smaller in size compared to LLMs, require less computational power, and have been successfully used in various NLP tasks (Cho et al., 2014; Sutskever et al., 2014). However, their effectiveness in generating high-quality augmented data for NLP tasks (at the time of this study) remains an open question. To address this, we conducted a preliminary study to assess the viability of using pre-configured RNNs for textual data augmentation.

2.3.1 Experiment Setup

Dataset

The dataset used was AG News which consists of four different classes, 1–4, where 1 is world, 2 is sports, 3 is business and 4 is science/technology.

We first established baseline accuracy rates using a CNN model trained on small, medium, and large subsets of the original data. The dataset was split into 80% for training and 20% for validation, ensuring each label had an equal number of sentences in the training sets and a mix in the validation sets. The datasets were divided into three different sizes:

Dataset Size	Training Sentences	Validating Sentences	Test Sentences
Small	50	40	10
Medium	500	400	100
Large	1000	800	200

Table 2.1 Dataset Sizes and their respective Training, Validating, and Test sentences

Each dataset generated 200 lines of new training data, with an equal number of labels in the training sentences. For instance, in the "medium" training dataset of 400, there were 100 sentences from each of the four labels. The sizing is relative to this experiment.

Generative Neural Network

We used an RNN based on the works of (Woolf, 2017) to generate additional synthetic training data for each subset. The RNN embeddings pass through two 128-cell LSTM layers. An attention layer then weights and averages the outputs from both LSTM layers and the embeddings. The architecture is illustrated in Figure 2.1.

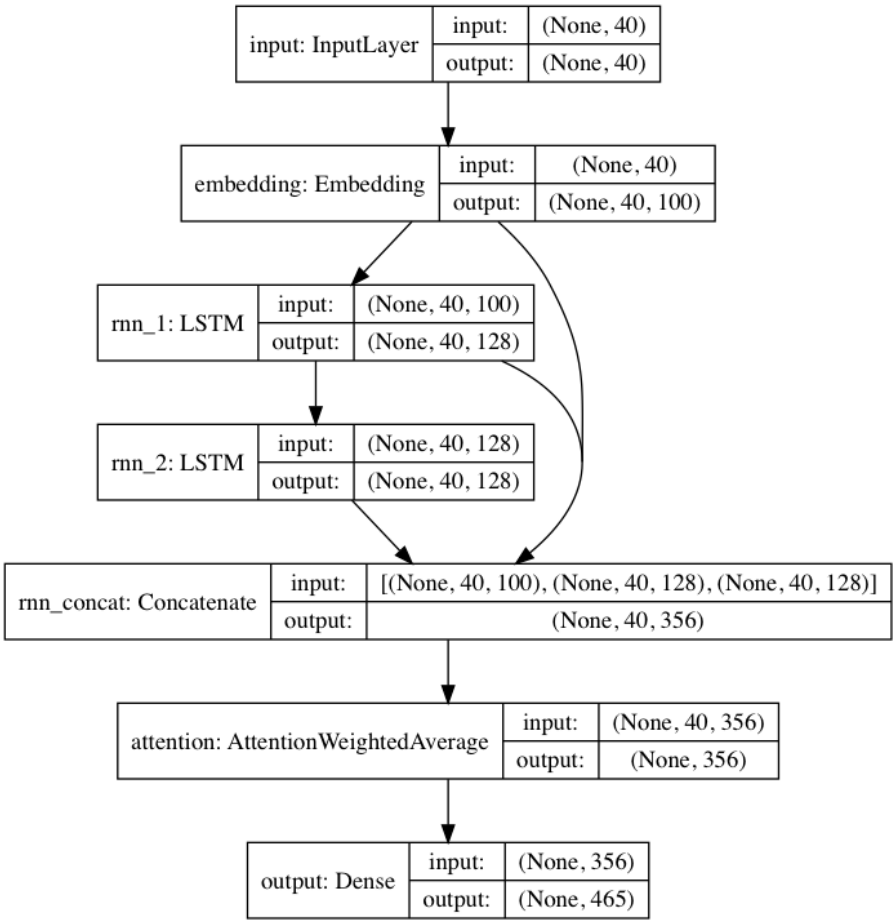


Fig. 2.1 RNN Generative Model (Woolf, 2017)

Classification Model: A CNN model was trained on the baseline dataset and then on the combined original and synthetic data to evaluate the impact on classification accuracy. The CNN architecture is as follows:

- Input: Up to 50 words
- 1D convolutional layer with 128 filters of size 5
- ReLU activation function

- Softmax output layer
- Word Embedding: GloVe (100 dimension)

2.3.2 Results and Discussion

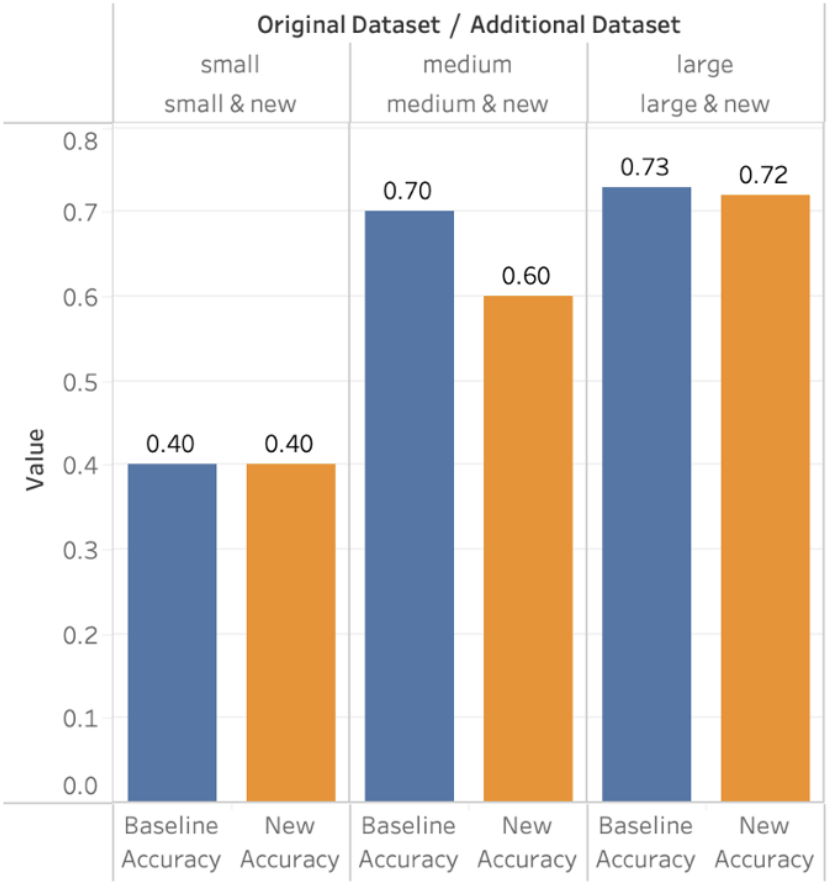


Fig. 2.2 Accuracy Results Visualized

```
Southern Sunday hit the Brate Star victory and now skin.
```

Fig. 2.3 Generated sample from Label 2

The results showed that for the small dataset, the accuracy remained the same after augmentation. For the medium dataset, the accuracy decreased, while for the large dataset, the augmented model nearly matched the baseline accuracy. However, upon closer inspection of the generated text with an example shown in Figure 2.3, we observed that while the synthetic examples generally stayed on-topic, there were still notable differences compared to the original data. This raises questions about the characteristics of the generated words that affect the

CNN's performance and whether specific RNN parameters could be tuned for different types of text data.

While this experiment aimed to address the computational cost and model size issues associated with LLMs, it ultimately highlighted several crucial points. In terms of computational efficiency, the RNN model used for our augmentation required significantly fewer resources than state-of-the-art LLMs. Specifically, our RNN implementation had approximately 2-5 million parameters, compared to modern LLMs, which often exceed billions of parameters (e.g., GPT-3 with 175 billion parameters) (Brown et al., 2020). This translated to practical advantages including the ability to train on standard CPU hardware without requiring specialized GPUs, and a memory footprint of only a few hundred megabytes compared to several gigabytes for LLMs (Strubell et al., 2019). Despite these computational advantages, our study identified:

- The need for a robust analytical framework to explain and improve data augmentation in NLP.
- The importance of data quality over quantity in augmentation techniques.

The results of our preliminary study with RNNs highlight a crucial issue that extends beyond just the choice of model for data augmentation. While RNNs offer a more computationally efficient alternative to LLMs, their inconsistent performance across different dataset sizes points to a deeper, more fundamental challenge in the field of data augmentation for NLP. This challenge is not unique to RNNs but permeates through various data augmentation techniques, including those utilizing LLMs. It emphasizes the manifestation of adhoc experimentation in current research.

Even those leveraging powerful LLMs, leaves us with significant gaps in our understanding of why and how these techniques work. This realization motivates our decision to investigate language models through the lens of Topological Data Analysis and geometric approaches, aiming not only to unravel the reasons behind their superior performance but also to address the efficiency concerns highlighted. The next sections will provide a review of the current state of explainability split by the physical and learning aspects of NLP models.

2.4 Understanding and Explaining NLP Models: Current Methods and Challenges

As NLP models grow in complexity and capability, the need for explainable AI and robust theoretical foundations becomes increasingly critical. This section explores the current landscape of NLP model interpretation, focusing on two key aspects: the explainability of model behaviors and the theoretical underpinnings of their learning processes, particularly in the context of data augmentation techniques.

Theoretical understanding and explainability in NLP models goes beyond merely interpreting their physical structures. It involves developing theoretical frameworks that can account for model behaviors, predict their outcomes, and provide insights into their decision-making processes. This approach aims to transform NLP models from "black boxes" into systems with understandable and theoretically grounded operations.

Equally important is the development of theoretical foundations for the learning processes within these models, especially when considering data augmentation techniques. By examining how data augmentation affects the inner workings of NLP models, we can gain valuable insights into their learning dynamics and generalization capabilities. This theoretical perspective allows us to view data augmentation not just as a method for expanding datasets, but as a window into the fundamental principles that govern how NLP models learn and improve.

Both areas present unique challenges and opportunities in our quest to make NLP models more transparent, explainable, and theoretically sound. In the following subsections, we will explore the current methods employed in each of these areas, as well as the challenges that researchers and practitioners face in advancing our understanding of these complex systems. By examining both the explainability of model behaviors and the theoretical foundations, we aim to provide a comprehensive view of the current state and future directions in NLP model interpretation and understanding.

2.4.1 Physical Structures in NLP Models

Physical explainability in NLP refers to the ability to understand and interpret the internal workings of a model, particularly how it processes input data and arrives at its predictions. Despite the impressive performance of models like BERT and GPT, their internal mechanics often remain opaque, earning them the reputation of "black box" models (Doshi-Velez and Kim, 2017). This lack of transparency poses significant challenges, particularly in domains

where explainability is essential for trust and accountability. In recent years, several research efforts have been made to enhance the explainability of NLP models. Some of the prominent approaches include:

Attention Mechanisms as Explanations

One of the most widely explored methods is the use of attention mechanisms as a proxy for model explainability. In models like BERT and GPT, attention weights are often interpreted as indicators of the importance of different input tokens in the model's decision-making process (Clark et al., 2019; Vig and Belinkov, 2019). However, attention weights are not always reliable indicators of importance. Studies have shown that attention may not always correlate with the true underlying decision process, leading to potentially misleading interpretations (Jain and Wallace, 2019).

Layer-wise Relevance Propagation (LRP)

Layer-wise Relevance Propagation is another technique used to backtrack the contributions of input features to the final decision made by the model (Arras et al., 2017). By propagating the prediction backward through the network, LRP aims to attribute the prediction to specific parts of the input. LRP, while useful, can become complex and computationally intensive, especially with deep architectures like BERT and GPT. Additionally, it may not always provide a clear, intuitive understanding of the model's behavior.

Saliency Maps

Saliency maps highlight which parts of the input are most influential in the model's predictions. They have been widely used in computer vision and are adapted for NLP by visualizing the influence of each word or phrase on the final output (Li and Yu, 2016). But, saliency maps often suffer from instability and may vary significantly with small changes in the input, making them less reliable for consistent explanations (Adebayo et al., 2018).

Model Distillation and Simplified Surrogates

Another approach is to distill complex models into simpler, more interpretable models, such as decision trees or linear models, that approximate the behavior of the original model (Che et al., 2016). These surrogate models can then be used to provide explanations. While these surrogate models can offer insights, they often fail to capture the full complexity of the original

model, leading to oversimplified explanations that may not accurately reflect the model's true decision-making process.

Persisting Challenges and Limitations

Despite these advancements, several challenges and limitations persist in the quest for physical explainability in NLP models:

Opacity of Deep Networks: Deep neural networks, especially those with multiple layers and complex architectures, inherently lack transparency. The sheer number of parameters and interactions makes it difficult to trace how specific input features contribute to the final output. Even with techniques like attention mechanisms or LRP, the internal decision-making process remains largely inaccessible. In addition, phenomena such as neural collapse, where the last layer's features converge to a simple structure during training (Papyan et al., 2020), further complicate our understanding. This phenomenon, which we explore in depth in Chapter 3, illustrates how even fundamental aspects of neural network behavior can be challenging to predict or explain. These issues underscore the complexity of these models and highlight how much remains unknown about their internal workings. This opacity poses significant challenges for debugging, improving, and trusting these models, especially in critical applications where understanding the reasoning behind a decision is crucial.

Inconsistent Interpretations: As highlighted by the limitations of attention mechanisms and saliency maps, there is often inconsistency in the explanations provided by different methods. This inconsistency can lead to confusion and mistrust, particularly in high-stakes applications where clear, consistent explanations are critical.

Scalability Issues: Many explainability techniques struggle with scalability. As models grow in complexity, the computational cost of generating explanations increases, making it difficult to apply these techniques to large-scale models or datasets in a practical manner.

Lack of Theoretical Understanding: A fundamental issue is the lack of a solid theoretical foundation for explainability in NLP. Most existing methods are empirical and rely on heuristic approaches, which may not generalize well across different models or tasks. This lack of theory-driven methods leaves a gap in our understanding of why certain techniques work or fail in different contexts. A better theoretical understanding of model explainability could lead to more principled approaches that improve both the interpretability and performance of NLP models.

2.4.2 Learning Process in Models: Data Augmentation Techniques

Data augmentation plays a crucial role in the learning process of NLP models by artificially increasing the diversity of training data. This makes it a valuable lens through which to study and explain the theoretical underpinnings of how models learn from data. By systematically manipulating input data and observing how models adjust their internal representations, researchers can gain insights into the mechanisms by which models generalize from examples, adapt to new tasks, and handle variations in input.

As discussed in Chapter 1, a significant challenge in NLP is the lack of an analytical framework for understanding how models learn and make decisions. Data augmentation offers a practical approach to probing these foundational issues. By observing how different augmentation techniques influence model performance and behavior, we can begin to develop hypotheses about the underlying learning processes. These hypotheses can then be tested and refined, contributing to a more systematic and theory-driven understanding of NLP.

The theoretical understanding and explanation of data augmentation in NLP remain active areas of research. While data augmentation techniques have shown empirical success, developing a robust theoretical framework to explain their effectiveness has been challenging. Here are some of the key existing approaches to explaining data augmentation:

Invariance Learning Perspective

One theoretical approach to explaining data augmentation is through the lens of invariance learning (Chen et al., 2020). This perspective posits that data augmentation helps models learn invariances to certain transformations, leading to improved generalization. For example, some researchers have used group theory to formalize the invariances learned through augmentation (Lyle et al., 2020). This approach helps explain why certain augmentations are effective for specific tasks. Another view is that augmentation helps models better learn the underlying data manifold (Verma et al., 2019). This perspective can explain why techniques like mixup can improve model robustness. However, these approaches often struggle to fully capture the complexities of natural language, where invariances can be subtle and context-dependent.

Information Theoretic Approaches

Some researchers have attempted to explain data augmentation through information theory (Dao et al., 2019). By analyzing the mutual information between augmented samples and labels,

these approaches aim to quantify the "usefulness" of different augmentation techniques (Saunshi et al., 2019). Additionally, the information bottleneck principle has been used to explain why certain augmentations can lead to more robust representations (Tishby and Zaslavsky, 2015). However, these approaches often rely on simplifying assumptions that may not hold in complex NLP scenarios.

Statistical Learning Theory

Efforts have been made to incorporate data augmentation into the framework of statistical learning theory (Chen et al., 2020). Some work has focused on how data augmentation affects the Vapnik-Chervonenkis dimension of the hypothesis space (Dao et al., 2019). The PAC-Bayesian analysis framework has also been used to derive generalization bounds for models trained with augmented data (Chen et al., 2020). These approaches provide valuable insights but often struggle to account for the unique characteristics of language data, such as its discrete nature and complex hierarchical structure.

Computational Learning Theory

Researchers have explored data augmentation through the lens of computational learning theory (Lopes et al., 2020). This approach focuses on how data augmentation affects the sample complexity, or the number of samples required to learn a concept (Chen et al., 2020). Additionally, some work has examined how augmentation impacts the algorithmic stability of learning algorithms (Kuzborskij and Lampert, 2018). While these approaches offer formal guarantees, they often rely on strong assumptions that may not hold in practice for complex NLP tasks.

Interpolation Approaches

Some researchers have proposed interpolation-based interpretations of data augmentation (Chen et al., 2020). This perspective views augmentation as a method for generating new data points by interpolating between existing samples, effectively filling in gaps in the data manifold (Verma et al., 2019). Furthermore, some studies have examined how different interpolation strategies can influence the smoothness and robustness of the decision boundary, such as using techniques like tangent space analysis to understand local data variations (Wu et al., 2022). While these approaches provide a useful framework for understanding data augmentation, applying them rigorously to high-dimensional, discrete language data can be challenging.

Persisting Challenges and Limitation

Despite significant efforts, several challenges remain in developing a comprehensive theoretical framework for explaining data augmentation in NLP. One significant challenge is the discrete nature of language; many theoretical tools are better suited to continuous data, making their application to discrete text data challenging (Bowman et al., 2016; Raganato et al., 2020). Additionally, the meaning and appropriateness of language augmentations can be highly context-dependent, complicating theoretical analysis (Andreas, 2020; Belinkov et al., 2019). The effectiveness of augmentation can depend strongly on the specific model architecture, making it difficult to develop general theories (Shen et al., 2020). Ensuring that augmentations preserve semantic meaning is crucial in NLP but difficult to formalize theoretically (Kobayashi, 2018; Zhang et al., 2020). Moreover, data augmentation often relies on heuristic methods, which can introduce noise and require careful tuning, adding to the computational cost (Feng et al., 2021; Shorten and Khoshgoftaar, 2019). These challenges highlight the complexity of developing robust and theoretically sound data augmentation techniques for NLP, underscoring the need for approaches that can better capture the unique characteristics of language data and the intricacies of modern NLP models.

2.5 Towards a Topological and Geometric Framework

The remarkable performance of LLMs in tasks like data augmentation is not just a result of their size, but also their ability to generate contextual embeddings and capture intricate linguistic patterns. These capabilities, rooted in the mathematical foundations of word embeddings, invite us to delve deeper using advanced mathematical tools (Reif et al., 2019). In this thesis, topology and geometry tools are applied for such analysis.

By combining topological and geometric approaches, we aim to develop a comprehensive understanding of how LLMs represent and manipulate language. This dual perspective allows us to capture both the global structure and local relationships within the learned representations.

This mathematical lens not only promises to enhance our theoretical understanding but also has practical implications. It could lead to more effective and interpretable data augmentation techniques, as well as guide improvements in model design and training (Reif et al., 2019). Moreover, this approach aligns with our goal of addressing the limitations discussed earlier, potentially helping to build a stronger theoretical foundation for data augmentation techniques and informing the development of more efficient models.

To address these gaps and potentially mitigate the resource-intensive nature of LLMs highlighted in Chapter 1, we propose the approach of using Topological Data Analysis and geometric methods. By applying these tools to analyze the behavior of LLMs in data augmentation tasks, we aim to uncover the underlying structures and patterns that contribute to their superior performance (Carlsson, 2009; Reif et al., 2019).

This approach has the potential to provide insights into the explainability of the physical structure of NLP models and the training data. Specifically, it can help us understand why LLMs are so effective for data augmentation and could lead to the development of more efficient augmentation techniques.

By understanding the essential geometric and topological properties that make LLM-based augmentations successful, we may be able to design more lightweight models or techniques that capture these crucial characteristics without the computational overhead of full-scale LLMs. Thus, our research aims to address both the theoretical gap in data augmentation and the practical limitations of current state-of-the-art methods, potentially paving the way for more explainable, efficient, and effective data augmentation techniques in NLP.

In the following subsections, we will review topological and geometric methods and how it can be applied to NLP tasks, particularly in the context of model and textual data analysis.

2.5.1 Topology: Unveiling Global Structures

Topology provides a powerful framework for analyzing the global structure of high-dimensional data, making it particularly well-suited for studying the complex representations learned by LLMs (Rawson et al., 2022). Unlike geometric approaches that focus on precise measurements, topology concerns itself with properties that remain invariant under continuous deformations. This makes it especially valuable for understanding the intrinsic structure of data manifolds in high-dimensional spaces.

Topology has found applications in various fields, including physics, chemistry, biology, and computer science (Carlsson, 2009). In the context of data analysis and machine learning, topological methods have been used to study the shape and structure of high-dimensional datasets, detect patterns and anomalies, and build robust models that are invariant to certain types of data transformations (Lum et al., 2013; Zomorodian, 2005).

The main idea behind topology is to capture the essential connectivity and structure of a space, disregarding the specific details of its shape or size (Hatcher, 2002). Two objects are

considered topologically equivalent, or homeomorphic, if they can be continuously deformed into each other without creating any new holes or tearing the space apart (Munkres, 2018).

Some fundamental concepts in topology include:

Open sets: A subset of a topological space is called an open set if it does not include its boundary points. Open sets form the basis for defining continuous functions and homeomorphisms between spaces (Armstrong, 2013).

Connectedness: A topological space is connected if it cannot be divided into two or more disjoint non-empty open subsets. Intuitively, a connected space is one that consists of a single piece, with no gaps or breaks (Munkres, 2018).

Compactness: A topological space is compact if every open cover of the space has a finite subcover. Compact spaces behave nicely under continuous functions and have properties that make them easier to analyze (Armstrong, 2013).

Homotopy: Two continuous functions between topological spaces are homotopic if one can be continuously deformed into the other. Homotopy provides a way to classify topological spaces and study their fundamental groups, which capture the different types of loops in the space (Hatcher, 2002).

Topological invariants: These are properties of topological spaces that remain unchanged under homeomorphisms. Examples include the number of connected components, the fundamental group, and the Euler characteristic. Topological invariants help classify and distinguish between different topological spaces (Nash and Sen, 1988).

Simplicial complexes: A simplicial complex is a combinatorial object that can be used to represent topological spaces. It consists of vertices, edges, triangles, and higher-dimensional simplices glued together in a consistent manner. Simplicial complexes are often used in computational topology and persistent homology (Edelsbrunner and Harer, 2022).

Morse theory: Morse theory studies the relationship between the topology of a manifold and the critical points of a smooth function defined on it. It provides a way to analyze the shape of a space by examining the behavior of a function, such as its gradient and Hessian. Morse theory has applications in data analysis and visualization (Milnor, 1963).

Persistent Homology

Persistent homology is a powerful tool in topological data analysis that captures the multi-scale topological features of a dataset (Edelsbrunner and Harer, 2022). It extends the concept of

homology, which studies the holes and connectivity of a topological space, to filtrations of simplicial complexes (Zomorodian, 2010).

In persistent homology, topological spaces are represented by simplicial complexes. A simplicial complex is a collection of simplices, where a simplex σ is a subset of vertices (Munkres, 2018). Let V be a set of vertices, typically denoted as:

$$V = \{1, \ldots, |V|\}$$

where $|V|$ represents the total number of vertices in the complex. For example, if we have a simplicial complex with three vertices, we can represent it as $V = \{1, 2, 3\}$. Each subset of vertices $\sigma \subseteq V$ corresponds to a simplex.

A simplicial complex K on the vertex set V is a collection of simplices $\{\sigma\}$, where each $\sigma \subseteq V$. Additionally, for any simplex τ that is a subset of a simplex σ in K, we have $\tau \in K$. This means that if, for example, we have a simplex $\{1, 2\}$ in K, then its subsets $\{1\}$ and $\{2\}$ must also be included in K. The dimension n of a simplex σ is given by:

$$n = |\sigma| - 1 \tag{2.1}$$

where $|\sigma|$ represents the number of vertices in σ. For instance, if $\sigma = \{1, 2, 3\}$, then

$$n = 3 - 1 = 2 \tag{2.2}$$

A filtration of a simplicial complex is a function:

$$f : K \to \mathbb{R} \tag{2.3}$$

that assigns a real value to each simplex in the complex. This function satisfies the property that for any simplex τ that is a subset of another simplex σ in the filtration, the assigned value of τ is less than or equal to the assigned value of σ. In mathematical terms, this can be expressed as:

$$f(\tau) \leq f(\sigma) \quad \text{whenever} \quad \tau \subseteq \sigma \tag{2.4}$$

The filtration function f allows us to order the simplices based on their assigned values, providing an ordering of the simplices in the complex.

For concrete illustrations of how these topological features manifest, refer to the illustrations in Chapter 3, such as Figure 3.8 which demonstrates how topological invariants like connected components and holes appear.

Homology groups: For each dimension k, the k-th homology group $H_k(K)$ of a simplicial complex K captures the k-dimensional holes in the complex (Hatcher, 2002). The rank of $H_k(K)$, called the k-th Betti number β_k, counts the number of independent k-dimensional holes. For example, β_0 counts the number of connected components, β_1 counts the number of loops, and β_2 counts the number of voids.

Persistence diagrams: Given a filtration $f : K \to \mathbb{R}$, persistent homology tracks the birth and death of topological features (i.e., holes) across the filtration (Edelsbrunner et al., 2008). The persistence diagram is a multiset of points in the plane, where each point (b, d) represents a topological feature that appears (is born) at filtration value b and disappears (dies) at filtration value d. The persistence of a feature is given by $d - b$, indicating its lifetime in the filtration.

Stability: One of the key properties of persistent homology is its stability under perturbations of the input data (Cohen-Steiner et al., 2005). The bottleneck distance between two persistence diagrams is bounded by the Hausdorff distance between the corresponding filtrations, providing a robustness guarantee for topological features.

There are various types of persistent homology, depending on the choice of filtration and the underlying topological space. Four main types of such spaces are the Čech complex, Vietoris-Rips complex, Alpha complex, and Cubical complex.

Čech complex: The Čech complex is a simplicial complex constructed from a set of points in a metric space (Ghrist, 2014). It captures the multi-scale connectivity of the points and is closely related to the nerve theorem in topology.

Vietoris-Rips complex: The Vietoris-Rips complex is a simplicial complex that approximates the Čech complex and is computationally more tractable (Zomorodian, 2010). It is commonly used in persistent homology computations.

Alpha complex: The alpha complex is a subcomplex of the Delaunay triangulation of a set of points in Euclidean space (Edelsbrunner et al., 2008). It is useful for studying the shape and topology of point clouds.

Cubical complex: Cubical complexes are used for studying the topology of images and voxel-based data (Kaczynski et al., 2004). They are constructed by decomposing the space into a grid of cubes and applying a filtration based on pixel intensities or a distance function.

Topology Applications in Textual Data

Topological methods have recently gained attention in the field of NLP as a means to study the intrinsic structure of textual data and improve the interpretability and robustness of NLP models (Perez and Reinauer, 2022). By representing text as a topological space and analyzing its connectivity, holes, and higher-dimensional features, researchers can gain valuable insights into the underlying patterns and relationships within the data.

Topological Data Analysis (TDA) has emerged as a prominent subfield, offering a powerful framework that combines topology, geometry, and statistics to extract meaningful features from complex datasets (Carlsson, 2009). Through methods like persistent homology, TDA enables researchers to quantify and visualize multi-scale topological properties of data, revealing connectivity patterns, loops, and higher-dimensional holes.

TDA has demonstrated significant utility across various NLP applications. At the word level, Gholizadeh et al. (2020) applied persistent homology to analyze the topological structure of word embedding spaces, revealing significant loops and holes that capture semantic relationships between words. Similarly, Zhu (2013) demonstrated TDA's capability to capture both syntactic and semantic information in language data.

At the document level, researchers have leveraged TDA to uncover broader patterns and relationships. Doshi (2018) employed TDA to examine the geometry of document embeddings, successfully identifying meaningful clusters and topic transitions within document corpora. Naitzat et al. (2020) extended this approach to detect anomalous structures in document collections, including outliers and clusters with unusual connectivity patterns.

TDA has also proved valuable in studying the temporal dynamics of language and knowledge. Michel et al. (2011) applied topological methods to analyze word co-occurrence networks across different time periods, revealing significant changes in word connectivity and clustering that reflect cultural and linguistic evolution. In academic literature, Sizemore et al. (2019) used TDA to track the emergence and dissipation of research topics over time in scientific papers.

A notable example of integrating topology with state-of-the-art NLP models is the Topological BERT, proposed by Perez and Reinauer (2022). They introduced a topological attention mechanism that captures the intrinsic structure of the input text and incorporates it into the self-attention layers of the BERT model. By transforming the attention weights into a topological space and applying persistent homology, Topological BERT can capture higher-order interactions between words and phrases, leading to improved performance on various NLP tasks, such as sentiment analysis and question answering. This work demonstrates the potential of

combining topological methods with deep learning architectures to enhance the expressiveness and interpretability of NLP models.

Despite the numerous successful applications of TDA in NLP, significant challenges and unexplored territories remain. Although topological methods offer a unique lens into model behavior by revealing the structure and connectivity of learned representations (Hensel et al., 2021), this theoretical advantage has yet to translate into comprehensive practical insights. Current applications primarily focus on analyzing static data patterns rather than understanding the dynamic decision-making processes of NLP models.

The gap between topology's theoretical promise and practical implementation is particularly evident in model interpretability. While researchers have shown that analyzing the topological properties of hidden layers and attention mechanisms could illuminate how models process linguistic information, developing robust methodologies for this analysis remains challenging. Furthermore, Corneanu et al. (2020) demonstrated that while topological analysis can help identify model limitations and vulnerabilities, such as sensitivity to adversarial examples or reliance on superficial patterns, translating these findings into actionable improvements for model architecture or training remains an open challenge.

In summary, topology offers a powerful framework for studying the intrinsic structure of textual data and improving the interpretability and robustness of NLP models. While the application of topological methods in NLP is still in its early stages, recent research has demonstrated their potential for addressing various challenges, from data augmentation and bias detection to model interpretation and explanation.

In addition, although topology can reveal patterns and holes within textual data, it has its limitations in providing a complete understanding of the underlying structure. Topological methods focus on the connectivity and qualitative properties of data, but they do not capture the specific shapes, sizes, or orientations of the features. Such features are in the domain of geometric methods. They could offer a complementary perspective that reveal more detailed, quantitative information about the data.

2.5.2 Geometry: Analyzing Local Relationships

While topology gives us a global view, geometric approaches allow us to analyze the local relationships within the embedding space. The vector space nature of word embeddings and the high-dimensional representations in LLMs naturally lend themselves to geometric analysis (Kozlowski et al., 2019). Geometric techniques can help us understand the spatial relationships

between words or concepts in the embedding space. For instance, we can use cosine similarity or Euclidean distance to measure the relatedness of words (Mikolov et al., 2013c). These methods can provide insights into how LLMs capture semantic and syntactic relationships. Furthermore, geometry allows us to explore linear substructures in the embedding space that correspond to semantic or syntactic relationships, offering a window into how the model encodes linguistic information (Arora et al., 2018). This can be particularly useful for understanding how data augmentation techniques affect the geometric properties of the embedding space.

Geometric methods have been widely applied in machine learning and NLP to study the structure and properties of data representations, such as word embeddings and feature spaces. By analyzing the geometric relationships between data points, researchers can gain insights into the underlying patterns, similarities, and differences within the data, and develop more effective and interpretable models.

Convex Hull

The convex hull is a fundamental concept in computational geometry that has found applications in various aspects of machine learning, including data visualization, outlier detection, and model regularization (de Berg et al., 2008). Given a set of points in a Euclidean space, the convex hull is defined as the smallest convex set that contains all the points. Intuitively, it can be thought of as a tight-fitting convex envelope around the data points.

There are several algorithms for computing the convex hull of a set of points, each with its own strengths and weaknesses. The choice of the specific algorithm depends on factors such as the dimensionality of the data, the number of points, and the desired trade-off between computational efficiency and robustness. Some common algorithms used in convex hull includes:

Graham Scan: This algorithm sorts the points based on their polar angles with respect to a reference point and then iteratively constructs the convex hull by considering each point in order and updating the hull accordingly (Graham, 1972). It has a time complexity of $O(n \log n)$, where n is the number of points.

Quickhull: This algorithm is based on the divide-and-conquer paradigm and recursively partitions the points into subsets based on their relationship to a line connecting two extreme points (Barber et al., 1996a). It has an average time complexity of $O(n \log n)$ but can degrade to $O(n^2)$ in the worst case.

Incremental Algorithm: This algorithm constructs the convex hull incrementally by adding points one at a time and updating the hull accordingly (Kallay, 1984). It has a time

complexity of $O(n \log n)$ in the worst case but can be faster in practice for certain point distributions. In the context of NLP, the convex hull has been used to study the geometric properties of word embeddings and to identify salient regions in the embedding space (Bouraoui et al., 2022).

Delaunay Triangulation

Delaunay triangulation is a fundamental geometric concept that has found applications in various fields, including computer graphics, mesh generation, and data visualization (de Berg et al., 2008). Given a set of points in a Euclidean space, the Delaunay triangulation is a triangulation of the points such that no point is inside the circumcircle of any triangle. It provides a natural way to connect nearby points and to capture the local structure of the data.

There are several algorithms for computing the Delaunay triangulation of a set of points, including:

Bowyer-Watson Algorithm: This is an incremental algorithm that constructs the Delaunay triangulation by adding points one at a time and updating the triangulation accordingly (Bowyer, 1981; Watson, 1981). It has a time complexity of $O(n^2)$ in the worst case but can be faster in practice.

Divide-and-Conquer Algorithm: This algorithm recursively divides the set of points into subsets, computes the Delaunay triangulation for each subset, and then merges the resulting triangulations (Lee and Schachter, 1980). It has a time complexity of $O(n \log n)$ in the worst case.

Incremental Insertion Algorithm: This algorithm starts with an initial triangulation of a subset of points and then incrementally inserts the remaining points, updating the triangulation after each insertion (Lawson, 1977). It has a time complexity of $O(n^2)$ in the worst case but can be faster for certain point distributions.

Geometry in Textual Data

Word embeddings, in particular, exhibit interesting geometric properties that can be exploited for various NLP tasks and analyses (Arora et al., 2016; Mimno and Thompson, 2017). For example, word embeddings have been shown to form a cone-like structure in the high-dimensional space, with more frequent words located closer to the origin (Arora et al., 2016). This geometric structure has been used to develop more efficient and interpretable word embedding models, such as the squared-norm word embedding model proposed by Arora et al. (2016).

Moreover, the cosine similarity between word vectors has been widely used as a measure of semantic similarity, with words having similar meanings clustered together in the embedding space (Mikolov et al., 2013a). This property has been exploited in tasks such as word sense disambiguation (Iacobacci et al., 2016) and semantic role labeling (Roth and Lapata, 2016), where the geometric proximity of word embeddings can provide valuable information about their semantic relationships.

The geometric relationships between word embeddings have also been employed to study the structure and evolution of language (Hamilton et al., 2016; Kutuzov et al., 2018). By comparing word embeddings trained on text corpora from different time periods, researchers have tracked semantic shifts and identified changes in word usage over time (Hamilton et al., 2016). This approach has shed light on the dynamic nature of language and has implications for understanding language change and development.

In addition to word embeddings, geometric methods have been applied to analyze the feature spaces learned by neural networks in NLP tasks. For example, Ethayarajh (2019) studied the geometry of contextualized word representations, such as those obtained from BERT (Devlin et al., 2019), and found that they exhibit anisotropic properties, with most of the variance concentrated in a few dominant directions. This finding has implications for understanding the limitations and potential biases of contextualized word representations and highlights the need for further research into their geometric properties.

While computational geometry concepts such as Delaunay triangulation have proven to be a powerful tool in computer vision applications (Elshakhs et al., 2024), its potential in NLP remains largely unexplored. For our investigations, the specific implementation details, performance characteristics, and rationale behind our choice of algorithms will be discussed in each experimental section.

For practical illustrations of convex hull analysis applied to text embeddings and data augmentation, see the experimental results in Figure 5.5 (Chapter 5), which demonstrate how augmented data points relate to the semantic boundaries defined by the original data's convex hull. Our experimental results in Figure 5.6 (Chapter 5) provide a concrete example of Delaunay triangulation applied to word embeddings, illustrating how this geometric structure captures local connectivity patterns in text data and how augmentation affects the triangulation density.

2.6 Summary

In this chapter, we have delved into the background of NLP and a review on the the critical challenges and opportunities for advancing the analytical frameworks that can explain NLP model behaviour. The challenges have also been highlighted with a preliminary experiment.

We then explored the potential of integrating topological and geometric methods as novel tools to enhance our understanding of NLP models. These mathematical approaches offer promising avenues to uncover both global and local structures within the high-dimensional spaces where NLP models operate. By applying these techniques, we can gain deeper insights into the inner workings of complex models like LLMs and potentially improve their efficiency and explainability. Below are some key insights in this chapter.

- **Topological Analysis**: We highlighted the role of topological data analysis in revealing the global structure of high-dimensional data representations. Topology's ability to capture invariant properties under continuous transformations makes it a powerful tool for understanding the underlying patterns in word embeddings and neural networks.
- **Geometric Methods**: We discussed how geometric approaches can provide a detailed analysis of local relationships within embedding spaces, offering a nuanced understanding of how models capture semantic and syntactic relationships.
- **Bridging Theory and Practice**: The integration of topological and geometric methods serves as a bridge between theoretical understanding and practical improvements in NLP. By grounding in these mathematical frameworks, we aim to develop more efficient and explainable models that maintain high performance without the need for extensive computational resources.

This chapter sets the stage for the subsequent exploration of how topological and geometric methods can be applied to enhance NLP model performance and explainability. The insights gained here will inform the development of new algorithms and techniques that address the challenges of resource intensity and lack of theoretical grounding.

In the next chapter, we will first examine the phenomenon of Neural Collapse in NLP models using topological data analysis. This investigation will further our understanding of the physical behavior of neural networks.

Chapter 3

A Study of Neural Collapse in NLP Models

In the previous chapter, we examined the challenges surrounding NLP models, particularly the opacity of their internal workings and the lack of robust explanations for how these models process and understand language. Topological and geometric methods were introduced as promising tools for uncovering the underlying structures within these models, aiming to bridge the gap between performance and explainability. This sets the stage for a deeper investigation into the physical explainability of NLP models, a critical aspect of developing more interpretable and trustworthy AI systems.

Understanding the physical structure and behavior of deep learning networks is crucial for advancing the explainability of NLP models. The complexity of these models, particularly in their final layers, often leads to a "black box" problem, where the decision-making process remains hidden from human understanding. In this chapter[1], this challenge is addressed by focusing on Neural Collapse (NC), an observed phenomenon that offers a unique lens through which to study the physical structure of neural networks. NC provides a case study for how deep learning models, particularly those used in NLP, converge to a simplified and interpretable state under certain conditions.

By investigating the occurrence and characteristics of Neural Collapse in NLP, we uncover how deep learning models transition into a simplified state where their decision boundaries become more structured and interpretable. This investigation provides critical insights into the

[1]The main contents of this chapter are based on the author's publication *A Study of Neural Collapse for Text Classification* cited on Page-*iv*.

geometric and topological properties of NLP models, offering a deeper understanding of their internal mechanisms.

This chapter addresses **RQ1** stated in Section 1.3.

3.1 The Phenomenon of Neural Collapse

Deep Learning (DL) is a subfield of machine learning that utilizes artificial neural networks to learn hierarchical representations of data (Goodfellow et al., 2016). In recent years, DL has achieved remarkable success in various domains, including computer vision, speech recognition, and NLP (LeCun et al., 2015). NLP focuses on the interaction between computers and human language, aiming to enable machines to understand, interpret, and generate human language (Hirschberg and Manning, 2015).

Neural networks are the foundation of deep learning. They are composed of interconnected nodes (neurons) organized in layers, loosely inspired by the structure of the human brain (Schmidhuber, 2015). Each neuron receives input from other neurons, applies a non-linear activation function, and passes the output to the next layer. Through this process, neural networks can learn complex patterns and representations from data (Goodfellow et al., 2016).

The connections between neurons are weighted, and these weights are learned during the training process using optimization algorithms like stochastic gradient descent (SGD) (Bottou, 2010). The goal of training is to minimize a loss function that measures the discrepancy between the predicted and true outputs. As the network learns, it adjusts the weights to better capture the underlying patterns and relationships in the data.

The most common types of neural networks used in NLP include:

- Feedforward Neural Networks (FFNNs): These networks have a unidirectional flow of information from input to output, without any loops or cycles (Bengio et al., 2000).
- Recurrent Neural Networks (RNNs): RNNs are designed to handle sequential data by maintaining a hidden state that captures information from previous inputs (Elman, 1990). They are particularly useful for tasks such as language modeling and machine translation (Sutskever et al., 2014).
- Convolutional Neural Networks (CNNs): Originally developed for computer vision tasks, CNNs have also been successfully applied to NLP tasks such as text classification and

sentiment analysis (Kim, 2014). They are effective in capturing local patterns and features in text data.

- Transformer Networks: Introduced by Vaswani et al. (2017), Transformers have revolutionized NLP by relying solely on attention mechanisms to capture dependencies between input and output sequences. They have become the foundation for state-of-the-art language models such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018).

In the last decade or so, using DL (Goodfellow et al., 2016) for classification has become standard practice. A lot of experience and insights into training such networks have been gained. For instance, it is generally not advisable to over-train an over-parametrized model such as a neural network. This is to prevent *over-fitting* the model to the training data and therefore affects its generalization ability (He and Tao, 2021; Jabbar and Khan, 2015).

However, an unusual phenomenon has been discovered empirically when a network is over-trained. This occurs when training persists beyond the point where the error vanishes. At this point, if training continues to drive the loss towards zero, then the network will enter the so-called NC state (Papyan et al., 2020). The network is said to be in the terminal phase of training (TPT) when the error is very small or zero. Surprisingly, such an over-trained network is able to achieve high levels of classification performance. Experimental results by Papyan et al. (2020) showed that NC networks achieves a performance that is better than a non-collapsed neural network. This is because in the neural collapsed state, the last layer activations concentrates towards a symmetric structure, making the job of the classifier relatively simple. It has also been shown to be a model that has generalizable and transferable learning properties (Galanti et al., 2022). Authors Hui et al. (2022) noted that the NC phenomenon occurs only with the training data. They found that when separate test data are used with the NC network, it does not apply and hence they concluded that NC is primarily an optimization phenomenon.

3.1.1 Characteristics of Neural Collapse

There are four inter-related characteristics that are exhibited when neural collapse has occurred. They are described as follows.

NC1: *Collapse of variability*

In the penultimate layer of the network, the data samples that belong to the same class moves

towards their class mean, and the variability of the intra-class features is lost as they collapse to their class mean. That is, the feature within-class variance. To quantify this collapse, we can use the covariance matrix, denoted by $\mathbf{\Sigma}_W$, which captures the within-class variance. The covariance matrix measures the relationships between pairs of features and can provide insights into the spread and correlations within the data.

The equation for the covariance matrix is given by:

$$\mathbf{\Sigma}W = \underset{i,c}{\text{Ave}} \left(\boldsymbol{h}i, c - \boldsymbol{\mu}c\right) \left(\boldsymbol{h}i, c - \boldsymbol{\mu}_c\right)^\top, \tag{3.1}$$

Here, $\boldsymbol{h}_{i,c}$ represents the feature vector of the i-th sample belonging to class c. The symbol $\boldsymbol{\mu}_c$ denotes the class mean, which is the average of all feature vectors belonging to class c. The operator $\underset{i,c}{\text{Ave}}$ indicates the averaging operation, which computes the average covariance matrix over all samples and classes.

By examining the covariance matrix $\mathbf{\Sigma}_W$, we can observe how the within-class variance changes throughout the network. In the case of the collapse of variability, as the network progresses, the values in the covariance matrix tend to approach zero, indicating a significant reduction in the variability of the intra-class features.

NC2: *Preference towards a simplex equiangular tight frame (ETF)*
The class means of the final layer moves towards forming a simplex ETF, which can be mathematically represented by the following:

$$\begin{gathered}\frac{\langle \boldsymbol{\mu}_c - \boldsymbol{\mu}_G, \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G \rangle}{\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2} \to \begin{cases} 1, & c = c' \\ \frac{-1}{C-1}, & c \neq c' \end{cases} \\ \|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 - \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2 \to 0 \quad \forall c \neq c'\end{gathered} \tag{3.2}$$

where

$$\boldsymbol{\mu}_G = \underset{i,c}{\text{Ave}}\, \boldsymbol{h}_{i,c}, \tag{3.3}$$

is the feature global mean.

In these equations, $\boldsymbol{\mu}_c$ represents the mean feature vector of class c, while $\boldsymbol{\mu}_{c'}$ represents the mean feature vector of class c'. The symbol $\boldsymbol{\mu}_G$ denotes the global mean of the features, which is computed as the average of all feature vectors across all classes. The angle brackets $\langle \cdot, \cdot \rangle$ represent the dot product, and the notation $\| \cdot \|_2$ denotes the Euclidean norm.

The first equation expresses the cosine similarity between the differences of the class means and the global mean, normalized by the product of their Euclidean norms. As the network progresses, this similarity tends towards specific values: 1 when comparing the same class means ($c = c'$), and $-\frac{1}{C-1}$ when comparing different class means ($c \neq c'$), where C is the total number of classes.

The second equation states that the difference in Euclidean norms between any pair of class means ($c \neq c'$) tends towards zero. In other words, the magnitudes of the class mean vectors become approximately equal.

Together, these equations indicate the network's preference towards a simplex ETF formation in the final layer, where the class means exhibit specific angular relationships and similar magnitudes. This preference suggests a structured representation of the classes, potentially aiding in better discriminability and classification performance.

NC3: *Convergence to self-duality*

There is also a convergence to self-duality observed in the vectors of the last layer, represented mathematically as follows:

$$\frac{\boldsymbol{w}_c}{\|\boldsymbol{w}_c\|_2} - \frac{\boldsymbol{\mu}_c - \boldsymbol{\mu}_G}{\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2} \to 0 \tag{3.4}$$

In this equation, $\boldsymbol{w}_c$ represents the weight vector associated with class c in the last layer of the network. The term $\boldsymbol{\mu}_c - \boldsymbol{\mu}_G$ represents the difference between the class mean $\boldsymbol{\mu}_c$ and the global mean $\boldsymbol{\mu}_G$ of the features.

The equation states that as the network progresses, the difference between the normalized weight vector $\frac{\boldsymbol{w}_c}{|\boldsymbol{w}_c|_2}$ and the normalized difference between the class mean and the global mean $\frac{\boldsymbol{\mu}_c - \boldsymbol{\mu}_G}{|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G|_2}$ tends towards zero. In other words, the weight vectors become increasingly aligned with the difference between the class mean and the global mean, in a normalized sense.

NC4: *Simplification to the nearest class mean*

In the last layer of the network, the classifier examines the features from the penultimate layer of a test point and assigns a label to the test point based on the closest train-class mean. This process is similar to a 1-Nearest Neighbour classifier (Kothapalli, 2023). Mathematically,

this can be expressed as

$$\arg\max c' \langle \boldsymbol{w}c', \boldsymbol{h} \rangle + b_{c'} \rightarrow \arg\min c' \|\boldsymbol{h} - \boldsymbol{\mu}c'\|_2 \tag{3.5}$$

where $\boldsymbol{w}c'$ is the weight vector associated with the class c' in the last layer of the network. $b_{c'}$ denotes the bias term associated with class c'. $\boldsymbol{h}$ denotes the feature vector of the test point, which is extracted from the penultimate layer. The term $\boldsymbol{\mu}c'$ represents the class mean of the training samples for class c'.

The equation indicates that the classifier calculates the dot product between the weight vectors and the feature vector $\boldsymbol{h}$, adds the corresponding bias terms, and selects the class c' that maximizes this expression. This process effectively assigns the test point to the class with the highest similarity between its weight vector and the test point's feature representation. Simplifying this equation, we can rewrite it as selecting the class c' that minimizes the Euclidean distance between the feature vector $\boldsymbol{h}$ and the class mean $\boldsymbol{\mu}_{c'}$. This approach is analogous to the nearest neighbor classification, where the test point is assigned the label of the closest training sample based on the Euclidean distance. This simplification suggests that the network is learning to map the features of a test point to the class means in a manner that resembles a nearest neighbor approach. It implies that the network's decision-making process is driven by the similarity between the test point and the class prototypes represented by the class means.

A visualization of NC with three labels is shown in Figure 3.1. To complement the visual, the experiments by Papyan et al. (2020) were re-run, and the feature collapse points on the MNIST dataset using T-SNE are presented in Figure 3.2.

3.1.2 Domain of Investigations

Many subsequent works have extended the original research on NC of Papyan et al. (2020) to further understand this phenomenon. Some works contributed theoretic understanding using unconstrained features model (Ji et al., 2022; Mixon et al., 2022). Further geometric insights are provided by Zhu et al. (2021) and Zhou et al. (2022).

Others studied the effects of the loss function on NC (Han et al., 2022; Lu and Steinerberger, 2022; Zhou et al., 2022). The effects of imbalanced training have also been considered (Thrampoulidis et al., 2022). It has also been shown that an NC model has generalizable and transferable learning properties (Galanti et al., 2022; Li et al., 2024). There are also potential improvements proposed to make NC more efficient and effective (Yaras et al., 2022). There has also been

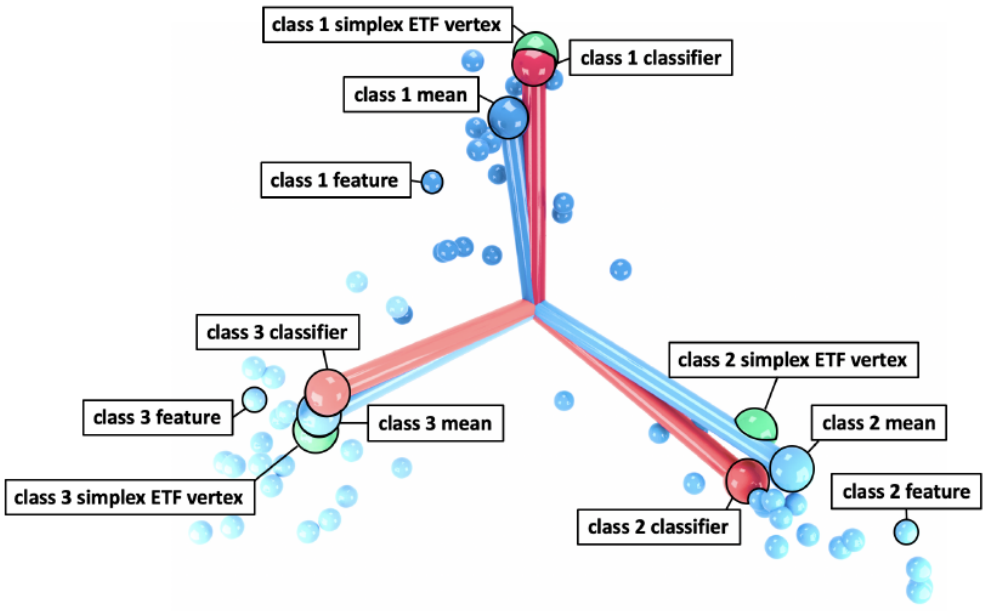
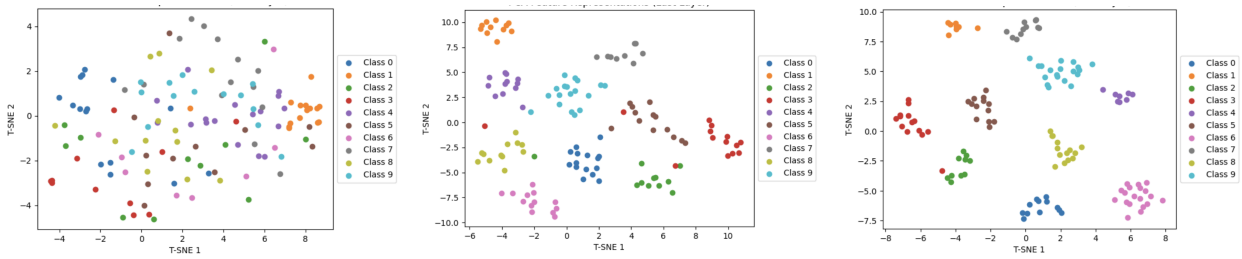


Fig. 3.1 Visualization of NC with labels (Han et al., 2022)



(a) Epoch 1 (b) Epoch 50 (c) Epoch 200

Fig. 3.2 Feature Points Collapse on MNST dataset

research looking into the limitations of NC. One limitation researchers have discovered is that NC can only occur with weight decay (Rangamani and Banburski-Fahey, 2022). Also, it has been found that NC does not occur on imbalanced datasets (Thrampoulidis et al., 2022).

However, these studies have all been conducted in the context of computer vision tasks, such as image classification. To the best of the author's knowledge, NC has not been demonstrated to occur for NLP tasks before. Thus, there is a need to ensure that this phenomenon is not unique to computer vision tasks. Furthermore, investigating NC in NLP models could provide valuable insights into their learning dynamics and generalization properties. For example, understanding how the learned feature representations in language models like BERT (Devlin et al., 2019) or GPT (Radford et al., 2019) evolve during training and converge to a simple, well-defined structure. This knowledge could help explain their impressive performance and ability to generalize across different tasks.

Moreover, understanding NC in NLP models could contribute to the development of more explainable and interpretable language models. The convergence of learned features to a simple, well-defined structure can potentially make it easier to interpret and explain the model's predictions. By analyzing the properties of NC in language models, researchers could gain insights into the key features or patterns that the model relies on for making its decisions.

3.2 Neural Collapse in NLP Models

Text data exhibits hierarchical and sequential properties, such as word order and syntactic structure, which may influence the learning dynamics and manifestation of NC in NLP models. More research is needed to fully understand how NC manifests in different NLP tasks and architectures, and how it relates to other phenomena observed in language models, such as attention mechanisms (Vaswani et al., 2017) and contextualized word embeddings (Peters et al., 2018b). We hypothesize that NC would also occur in text classification tasks, given the agnostic nature of deep learning models to the type of input data.

One area where NC may be particularly relevant in NLP is text classification. Similar to image classification, text classification tasks involve assigning input text to predefined categories or classes. Computational experiments have been designed for studying the NC phenomena used in text classification. Details of the experiments are provided in Section 3.2.1. It is followed by the results and discussions. To the best of the author's knowledge, these are the first published results on NC with textual data.

Our initial findings indicate the occurrence of NC on textual data, which initially underperforms compared to a non-collapsed CNN (Feng et al., 2023). However, upon closer examination, we uncover an intriguing insight: certain data points converge towards an unknown cluster during NC. Further analysis reveals that this additional cluster represents an additional topic within the dataset, challenging the initial assumption of four distinct classes in AG News (Feng et al., 2023). This significant discovery suggests a promising research direction, where NC can serve as a tool for cluster discovery in semi-supervised learning scenarios (Feng et al., 2023).

3.2.1 Experimental Setup

The experiments for text classification is set up as follows.

Dataset We use AG's corpus of news articles (Zhang et al., 2015) containing textual data of news articles and their corresponding class label. Each class label represents a news category. There are four categories – world, sports, business and science/technology. In our experiments, 50 samples for each label are randomly chosen for each dataset. 45 of these samples are used for training and 5 for testing, i.e. a 90/10 split.

Network Architecture The same CNN architecture as that used in (Zhang and Wallace, 2017) is employed here, it has proven to work well in detecting local and positional invariant patterns (Minaee et al., 2021). The architecture is shown in Figure 3.3. In addition, weight decay is introduced into the training as it has been shown that it can lead to neural collapse (Rangamani and Banburski-Fahey, 2022). But the baseline model does not use weight decay. The specific details of the network are:

- Embedding dimension: 300
- Filters: (100, 100, 100)
- Activation: ReLU
- Pooling: 1-max pooling
- Dropout rate: 0.5
- Word Embedding: fastText
- Weight Decay: $5e-4$

fastText is used as the word embedding as it has shown promising results for CNN in text classification (Umer et al., 2023a).

Training The baseline model is trained for 20 epoch. The NC model is trained until neural collapse occurs. In this case, it takes 500 epochs.

The four characteristics of NC described in Section 3.1.1 are shown in Figures 3.4b (NC1), 3.4c (NC2), 3.4d (NC3), and 3.4e (NC4) respectively, where NC1, NC3, and NC4 show convergence. Although there are still fluctuations in NC2, the variation is small. Therefore, we can say that the network reaches the NC state.

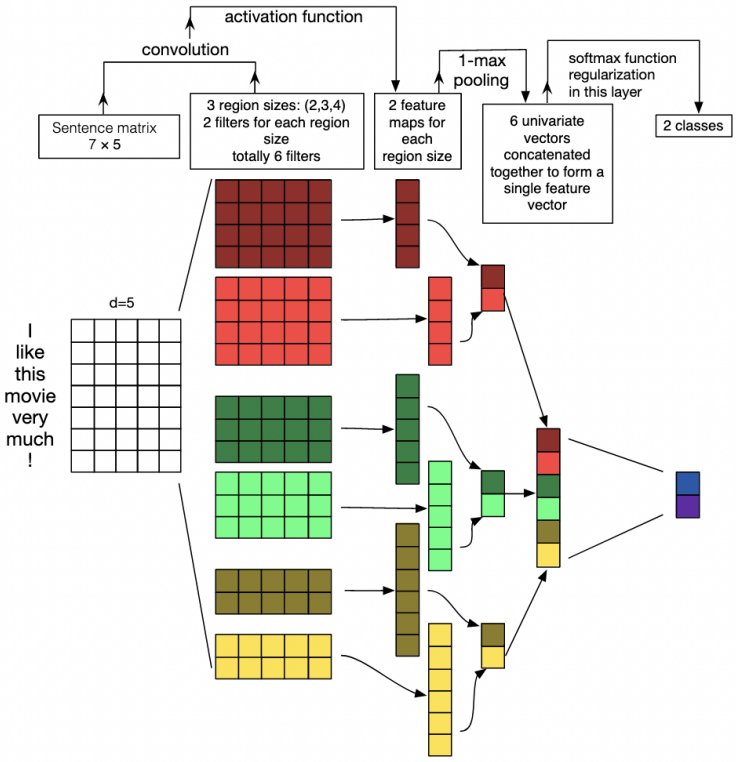


Fig. 3.3 CNN Architecture (Zhang and Wallace, 2017)

3.2.2 Results

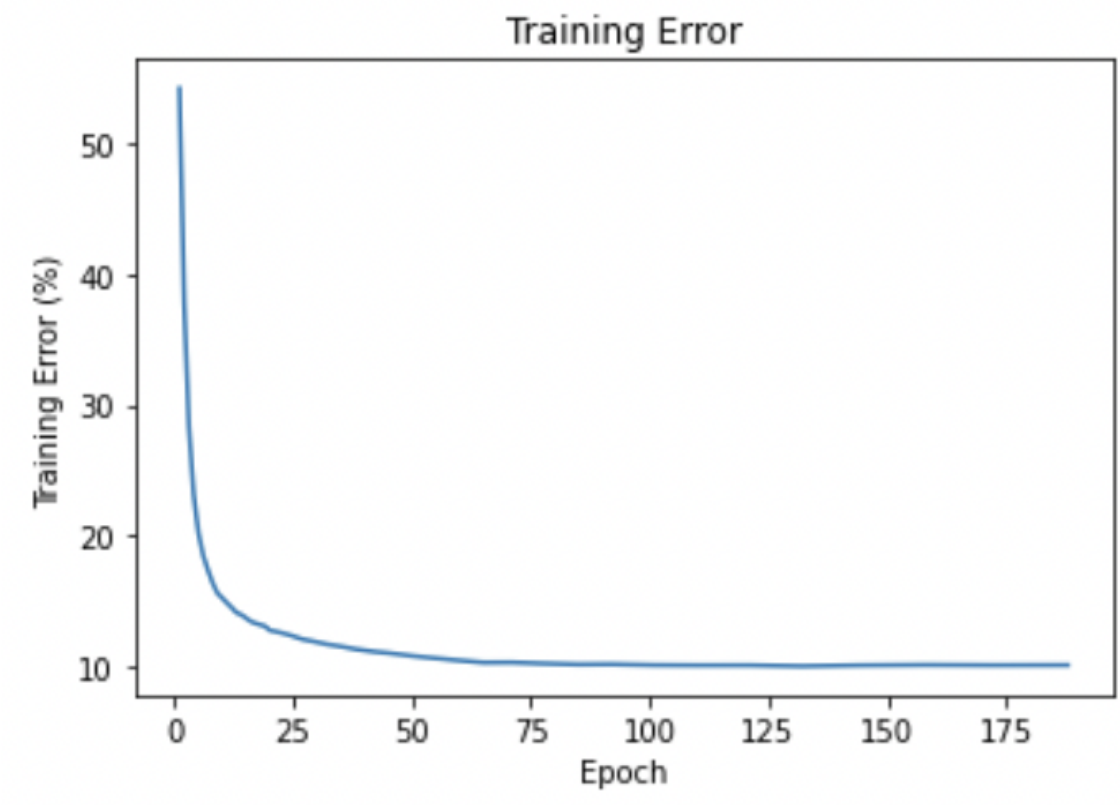
The classification accuracies of the baseline CNN and the NC network are shown in Table 3.1. While the baseline model achieves a classification accuracy of over 92% after being trained for 20 epochs, the NC model's accuracy is only around 55%. This indicates that the NC network does not generalize well for the test data.

Model	Acc.
Baseline	$92.3\% \pm 0.2\%$
NC CNN	$55.2\% \pm 0.5\%$

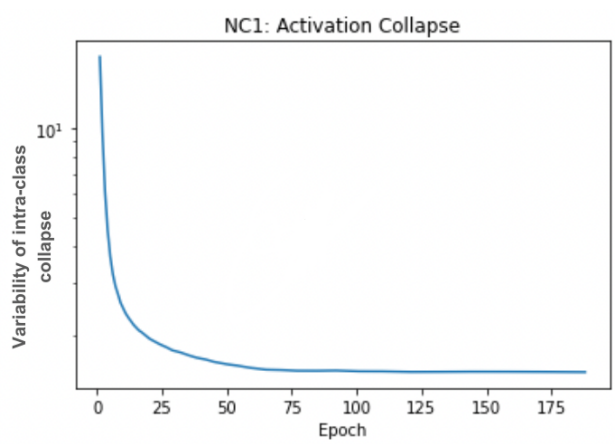
Table 3.1 Classification accuracies of baseline CNN and NC network

Next, we examine the output confidence of the two models using specific texts from the test data. Table 3.3 shows the output confidence for a sample text taken from the topic of science/technology shown in Table 3.2:

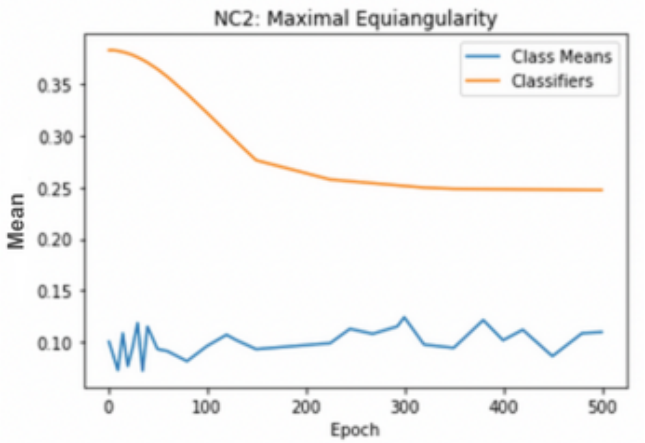
For the baseline CNN, the highest confidence for this input is for the *science/technology* label (79.3%), which is the correct label. However, with the NC network, the confidences of all four labels are similar, with the highest being 26.29% for the *world* label. It seems that after



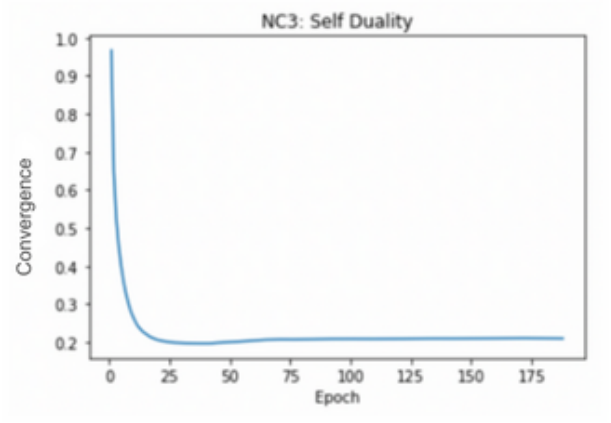
(a) Training error



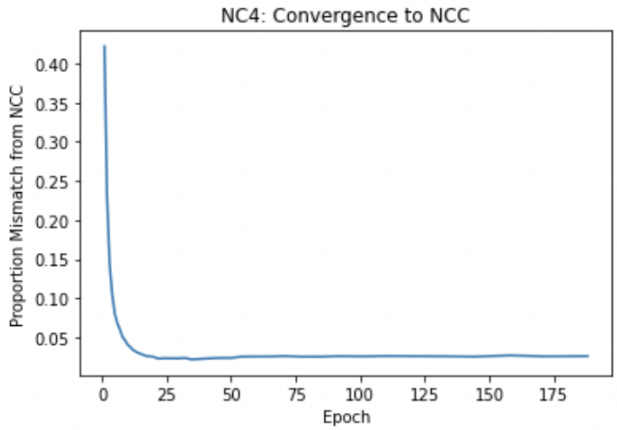
(b) NC1



(c) NC2



(d) NC3



(e) NC4

Fig. 3.4 Evidences of Neural Collapse

Table 3.2 Science/Technology sample text

A company founded by a chemistry researcher at the University of Louisville won a grant to develop a method of producing better peptides, which are short chains of amino acids, the building blocks of proteins.

NC, the prediction for each label for each misclassified data sample is very similar. Hence, the NC model's prediction cannot choose any of the classes with confidence.

Table 3.3 Output confidence of the two models

Topic	**Baseline**	**NC**
World	5.64%	**26.29**%
Sports	2.08%	25.69%
Business	12.98%	23.02%
Science/Technology	**79.30**%	25.00%

The output confidence values represent the softmax probabilities computed by each model's final classification layer. For both the baseline CNN and the NC network, confidence scores are obtained by passing the test sample through the trained model and extracting the probability distribution over all four classes using the standard softmax activation function. Specifically, the confidence for each class c is computed as:

$$\text{confidence}_c = \frac{e^{z_c}}{\sum_{i=1}^{4} e^{z_i}}$$

where z_c is the raw logit output for class c before the softmax activation. These probabilities sum to 100% across all classes, and the highest confidence value indicates the model's predicted class. The confidence computation was implemented using standard deep learning frameworks (e.g., `model.predict_proba()` in scikit-learn style or `torch.softmax()` in PyTorch), which apply the softmax function to the model's final layer outputs to produce normalized probability distributions.

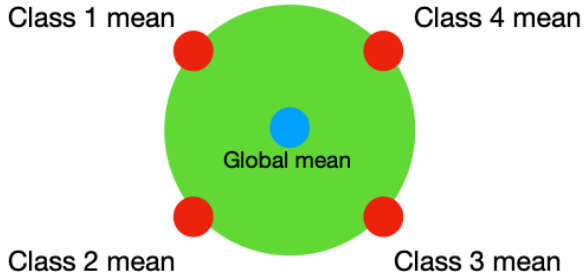


Fig. 3.5 Geometric Distribution of Class Means

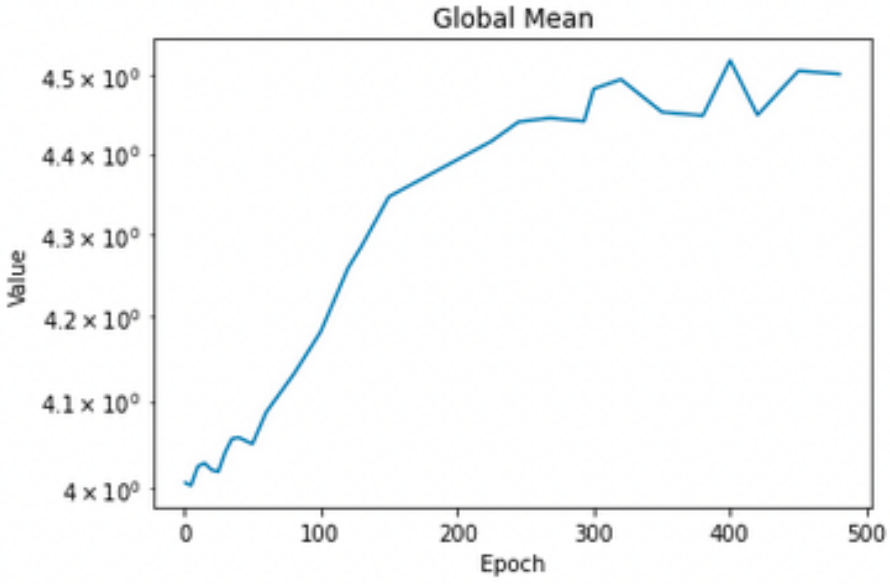


Fig. 3.6 Global Class Mean during NC training

3.2.3 Finding Misclassified Data Activations through Topological Data Analysis

The results given in Table 3.3 seems to indicate that the activations of the misclassified data may be clustered. Since each of the four classes form an ETF, it is reasonable to postulate that misclassified samples are clustered around the global mean, as demonstrated in Figure 3.5.

A graph of the global mean is shown in Figure 3.6. Comparing this with the misclassified data mean as shown in Figure 3.7, it is clear that the misclassified sample activations do not reside around the global mean. Even though this is the case, it could still be possible that the misclassified samples are still clustered together since their output confidences are similarly distributed.

In order to confirm this, we visit the concept of topology. This is because topological properties of word embeddings are fundamental building blocks of many NLP models (Mikolov

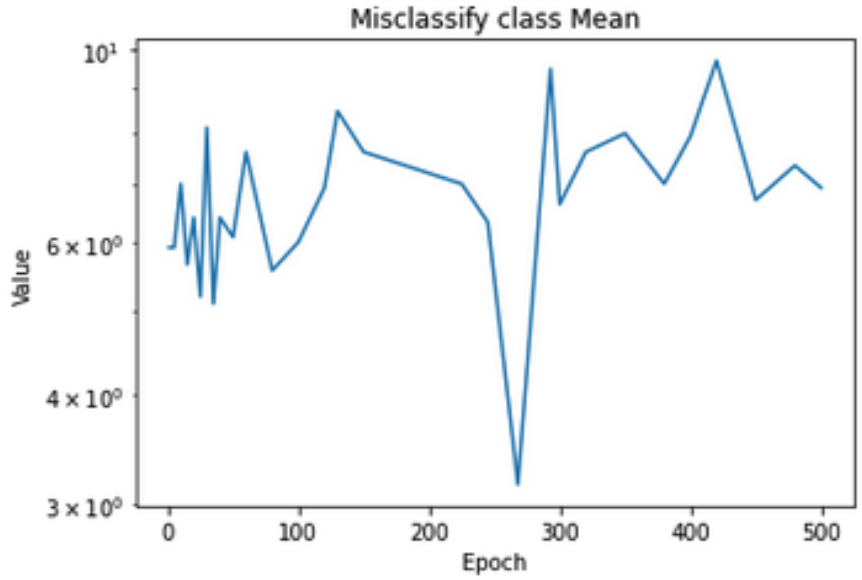


Fig. 3.7 Mislabeled Class Mean during NC training

et al., 2013b; Pennington et al., 2014). By studying the shapes and structures formed by these embeddings, we can gain insights into the relationships between words and their meanings (Gao et al., 2019). Although topology has already been introduced and explored in Chapter 2, we briefly explain topology properties in the context of this study.

Topological Data Analysis

Topology is a branch of mathematics that studies the properties of spaces that are preserved under continuous deformations, such as stretching, twisting, or bending, but not tearing or gluing (Munkres, 2018). In other words, topology focuses on the qualitative aspects of geometric objects, rather than their precise measurements or coordinates.

Topology offers a unique perspective on understanding the structure and behavior of NLP models. By representing text data in spatial shapes and studying their topological properties, researchers can gain insights into the inherent relationships and patterns within the data (Gao et al., 2010). This approach has the potential to provide a more interpretable and explainable framework for NLP models, addressing the limitations of current black-box approaches (Naitzat et al., 2020). Moreover, topological methods have been successfully applied in various domains, such as computer vision (Bronstein et al., 2017) and network analysis (Sizemore et al., 2019), demonstrating their potential to uncover meaningful structures and relationships in complex data. Applying these techniques to NLP can lead to novel insights and advancements in the field (Zhu et al., 2003).

TDA is a statistical method utilising algebraic topological concepts to find structure in data. It has been used in a variety of applications. More recently, it has been applied for cluster analysis in machine learning (Wasserman, 2018). TDA is based on the ideas of persistent homology, a mathematical tool in algebraic topology (Edelsbrunner et al., 2008), which enables the study of the topological properties of data from a point cloud in a metric space such as $\mathbb{R}^d$, where $\mathbb{R}^d$ represents a Euclidean space of dimension d. Very often, the data points exist in high dimensions and therefore are difficult to visualize. More specifically, persistent homology enables the visualization of the formation and destruction of loops and voids in the data space through persistent diagrams and barcodes.

A persistence diagram shows the filtration values at which a topological feature (such as a connected component, loop, or void) is born and when it dies. It consists of points in a plane, where each point represents a topological feature and its coordinates (birth, death) indicate the filtration values at which the feature appears and disappears. Points further from the diagonal represent more persistent features, suggesting they are more significant to the underlying structure of the data.

A barcode representation provides the same information but in a different visual format. Each horizontal bar corresponds to a point in the persistence diagram, with the left endpoint representing birth time and the right endpoint representing death time. The length of each bar (death - birth) represents the persistence or lifespan of a topological feature. Longer bars typically indicate more significant features in the data's structure.

Figure 3.8 provides a didactic example of the filtration process on a simple point cloud. This visual illustration demonstrates how topological features emerge and disappear as the radius parameter increases. As the radius grows from $r = 0.3$ to $r = 3.0$, we observe the formation and eventual merging of clusters, the creation and filling of loops, and ultimately the construction of a densely connected complex. The persistence diagram in the lower right shows the corresponding birth and death coordinates of topological features identified during this filtration process.

TDA on Textual Dataset

Using topology provides analysis of the learned representations, such as the study of class means and their spatial relationships, which reveals the importance of considering the shape and organization of the data in the feature space. The concept of simplicial complexes, which capture the connectivity and higher-order interactions between data points, provides a powerful

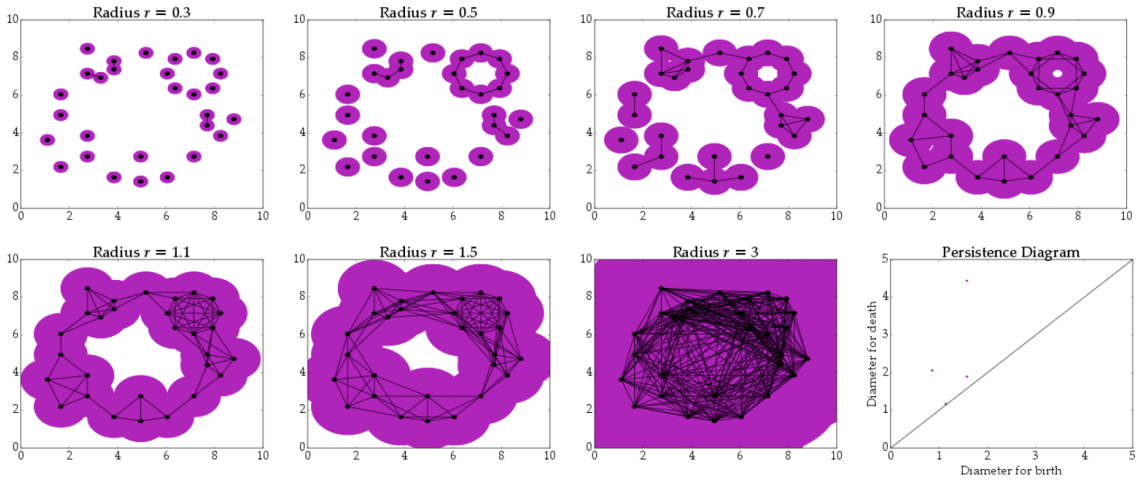


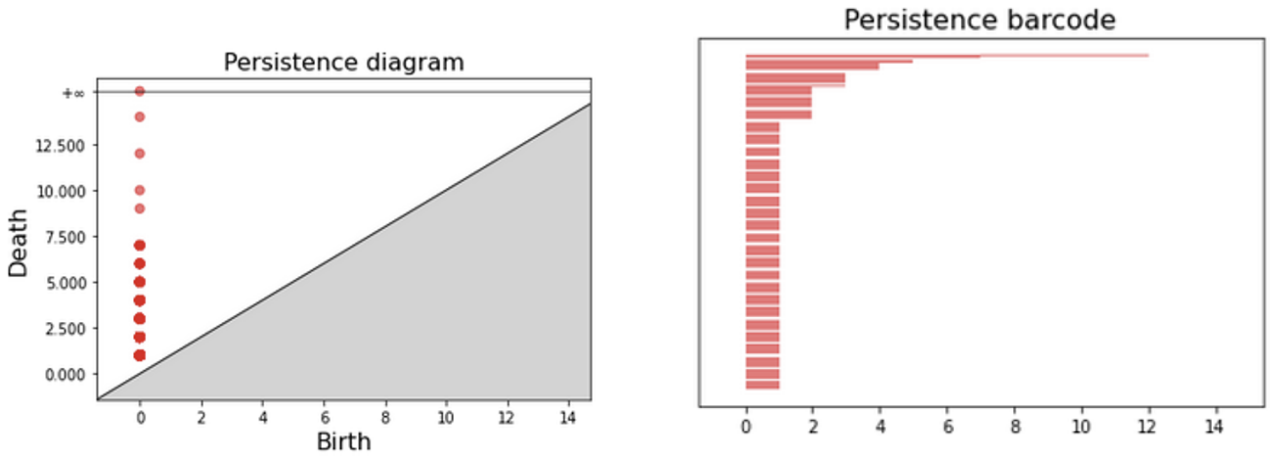
Fig. 3.8 Illustrative example of a series of filtrations on a set of point cloud data (Munch, 2017). As the radius r increases from 0.3 to 3.0, we observe the evolution of the simplicial complex structure. The persistence diagram (bottom right) records the birth and death times of topological features detected during this filtration process.

framework for studying the geometric properties of the learned representations. By investigating the formation and evolution of simplicial complexes during the learning process, this will gain insights into how the model captures and represents the underlying structure of the textual data.

To build intuition for how persistent homology captures structural information, we first present a simplified, conceptual example from prior work. This example (Figure 3.8) illustrates how topological features such as connected components and loops appear and disappear during a filtration process on a 2D point cloud. As the radius parameter r increases, the simplicial complex evolves, and the corresponding persistence diagram records the birth and death of features across scales. This illustrative figure provides conceptual grounding for interpreting topological features in more complex, high-dimensional settings.

Building on this conceptual foundation, we apply the same topological data analysis method to the high-dimensional activation space of our CNN model. Using the Python library *gudhi* (Maria, 2016), we analyzed the activations of the penultimate layer of the CNN for the dataset that is used for training and testing. Unlike the simple point cloud in Figure 3.8, our analysis involves high-dimensional activation vectors that cannot be directly visualized, making TDA particularly valuable for uncovering structural patterns in this context.

Figure 3.9 (Fig. 3.10) presents persistent homology analysis of actual CNN activations from our experiment. The persistence diagram (Figure 3.9a) and barcode (Figure 3.9b) represent the same topological information extracted from the penultimate layer of our trained CNN model.



(a) Persistence diagram

(b) Persistence barcode

Fig. 3.9 Persistent homology analysis of CNN activations.

The persistence diagram from our experimental results shows birth and death filtration values of topological features in our CNN's activation space. Points further from the diagonal represent more persistent features. The two notable points between 7.5 and 10 on the death axis correspond to two distinct sub-classes that were combined into one in our data labeling.

The barcode representation of the same topological features, derived from our CNN's activation vectors, visualizes the persistence diagram as a collection of horizontal bars. Each bar corresponds to a point in the diagram. The longer bars at the top indicate significant topological features, suggesting the presence of more than 4 major clusters in our activation space.

Together, Figures 3.8 and 3.9 provide a conceptual-to-applied bridge: the former explains how topological features emerge in theory, and the latter shows how those features manifest in real neural activation spaces. These visualizations demonstrate that the CNN's representation space captures rich and nuanced structures, including sub-clusters beyond the labeled classes. This underscores TDA's power in revealing features not immediately evident from standard classification results.

Discussions

The distribution of the misclassified labels is shown in Figure 3.10. This means that over 70% of the misclassified samples are classified as *world* by the NC network. A small sample of texts from the topics of *Sports*, *Business*, and *Science/Technology* that have been misclassified as *World* are shown in Table 3.4. It can be observed that they all contain geographical locations in their texts. For example, under the topic of *sports*, the texts contain words such as 'Atlanta', 'Minnesota'. Also, under the topic of *business*, there are 'New York' and 'London'. Under

science/technology, there are 'Southern California', and also the word 'Canadian'. It seems that the model relates these geographic names to the topic of *world*. However, in the context of the dataset, this classification is incorrect.

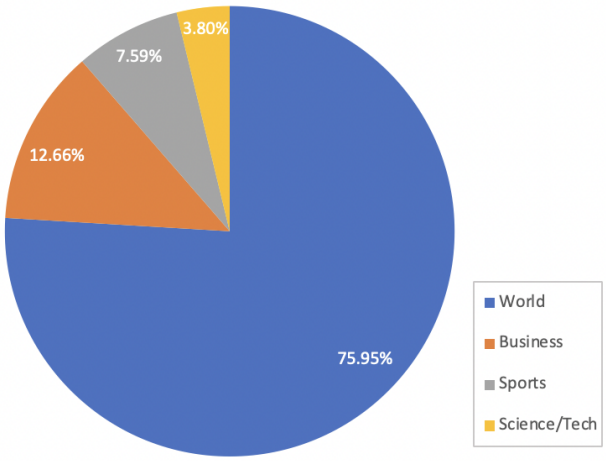


Fig. 3.10 Distribution of (incorrect) labels by model

Now consider some examples from the topic *world* that have been misclassified are shown in Table 3.5. These texts also contain geographic locations. So it cannot be the deciding factor here.

We notice that in both tables, there are text examples about oil (petrol) that was classified wrongly by the NC model. In Table 3.4, under the *business* category, the second text example talks about oil prices reported in London and discusses its impact on other countries. But the model has incorrectly labeled it as *business*. Furthermore, in Table 3.5, when it talks about oil in New York, the model has classified it as *business*. In terms of news context, one discusses more about the global impact of oil prices which would be closer to *world*, and the other is more of local impact of oil prices.

Based on this analysis, it seems that the label *world* can be split into two: one is **global** and the other is **local**. Both of them combined and the major topic would be **contextual impact**, instead of *world*.

What this seems to indicate is that we can make use of NC to help analyze the characteristics of the dataset when it comes to text classification.

3.2.4 Implications for Data Quality and Augmentation

The analysis of misclassified samples and the discovery of potential hidden clusters through TDA highlight the complexity of textual data and the challenges in capturing its full nuances.

Table 3.4 Samples Misclassified as Topic 1 (World)

Correct Topic	Misclassified Examples
Sports	Outfielder J.D. Drew missed the Atlanta Braves' game against the St. Louis Cardinals on Sunday night with a sore right quadriceps. The Cleveland Indians pulled within one game of the AL Central lead by beating the Minnesota Twins, 7-1, Saturday night with home runs by Travis Hafner and Victor Martinez.
Business	NEW YORK (Reuters) - U.S. Treasury debt prices slipped on Monday, though traders characterized the move as profit-taking rather than any fundamental change in sentiment. LONDON (Reuters) - Oil prices struck a fresh record above $48 a barrel on Thursday, spurred higher by renewed violence in Iraq and fresh evidence that strong demand growth in China and India has not been slowed yet by higher energy costs.
Sci/Tech	Southern California's smog-fighting agency went after emissions of the bovine variety Friday, adopting the nation's first rules to reduce air pollution from dairy cow manure. Vermont's Republican governor challenged the Bush administration's prescription drug policy in federal court yesterday, marking the first time a state has chosen a legal avenue in the expanding battle over Canadian imports.

Table 3.5 Incorrect Classification for Class 1 (World) Samples

NC Label	Text Examples (World)	Title
Business	The price of oil has continued its sharp rise overnight, closing at a record high. The main contract in New York, light sweet crude for delivery next month, has closed at a record $US46.75 a barrel - up 70 cents on yesterday.	Oil prices bubble to record high
Sports	Students at the Mount Sinai School of Medicine learn that diet and culture shape health in East Harlem.	Future Doctors, Crossing Borders

These findings point to a critical need for more sophisticated data handling techniques in NLP (Wei and Zou, 2019). Specifically, they underscore the potential value of data augmentation in:

- Enhancing the diversity of training data to better capture the complexity of language (Feng et al., 2021).
- Addressing the limitations of predefined class labels by introducing more varied examples (Shorten and Khoshgoftaar, 2019).

Possible Further Applications

Text classification models are traditionally trained by supervised learning (SL) techniques. However, the process of labelling data by human annotators can be time-consuming and expensive. When limited labelled data are available, semi-supervised learning (SSL) is used. SSL is a combination of both supervised and unsupervised learning. It allows the model to integrate the data that has not been categorized. The goal of SSL is to maximize the learning performance through pre-labeled and newly discovered labels and these newly-labeled examples (Hady and Schwenker, 2013).

The experimental process in this paper could be utilized to uncover hidden clusters. A NC model trained on labelled textual data. Unlabelled data could then be used as test data to this model. Some of them would be clustered to the existing class means. So these samples are then automatically labelled. The remaining samples could be analyzed using TDA on the activation pattern of the penultimate layer. Those forming new clusters could be assigned new labels.

Overall, discovery of potential hidden clusters within the data through TDA indicates that the predefined class labels may not always capture the full complexity and nuances of the textual data. This finding suggests that there is room for improvement in the way we approach data labeling and annotation in NLP tasks. It also highlights the potential of using techniques like NC and TDA can be used for for cluster discovery and semi-supervised learning in NLP.

3.3 Summary

The study on NC in text classification tasks has unveiled significant insights that not only establish a foundation for the subsequent chapters of this thesis but also directly address **RQ1**.

Our experiments have confirmed that NC does indeed occur in NLP models, particularly within the context of text classification using CNNs. This discovery broadens the applicability of NC beyond its initial observation in computer vision, offering new opportunities to understand

the learning dynamics and generalization properties of deep learning models in NLP. More importantly, it provides a concrete answer to **RQ1** by demonstrating how topological approaches can reveal the internal structure of neural networks processing text data. The focus on NC emphasizes the importance of analyzing the physical structure and geometric properties of learned representations in neural networks. By examining how features converge to a simple, well-defined structure in the final layers, this study provides a novel perspective on the behavior and generalization capabilities of NLP models.

The application of TDA, persistent homology and simplicial complexes offers a complementary approach to traditional vector space representations, enriching our understanding of the inherent relationships and patterns within the data. It also shows how topological approaches can provide insights into the inner workings of deep neural networks.

Two pivotal insights emerge from this study:

- **Topological Perspective on Model Behavior and Explainability:** The convergence of features to a simplified structure through NC suggests new pathways for enhancing model interpretability. It demonstrates how topological approaches can elucidate the physical structure of neural networks, providing a clearer understanding of their decision-making processes.

- **Data Quality and Model Performance:** The analysis of misclassified samples underscores the critical role of training data quality in the performance and generalization of NLP models. This finding emphasizes the importance of advanced data augmentation techniques that not only increase the quantity of data but also enhance its quality and diversity. From a topological perspective, this insight reveals how the structure of the input data influences the internal representations learned by the network, further addressing **RQ1** by showing how topological methods can link input data characteristics to model behavior.

These insights not only validate the relevance of NC in NLP but also provide a strategic direction for the following chapters. By demonstrating how topological approaches can reveal the inner workings and physical structure of deep neural networks processing textual data.

The next chapter will delve into manifold theory as applied to textual data, laying the groundwork for selecting appropriate dimensionality reduction methods in our ongoing exploration of topology and geometry in NLP.

Chapter 4

Manifold Theory of Textual Data

In Chapter 3, the exploration of NC uncovered potential hidden clusters within the data, suggesting that predefined class labels may not always capture the full complexity of textual information. This discovery highlighted the need for more sophisticated data handling techniques in NLP, particularly in data representation and augmentation.

Building on these findings, this chapter applies manifold theory to textual data, aiming to establish a justifiable approach for selecting the optimal number of principal components (k) in dimensionality reduction. Principal Component Analysis (PCA) is used to tackle the challenges of data quality and model performance identified in Chapter 3. The findings here pave the way to achieving our overall objective in developing more resource-efficient methods stated in Section 1.3.2.

This chapter begins by discussing the concept of manifold theory of NLP. It allows us to conceptualize how word embeddings distribute as points on a high-dimensional manifold. The core of our analysis focuses on applying PCA to augmented datasets. The optimal number of principal components required to effectively capture the critical aspects of the data is determined. How these components influence data distribution within the PCA space is then examined. By correlating PCA outcomes with augmentation performance, we aim to provide insights into the relationship between data structure and NLP task effectiveness.

4.1 Manifold Representations of Language Data

Word embeddings have revolutionized the way we represent and process textual data. These dense vector representations capture semantic and syntactic information about words, enabling various downstream tasks such as sentiment analysis, named entity recognition, and machine

translation (Mikolov et al., 2013a; Pennington et al., 2014). However, the high dimensionality of word embeddings often poses challenges in terms of interpretability and computational efficiency, necessitating the application of dimensionality reduction techniques.

Manifold theory, a branch of mathematics that studies topological spaces locally resembling Euclidean space, provides a powerful framework for understanding and analyzing the structure of high-dimensional data in NLP. The manifold hypothesis posits that even though real-world data is high-dimensional, it often lies on or near a lower-dimensional manifold (Fefferman et al., 2016). This hypothesis has profound implications for how language data is conceptualized and processed, particularly in the context of word embeddings and data augmentation. The application of manifold theory to NLP offers a geometric interpretation of language that provides deep insights into the structure and relationships within textual data. At its core, this framework posits that word embeddings can be viewed as points on a high-dimensional manifold (Hashimoto et al., 2016). This theoretical framework serves as the backbone for exploring how different dimensionality reduction techniques can be optimized and applied to textual data, ensuring that NLP models derived from these techniques are both efficient and interpretable.

In the context of word embeddings, each word in a vocabulary is mapped to a high-dimensional vector, typically consisting of hundreds of dimensions (Mikolov et al., 2013b). The high-dimensional space in which these word embeddings reside can be conceptualized as a manifold. On this manifold, each point represents a word, and the geometric relationships between these points capture the semantic and syntactic relationships between the corresponding words. For instance, words with similar meanings or those belonging to the same semantic category tend to cluster together on the manifold (Pennington et al., 2014).

This geometric interpretation allows for the application of tools and concepts from differential geometry and topology to analyze and manipulate word embeddings. The curvature, dimensionality, and topology of the word embedding manifold provide insights into the intrinsic structure of the language space (Nickel and Kiela, 2017). By understanding these geometric properties, we can gain a deeper understanding of the organization and relationships of words in the embedding space.

Moreover, manifold theory offers a theoretical foundation for various operations on word embeddings. Techniques such as dimensionality reduction, which is a focus of this chapter, can be understood as projections or mappings of the high-dimensional manifold onto lower-dimensional spaces. Other operations, such as interpolation between word vectors or the computation of geodesic distances on the manifold, can reveal semantic paths between concepts. They can

also measure semantic similarity in ways that account for the curvature of the semantic space (Camacho-Collados and Pilehvar, 2018). The manifold hypothesis in NLP suggests the following principles:

- Semantic Continuity: Small movements along the manifold correspond to small changes in meaning or grammatical function (Mikolov et al., 2013c). This principle underlies many word analogy tasks and semantic interpolation techniques.
- Intrinsic Low Dimensionality: Despite the high-dimensional nature of word embeddings, the intrinsic dimensionality of language data is often much lower (Linzen, 2019). This motivates the use of dimensionality reduction techniques to uncover this lower-dimensional structure.
- Non-linear Structure: The manifold of language data is often inherently non-linear, reflecting the complex and nuanced relationships between words and concepts (Bengio et al., 2013). This non-linearity challenges simple linear models and motivates the use of more sophisticated techniques for analysis and processing.

Understanding NLP data through the lens of manifold theory not only provides theoretical insights but also has practical implications for data augmentation and processing. When performing data augmentation, it can be conceptualized as generating new points on or near the language manifold. Effective augmentation techniques should, therefore, preserve the local and global structure of this manifold while introducing beneficial diversity.

In the subsequent sections of this chapter, we will explore how this theoretical framework informs the approach to dimensionality reduction, particularly in the context of analyzing augmented datasets. By grounding the analysis in manifold theory, the aim is to provide a more nuanced and geometrically informed perspective on the challenges and opportunities in processing and augmenting NLP data.

4.2 Comparison of Dimensionality Reduction Techniques

Dimensionality reduction is a crucial step in processing high-dimensional data such as word embeddings in NLP. This section explores three prominent dimensionality reduction techniques: t-Distributed Stochastic Neighbor Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP), and PCA. Each method offers unique advantages and challenges when applied to NLP tasks, particularly in the context of data augmentation.

t-SNE t-SNE is a nonlinear dimensionality reduction technique particularly well-suited for visualizing high-dimensional data (Van der Maaten and Hinton, 2008). t-SNE works by converting similarities between data points to joint probabilities and minimizes the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data.

The t-SNE can be described mathematically as follows. Let $x_i \in \mathbb{R}^D$ be high-dimensional data points and $y_i \in \mathbb{R}^d$ be their low-dimensional representations. The similarity between two high-dimensional points x_i and x_j is given by:

$$p_{j|i} = \frac{\exp(-|x_i - x_j|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-|x_i - x_k|^2/2\sigma_i^2)} \tag{4.1}$$

The similarity between two low-dimensional points y_i and y_j is given by:

$$q_{ij} = \frac{(1 + |y_i - y_j|^2)^{-1}}{\sum_{k \neq l}(1 + |y_k - y_l|^2)^{-1}} \tag{4.2}$$

t-SNE minimizes the Kullback-Leibler divergence between these two distributions:

$$C = KL(P|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4.3}$$

Its strength lies in creating visually appealing and interpretable 2D or 3D representations that excel at revealing clusters and local structures in the data. This makes t-SNE particularly useful for exploratory data analysis in NLP, where it can help uncover semantic relationships between words or documents (Van Der Maaten, 2009). Additionally, t-SNE's ability to capture nonlinear relationships is crucial for complex linguistic data, allowing it to represent subtle semantic nuances that linear methods might miss (Wang et al., 2021).

However, t-SNE is not without its drawbacks. Its computational complexity can be a significant limitation, especially when dealing with large NLP datasets or extensive vocabularies (Van Der Maaten, 2014). While t-SNE excels at preserving local structures, it may not accurately represent global structures or distances between clusters, which can be problematic for tasks that rely on understanding overall relationships in the data (Wattenberg et al., 2016). The stochastic nature of the algorithm can also lead to varying results between runs, potentially affecting reproducibility (Kobak and Berens, 2019). Moreover, the sensitivity of t-SNE to hyperparameters, particularly the perplexity value, necessitates careful tuning to achieve optimal results (Cao and Wang, 2017). Despite these limitations, t-SNE remains a valuable tool in the NLP toolkit,

particularly for tasks where visualization and exploration of local semantic structures are primary goals.

UMAP UMAP is a more recent dimensionality reduction technique that addresses some of the limitations of t-SNE (McInnes et al., 2018). UMAP is based on manifold learning techniques and topological data analysis.

Let $X = x_1, ..., x_n$ be the high-dimensional data and $Y = y_1, ..., y_n$ be the low-dimensional representation. UMAP constructs a weighted graph in both the high and low-dimensional spaces. The edge weights in the high-dimensional space are given by

$$w_{ij} = \exp(-(\frac{d(x_i, x_j) - \rho_i}{\sigma_i})) \tag{4.4}$$

where $d(x_i, x_j)$ is the distance between points, ρ_i is the distance to the nearest neighbor of x_i, and σ_i is a normalizing factor. UMAP then optimizes the low-dimensional representation to minimize the cross-entropy between the high and low-dimensional graphs:

$$C = \sum_{i,j}[w_{ij} \log(\frac{w_{ij}}{v_{ij}}) + (1 - w_{ij}) \log(\frac{1 - w_{ij}}{1 - v_{ij}})] \tag{4.5}$$

where v_{ij} are the edge weights in the low-dimensional space.

One of UMAP's key advantages is its scalability; it can handle much larger datasets than t-SNE, making it particularly suitable for big NLP corpora with extensive vocabularies (McInnes et al., 2018). Unlike t-SNE, UMAP better preserves both local and global structures in the data, which is crucial for understanding overall relationships in word embeddings or document vectors (Becht et al., 2019). This ability to maintain global structure alongside local relationships makes UMAP valuable for tasks where understanding the broader context of linguistic elements is important. UMAP's has a strong mathematical foundation in manifold learning and topological data analysis, which gives a rigorous justification for its approach (McInnes et al., 2018). This theoretical grounding not only supports its effectiveness but also allows for its use beyond mere visualization. UMAP can serve as a general dimensionality reduction technique for various downstream NLP tasks, potentially improving performance in areas such as text classification or clustering (Stolarek et al., 2022).

However, UMAP is not without its challenges. While faster than t-SNE, it still involves higher computational complexity than linear methods like PCA (McInnes et al., 2018). The theoretical foundations of UMAP, while rigorous, can be more challenging to interpret intuitively

compared to simpler methods, which may be a consideration for some applications (Becht et al., 2019). Like t-SNE, UMAP's results can be sensitive to parameter choices, particularly the number of neighbors and minimum distance parameters, requiring careful tuning (Kobak and Linderman, 2019). Additionally, the stochastic nature of UMAP means that results can vary between runs, potentially affecting reproducibility in some scenarios (McInnes et al., 2018).

Despite these considerations, UMAP's ability to handle larger datasets, preserve global structure, and potentially improve downstream task performance makes it an increasingly popular choice in NLP, particularly for tasks involving large vocabularies or where relationships between distant words or documents are important.

PCA PCA is a linear dimensionality reduction technique that has been widely used in many applications including NLP (Jolliffe and Cadima, 2016). At its core, PCA works by identifying the principal components – orthogonal directions in the data that capture the most variance. In the context of word embeddings, PCA can be particularly useful for reducing the high-dimensionality of these representations while preserving their most salient features (Raunak et al., 2019). The technique operates by projecting the original high-dimensional data onto a lower-dimensional subspace, effectively compressing the information into fewer dimensions.

The mathematical formulation of PCA is as follows. Given a data matrix $X \in \mathbb{R}^{n \times p}$ where n is the number of samples and p is the number of features, PCA seeks to find a transformation matrix $W \in \mathbb{R}^{p \times k}$ that projects the data onto a k-dimensional subspace:

$$Y = XW \tag{4.6}$$

where $Y \in \mathbb{R}^{n \times k}$ is the reduced-dimensional representation. The columns of W, denoted as $w_1, ..., w_k$, are the eigenvectors of the covariance matrix $\Sigma = \frac{1}{n-1} X^T X$, corresponding to the k largest eigenvalues $\lambda_1, ..., \lambda_k$. These eigenvectors are called principal components. The proportion of variance explained by the i-th principal component is given by:

$$\frac{\lambda_i}{\sum_{j=1}^{p} \lambda_j} \tag{4.7}$$

In the context of word embeddings, X would represent the matrix of word vectors, with each row corresponding to a word and each column to a dimension of the embedding. After applying PCA, Y would represent the reduced-dimensional word embeddings.

One of PCA's main strengths in NLP applications is its ability to effectively reduce the dimensionality of word embeddings while preserving most of the information (Raunak et al., 2019). By focusing on the principal components that explain the most variance, PCA can help filter out noise in the data, potentially leading to more robust representations (Mu and Viswanath, 2018). This noise reduction aspect can be particularly beneficial in NLP, where word embeddings often contain spurious correlations or artifacts from the training data. Another advantage of PCA in the NLP context is its ability to provide insights into the most important features or directions in the embedding space (Lebret and Collobert, 2014). By examining the principal components, one can gain a better understanding of what the embeddings have learned and potentially uncover latent semantic or syntactic structures captured by the embeddings. Furthermore, PCA's computational efficiency compared to many non-linear dimensionality reduction techniques makes it suitable for large-scale NLP datasets, allowing for rapid analysis and processing of extensive vocabularies (Arora et al., 2018).

However, it is important to note that as a linear method, PCA would not be able to capture complex, nonlinear relationships in the data as effectively as techniques like t-SNE or UMAP. This limitation can be significant in NLP, where many semantic and syntactic relationships are inherently nonlinear. Despite this drawback, PCA remains a fundamental tool in NLP, often used as a first step in data analysis or as a preprocessing step before applying more complex methods. Its simplicity, interpretability, and efficiency make it particularly useful for understanding the broad structure of word embeddings and for efficient dimensionality reduction in various NLP tasks, ranging from text classification to information retrieval.

In practice, when applying PCA to word embeddings, each word vector is treated as a data point in high-dimensional space. The technique then finds the directions (principal components) along which the variation in the data is maximal. By projecting the word vectors onto these principal components, we obtain a lower-dimensional representation that still captures the most important aspects of the original embeddings. This can lead to more efficient downstream processing and can often improve the performance of NLP models by focusing on the most informative dimensions of the word representations.

While t-SNE and UMAP offer powerful visualization capabilities and are able to capture non-linear relationships in data, this study opted to use PCA for the following reasons:

- **Linear Interpretability**: PCA's linear nature allows for straightforward interpretation of the resulting components. Each principal component represents a direction of maximum variance in the data, which can be directly related to the original features (Jolliffe and

Feature	**t-SNE**	**UMAP**	**PCA**
Linearity	Non-linear	Non-linear	Linear
Scalability	Poor for large datasets	Good for large datasets	Excellent for large datasets
Preserves global structure	Poor	Good	Excellent
Preserves local structure	Excellent	Excellent	Moderate
Interpretability	Low	Moderate	High
Computational complexity	High	Moderate	Low
Stochastic nature	Yes	Yes	No
Best use case	Visualization of high-dimensional data	Visualization and general dimensionality reduction	Feature extraction and noise reduction

Table 4.1 Summary comparison of t-SNE, UMAP, and PCA

Cadima, 2016). This interpretability is crucial for understanding how data augmentation affects the underlying structure of word embeddings.

- **Computational Efficiency**: PCA is significantly more computationally efficient than t-SNE or UMAP, especially for large datasets (Raunak et al., 2019). Given the scale of modern NLP datasets and the need to perform multiple analyses across different augmentation techniques, PCA's efficiency is a substantial advantage. This also ties in with one of the limitations discussed in this thesis in regards to computational efficiency in NLP models.

- **Deterministic Results**: Unlike t-SNE and UMAP, which can produce different results across runs due to their stochastic nature, PCA is deterministic (Wattenberg et al., 2016). This consistency is vital for ensuring reproducibility of our results and for making reliable comparisons across different augmentation methods.

- **Preservation of Global Structure**: While t-SNE excels at preserving local structure, it can distort global relationships (McInnes et al., 2018). PCA, on the other hand, preserves global structure, which is crucial for understanding how augmentation affects the overall distribution of word embeddings.

- **Variance Explanation**: PCA provides a clear measure of the amount of variance explained by each component (Mu and Viswanath, 2018). This allows us to quantify how much information is retained in the reduced-dimensional space, which is particularly useful for comparing the effects of different augmentation techniques.

While non-linear methods like t-SNE and UMAP may capture complex relationships that PCA might miss, the advantages of PCA in terms of interpretability, efficiency, and consistency make it the most suitable choice for our specific research goals. The linear nature of PCA aligns well with our objective of understanding how data augmentation affects the fundamental structure of word embeddings, rather than focusing on complex, potentially non-linear relationships that might be specific to particular datasets or augmentation methods.

4.3 Optimal Number of Principal Components

When applying PCA for dimensionality reduction of word embeddings, an important decision is the choice of the number of principal components to retain, often denoted as k. This choice determines the dimensionality of the reduced embedding space and impacts the balance between information preservation and computational efficiency. In practice, it is common for researchers to arbitrarily choose a value for k, such as 2 or 3, without providing a clear justification for their selection (Heimerl and Gleicher, 2018; Raunak et al., 2019). The lack of justification for the choice of k can be problematic from both theoretical and explainable AI perspectives. Theoretically, the optimal value of k depends on the intrinsic dimensionality of the data, which reflects the minimum number of variables needed to accurately represent the underlying structure (Camastra and Staiano, 2016). Choosing a value of k that is too low may result in a loss of important information and oversimplification of the embedding space. On the other hand, a k value that is too high may retain noise and redundant dimensions, hindering interpretability and computational efficiency.

In the context of word embeddings, explaining the rationale behind the choice of k is crucial for building trust in the reduced representations and the subsequent analyses based on them. Thus, researchers should provide clear justifications for their selected value of k and consider using principled methods to determine the appropriate dimensionality. By doing so, they can enhance the transparency, interpretability, and reliability of their findings.

4.3.1 Methodology

Data Augmentation Experiment Overview

In Chapter 5, we conducted a comprehensive data augmentation experiment that evaluated six different augmentation techniques across four NLP datasets, measuring their impact on text classification performance. In this chapter, we take the augmented datasets generated from those experiments and apply PCA analysis to understand how different augmentation methods affect the underlying geometric structure of text embeddings. For context, we provide a brief overview of that experiment here:

Augmentation Techniques: The experiment employed six different augmentation methods:

- Back-Translation (BT)
- Contextual Augmentation (CA)
- Easy Data Augmentation (EDA)
- Word2Vec-based augmentation (word replacement)
- GloVe-based augmentation (word replacement)
- GPT-J-based augmentation

Datasets: Four diverse NLP datasets were used:

- Question Type (QT) (Li and Roth, 2002): for question classification
- SNIPS (Coucke et al., 2018): for intent detection
- Stanford Sentiment Treebank (SST2) (Socher et al., 2013): for sentiment analysis
- Text REtrieval Conference (TREC) (Hovy et al., 2001): for information retrieval

While the experiment in Chapter 5 evaluated these augmentation techniques across the datasets, measuring their impact on model performance, particularly in terms of accuracy improvements. In this chapter, we utilize PCA to examine how these augmentation methods affect the underlying structure of the data, and how these structural changes correlate with the performance gains observed in the previous chapter.

For a detailed description of the augmentation techniques, datasets, and experimental setup, please refer to Chapter 5. A brief summary of key aspects is briefly mentioned below in 4.3.1. Our focus in this chapter is on applying PCA to the augmented data to gain insights into the dimensionality and distribution changes introduced by various augmentation methods.

Experimental Framework Overview

To provide context for our PCA analysis, we briefly summarize the key aspects of our experimental setup:

Augmentation Sample Generation: For each of the four datasets, we generated augmented samples using six different methods: Back-Translation (BT), Contextual Augmentation (CA), Easy Data Augmentation (EDA), Word2Vec-based augmentation, GloVe-based augmentation, and GPT-J-based augmentation. These methods span a range of approaches from simple word substitution to complex neural generation.

Embedding Extraction: For each original and augmented text sample, we extracted word embeddings using pre-trained models corresponding to each augmentation method (e.g., Word2Vec embeddings for Word2Vec-augmented data).

Classification Model: A convolutional neural network (CNN) with the architecture described in Section 5.1.5 of Chapter 5 was used to evaluate classification performance on both original and augmented datasets.

Performance Evaluation: Each augmentation method was evaluated based on improvements in classification accuracy compared to the baseline model trained only on original data. These performance metrics were then correlated with PCA-derived measures to establish relationships between data geometry and augmentation effectiveness.

This experimental framework allows us to systematically investigate how different augmentation methods affect the underlying structure of the embedding space and how these changes correlate with classification performance improvements.

PCA Analysis Procedure

For each dataset and augmentation method combination, we perform the following steps:

1. Extract word embeddings for both the original and augmented datasets.
2. Apply PCA to these embeddings.
3. Analyze the resulting principal components, focusing on:
   (a) The number of components needed to explain a significant portion of the variance.
   (b) The distribution of the data in the space of the first few principal components.

We then correlate these PCA results with the accuracy improvements reported in Chapter 5 to gain insights into how changes in the data's structure relate to performance gains.

Four principal components (PCA=4) were chosen to be examined. This is due to having a more comprehensive analysis because many studies focus on two or three components, extending to four will help to capture potentially important patterns that might be missed with fewer components (Jolliffe and Cadima, 2016). Including more than 4 components would incrementally increase the total variance explained, but at the cost of higher dimensionality and increased computational complexity, potentially compromising the efficiency gains of dimensionality reduction.

Analytical Tools

Scree Plots One crucial step in PCA is determining the optimal number of principal components to retain, balancing the need to explain the majority of the variance in the data while avoiding overfitting (Cattell, 1966). Screeplots, which plot the eigenvalues or the percentage of variance explained by each principal component in descending order, are a graphical tool to aid in this decision (Jolliffe and Cadima, 2016).

The scree plot is defined as:

$$S(k) = \lambda_k \text{ vs. } k \tag{4.8}$$

where λ_k is the kth eigenvalue and k is the principal component index.

The *elbow point* in a scree plot marks the point of diminishing returns in terms of variance explained by additional principal components. Mathematically, it can be defined as the point of maximum curvature on the scree plot (Thorndike, 1956):

$$\text{Elbow} = \arg\max_k S''(k) \tag{4.9}$$

where $S''(k)$ is the second derivative of the scree function. We use the elbow point to determine the optimal number of principal components to retain for our analysis.

In our implementation, the elbow point is automatically identified and marked with a green line on all scree plots (Figures 4.1 to 4.4). This calculation is performed using the kneed Python library, which implements the knee/elbow detection algorithm based on the method proposed by Satopaa et al. (2011). The library identifies the point of maximum curvature by fitting piecewise linear functions and finding the point where the difference between the two segments is maximized. This automated approach ensures consistent and objective identification of the optimal number of principal components across all datasets and augmentation methods.

4.3.2 Results and Discussions

Figures 4.1, 4.2, 4.3 and 4.4 present the screeplots for various datasets augmented with different techniques.

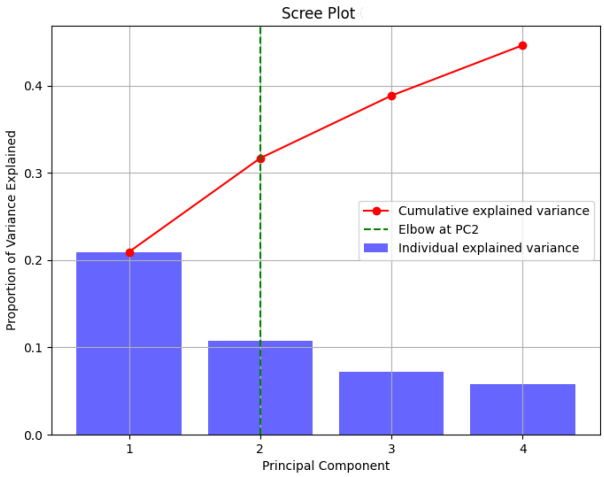
Overall, there is a drastic decrease which consistently appears at the 2nd principal component, regardless of the augmentation technique applied. This is also indicated by the green mark on the figures. While PCA=3 or PCA=4 would capture more variance, the gains are marginal compared to the potential increased computational cost.

As evident in the screeplots, PC3 and PC4 individually contribute less than 10% to the explained variance, whereas PC1 and PC2 together explain 42-44% for the QT dataset (Figure 4.1), about 30-35% for SST2 (Figure 4.3) and TREC datasets (Figure 4.4). Meanwhile the analysis on the SNIPS dataset (Figure 4.2) is in Section 4.3.2

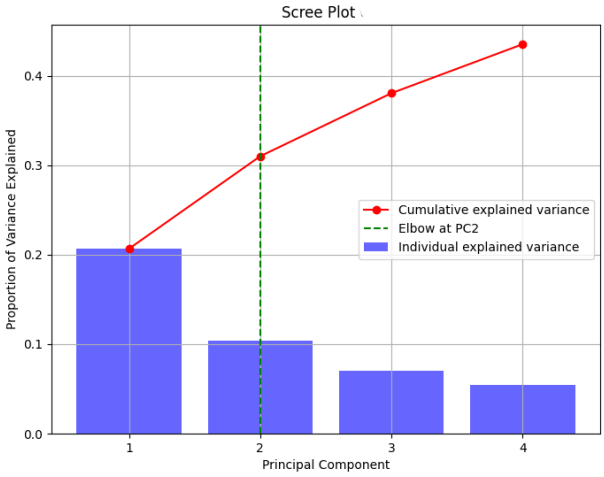
The elbow point at PC2 marks a clear transition where the rate of variance explained by each additional component begins to level off. For instance, in the TREC dataset, PC3 and PC4 each account for only about 5-7% of the variance, a small gain considering the added complexity. This approach significantly reduces computational complexity while preserving key features of the original embeddings.

The majority result across datasets and augmentation methods suggests that PCA=2 offers a robust, efficient dimensionality reduction strategy for various NLP tasks, balancing information preservation with computational efficiency.

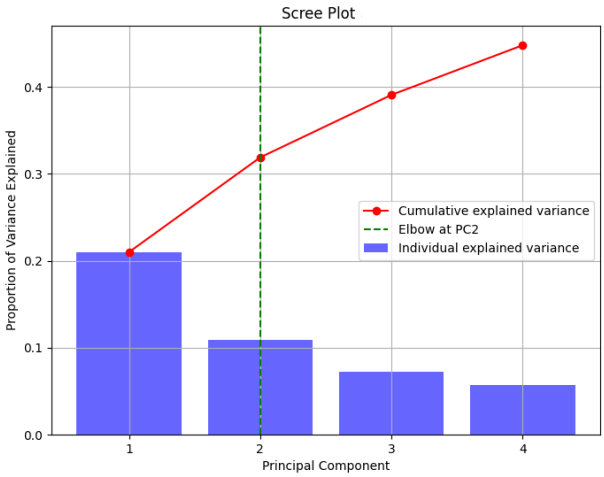
The findings presented here not only validate the use of PCA for dimensionality reduction in NLP but also highlight the nuanced relationship between data structure and task performance. These insights are critical for developing more effective and interpretable NLP models.



(a) QT BT

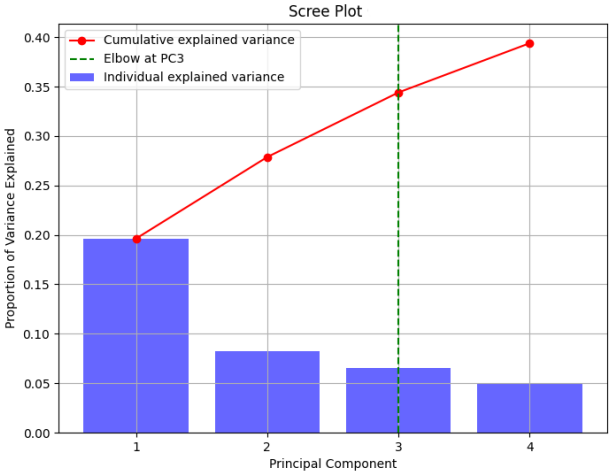


(b) QT CA

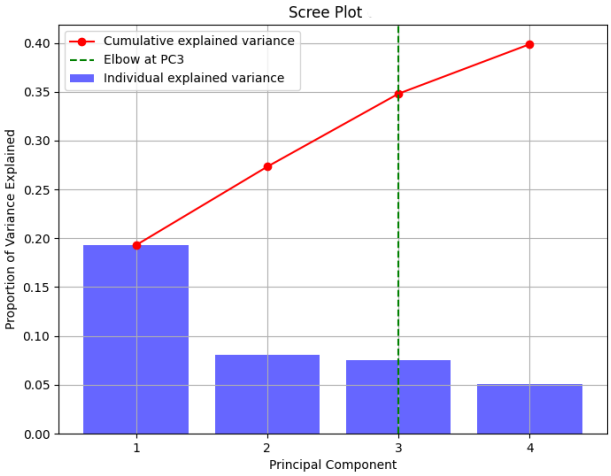


(c) QT EDA

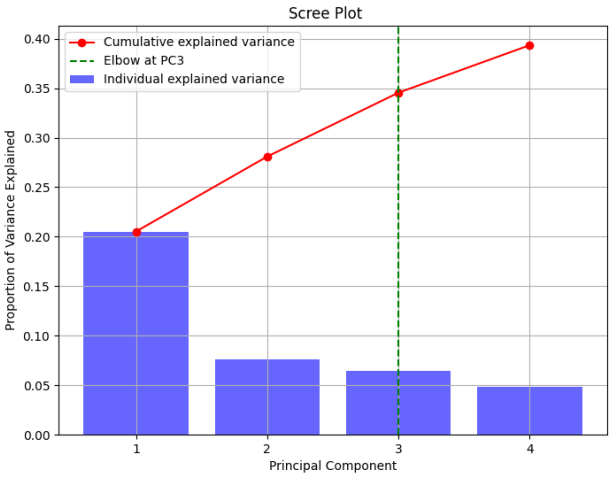
Fig. 4.1 Screeplots of Post-Augmentation PCA Components for QT Dataset



(a) SNIPS BT

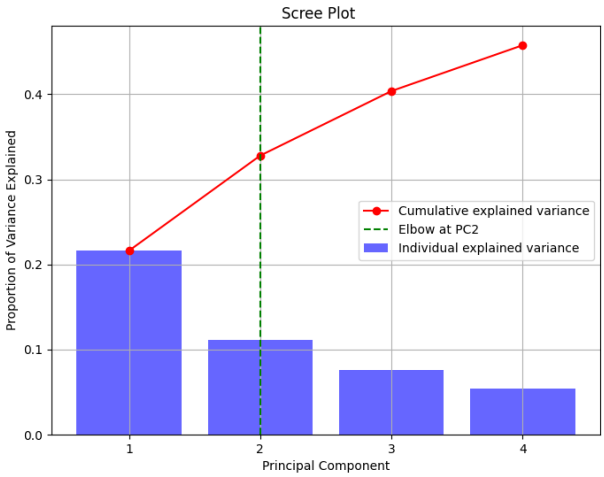


(b) SNIPS CA

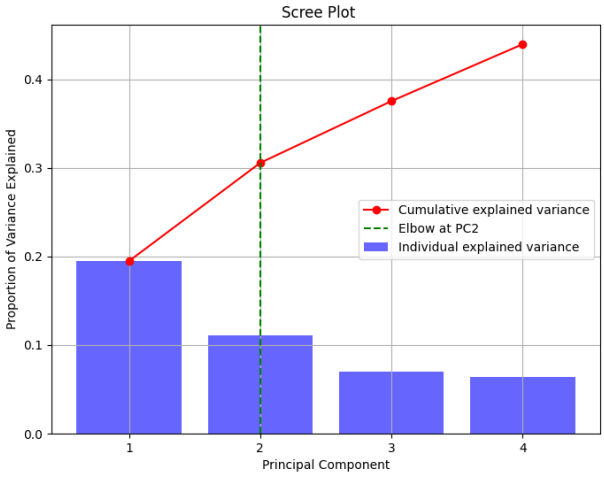


(c) SNIPS EDA

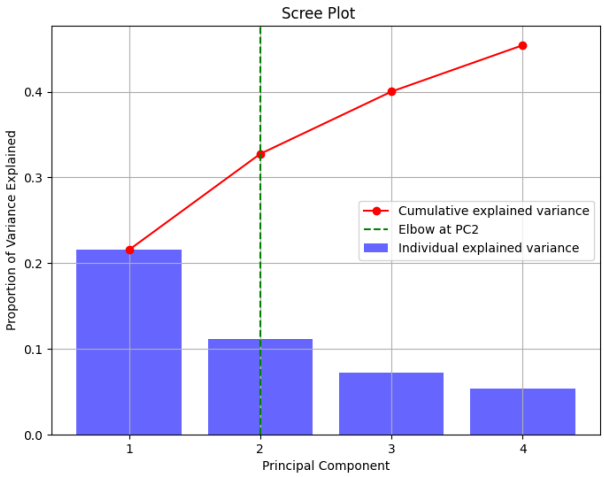
Fig. 4.2 Screeplots of Post-Augmentation PCA Components for SNIPS Dataset



(a) SST2 BT

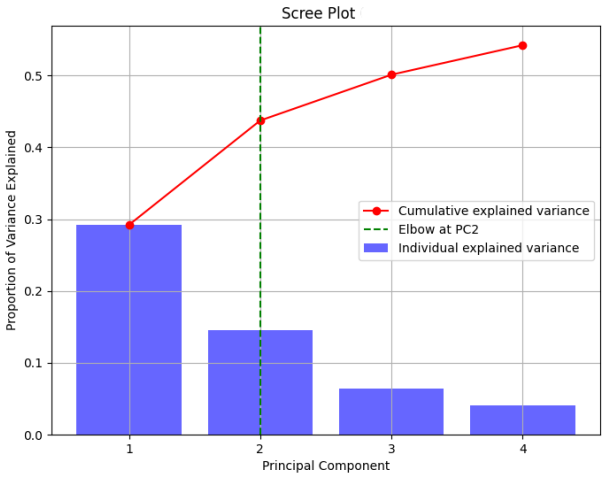


(b) SST2 CA

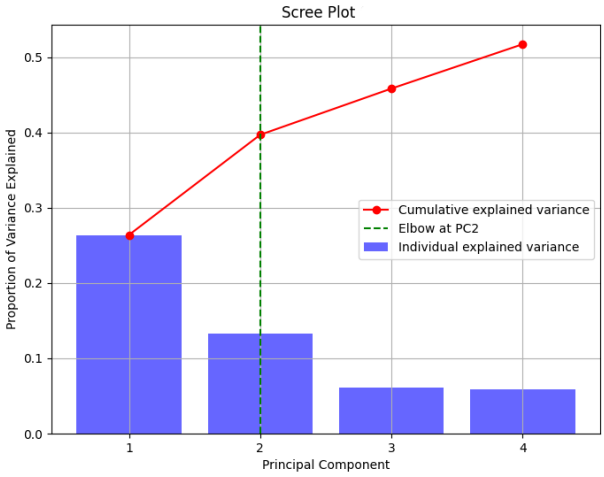


(c) SST2 EDA

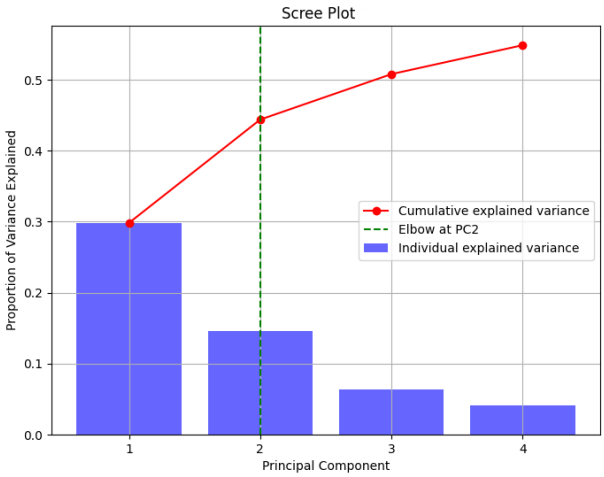
Fig. 4.3 Screeplots of Post-Augmentation PCA Components for SST2 Dataset



(a) TREC BT



(b) TREC CA



(c) TREC EDA

Fig. 4.4 Screeplots of Post-Augmentation PCA Components for TREC Dataset

Dataset feature differences

In the results for the SNIPS dataset (Figures 4.2a, 4.2b and 4.2c), it can be observed that the elbow point occurs at the third principal component instead of the second. This deviation is likely due to the inherent characteristics of the SNIPS dataset itself, rather than the augmentation methods applied, since these datasets have used all the same augmentation method. One crucial difference is that SNIPS is a dataset derived from voice commands, which have been transcribed to text. This process inherently introduces more variability due to factors such as:

- Variations in speech patterns, accents, and pronunciations.
- Potential transcription errors or inconsistencies.
- Capturing of filler words, pauses, or other speech nuances in the transcriptions.
- Domain-specific terms related to voice assistant tasks, which could introduce additional dimensions of variability not present in other datasets.

This exception highlights the importance of considering dataset-specific characteristics when determining the optimal number of principal components. While a two-component model may be sufficient for most augmented datasets, some techniques or datasets may benefit from retaining additional components to capture important variability.

Relationship Between Dimensionality Reduction and Task Performance

The choice of optimal dimensionality for PCA reduction directly impacts both the structural representation of text data and downstream task performance. Our analysis demonstrates this relationship in several key ways.

First, the elbow point in the scree plots (Figures 4.1, 4.2, 4.3, and 4.4) consistently appears at the second principal component across most datasets and augmentation methods. This indicates that the first two principal components capture the most significant variance in the data, with subsequent components providing diminishing returns. The practical implication is that PCA=2 offers an efficient dimensionality reduction strategy that preserves essential information while minimizing computational complexity.

The regression analysis presented in Table 4.2 further supports this finding. The marginal increase in R-squared from 0.275 to 0.277 when including the third principal component demonstrates that the additional complexity does not yield substantial improvements in explaining

variance related to classification performance. Moreover, the F-test results in Table 4.3 confirm that the inclusion of PCA3 does not provide a statistically significant improvement in explanatory power (p-value = 0.286).

This tight relationship between the optimal dimensionality and task performance can be explained through the lens of manifold theory. As discussed in Section 4.1, word embeddings can be viewed as points on a high-dimensional manifold, with semantically similar words clustered together. By identifying the principal directions of variation (PC1 and PC2), we effectively map this manifold to a lower-dimensional space that preserves the essential semantic relationships.

The practical benefits of this dimensionality reduction extend beyond computational efficiency. By focusing on the most informative dimensions, we filter out potential noise in the embedding space, allowing classification models to focus on the most relevant aspects of the data. This is particularly valuable in the context of data augmentation, where preserving semantic coherence is crucial.

The subsequent analysis in Section 4.4 will demonstrate how the distribution of data points in this reduced-dimensional space relates to augmentation effectiveness. The strong correlation between distributional characteristics in PCA space and classification accuracy improvements (Figure 4.6) provides further evidence that our dimensionality reduction approach effectively captures the structural elements most relevant to task performance.

Thus, our findings establish that a two-dimensional PCA reduction strikes an optimal balance between information preservation and computational efficiency for text augmentation analysis. This approach not only aligns with theoretical expectations from manifold theory but also demonstrates practical utility in predicting and explaining the effectiveness of different augmentation strategies.

4.3.3 Comparison of PCA2 and PCA3

While most datasets exhibited an elbow point at the second principal component, suggesting that a two-component model might be sufficient, the SNIPS dataset's elbow at the third component warranted further investigation. By comparing one principal component (PCA1), two components (PCA2) and three components (PCA3), we aimed to quantify the potential benefits of including the third principal component across all datasets. This approach allowed us to assess whether the additional complexity introduced by PCA3 provided significant improvements in explanatory power or model performance that would justify its inclusion. Furthermore, this comparison enabled us to make a more informed decision about whether to standardize our

approach using two components or to allow for dataset-specific flexibility in the number of components retained. Ultimately, this analysis served as a crucial step in determining the most appropriate dimensionality reduction strategy that could be applied consistently across all datasets while still accounting for the variability observed in cases like SNIPS.

Regression Models for Component Evaluation

To quantitatively assess the importance of each principal component in capturing meaningful variance in our textual data, we constructed regression models using principal components as predictors. This approach allows us to objectively evaluate whether including the third principal component (PCA3) provides significant additional explanatory power for the textual feature space beyond what is captured by the first two components.

In the context of our text classification tasks, these regression models help determine how effectively different dimensional representations of the text data correlate with classification performance. The dependent variable in our regression models is the accuracy improvement observed after augmentation (as measured in Chapter 5), while the independent variables are the principal components of the augmented data representations.

Two regression models were built: the first using only PCA1 and PCA2 as predictors, and the second including PCA3 as an additional predictor. By comparing these models, we can quantify the marginal contribution of PCA3 in explaining variance related to classification performance improvements. This comparative analysis addresses the key question: does the additional complexity introduced by including the third principal component provide meaningful benefits for representing the semantic structure of our augmented textual data?

Parameter	**Model (PCA1, PCA2)**	**Model (PCA1, PCA2, PCA3)**
Coefficient (PCA1)	+40.93	+40.00
Coefficient (PCA2)	-54.10	-42.33
Coefficient (PCA3)	–	-15.07
Intercept	3.37	3.43
R-squared	0.275	0.277

Table 4.2 Coefficients and Fit Statistics for Regression Models Predicting Augmentation Performance from Principal Components

Coefficient Interpretation in the Context of Text Data

The coefficients of the principal components in each model indicate the magnitude and direction of their influence on classification performance improvements for our textual data. As shown in Table 4.2, for the first model, PCA1 has a positive coefficient of +40.93, suggesting that textual features captured by this component contribute positively to augmentation effectiveness, while PCA2 has a negative coefficient of −54.10, indicating features that may counteract performance gains.

In the second model, the coefficients for PCA1 and PCA2 remain similar, while PCA3 shows a negative coefficient of −15.07. This suggests that the textual features uniquely captured by the third principal component have a moderate negative relationship with augmentation performance. The magnitude of these coefficients indicates that the semantic dimensions represented by PCA2 have the strongest influence on augmentation effectiveness, followed by those of PCA1 and PCA3.

Model Fit and Practical Significance for Textual Analysis

The marginal increase in R-squared from 0.275 in Model 1 to 0.277 in Model 2 indicates that including the third principal component provides only a slight improvement in explaining the variance in augmentation performance. This minimal increase raises questions about the practical value of retaining the third component when modeling the relationships between text embeddings and classification outcomes.

F-test Results

To assess the significance of including PCA3 in the model, an F-test was conducted and the results are shown in Table 4.3. The F-statistic of 1.29 measures the ratio of the variance explained by PCA3 to the unexplained variance. The corresponding p-value of 0.2860 indicates that the inclusion of PCA3 does not provide a statistically significant improvement in the model's explanatory power at the conventional significance level of 0.05.

F-Test Parameter	**Value**
F-statistic (PCA3)	1.29
p-value (PCA3)	0.286

Table 4.3 F-test Results for the Inclusion of PCA3

The F-statistic of 1.29 for the inclusion of PCA3, with a p-value of 0.286, indicates that the additional explanatory power brought by PCA3 is not statistically significant. The p-value being well above the conventional threshold of 0.05 suggests that the improvement in fit (as indicated by the minor increase in R-squared) does not justify the added complexity of including PCA3 in the model. The high p-value also implies that the improvement in the model's performance due to PCA3 is likely due to chance rather than a meaningful contribution. This lack of significance could be a result of PCA3 capturing aspects of the variability that are not as strongly associated with the response variable as those captured by PCA1 and PCA2.

Therefore, the experiments show that while PCA3 does contribute to explaining some variance in the dependent variable, its inclusion does not provide a statistically significant improvement in model performance. The results suggest that for practical purposes, especially when considering model parsimony and interpretability, Model 1 (using only PCA1 and PCA2) may be preferred. The slight increase in R-squared does not compensate for the additional complexity and potential overfitting that may arise from including an additional predictor with minimal impact. Therefore, the model with PCA1 and PCA2 alone may be sufficient to capture the main relationships between the principal components and the response variable.

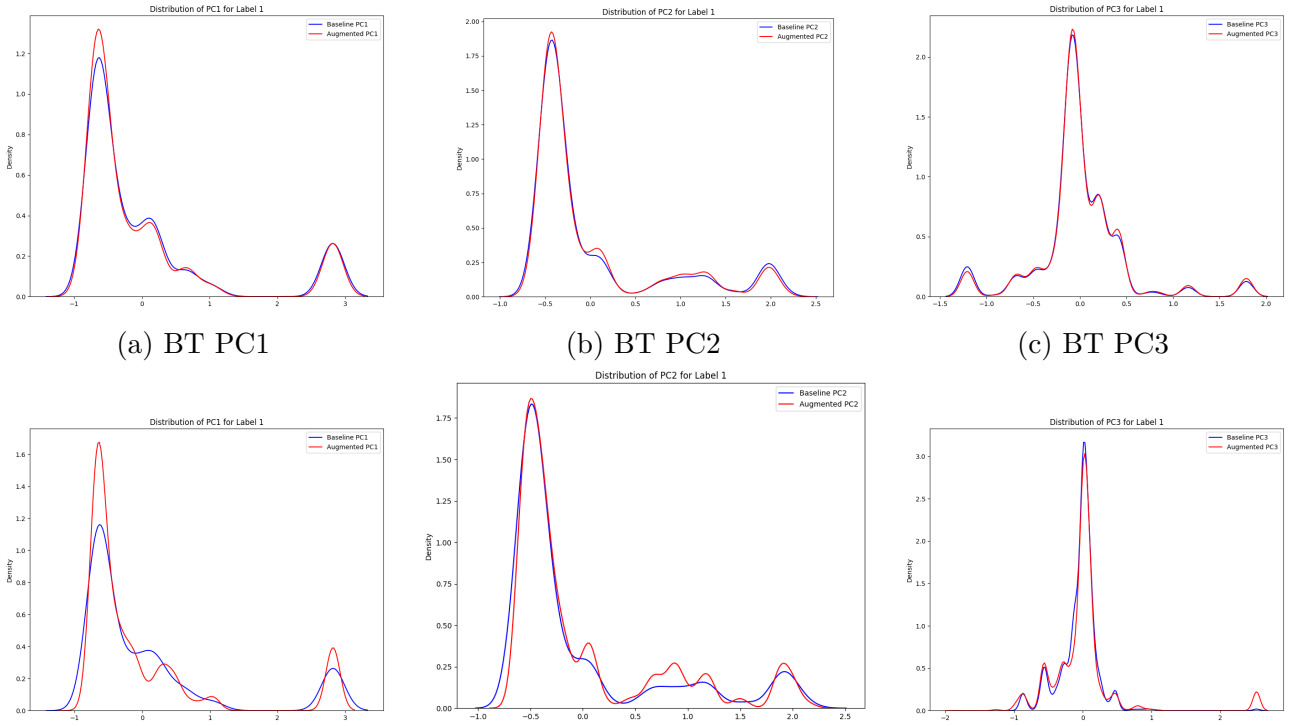
The next section explores deeper into the relationship of the distribution in PCA space.

4.4 Distribution in PCA Space

First, by applying PCA to the baseline word embeddings, we aim to identify the most important features that capture the semantic and syntactic relationships among words. This will provide a foundation for understanding the intrinsic structure of the embedding space. Second, by comparing the PCA results of the baseline and augmented data, we seek to assess the impact of data augmentation on the word embeddings. This comparison will shed light on whether data augmentation preserves the important features and relationships identified in the baseline analysis, or if it introduces any significant changes or improvements to the embedding space.

Figure 4.5 depicts density plots of the first three principal components (PC1, PC2, and PC3) for the baseline and augmented datasets of the dataset TREC, with backtranslation and contextual augmentation methods. These two are compared because they were respectively the best and worst performing augmentation methods.

Figures 4.5a, 4.5b and 4.5c show the distribution of backtranslation follows the pattern of the baseline. The peaks of the distributions closely align, and deviations are minors. This suggests



(a) BT PC1 (b) BT PC2 (c) BT PC3

(d) CA PC1 (e) CA PC2 (f) CA PC3

Fig. 4.5 Density plot of TREC (Label 1)

that the core structure of the data post-augmentation is preserved. In contrast, Figures 4.5d, 4.5e and 4.5f reveal less alignment between the augmented and baseline distributions for contextual augmentation. Divergence between the curves is more noticeable.

Since backtranslation is the best performing augmentation method, these results indicate that augmentated distributions that closely follow the pattern of the baseline are able to retain the meanings of the original text. The following key insights could be obtained from these results:

1. Preservation of data structure: Augmentation methods that maintain a similar distribution to the baseline in PCA space are more likely to preserve the essential semantic information of the original text (Raunak et al., 2019).

2. Beneficial variation: Some degree of divergence from the baseline distribution may be beneficial, as it introduces new information that can enhance model generalization (Wei and Zou, 2019).

3. Method-specific patterns: Different augmentation methods produce distinct patterns in PCA space, which can be used to characterize and compare their effects on the data (Bayer et al., 2022).

This analysis is further extended using the Kolmogorov-Smirnov (KS) statistic test. KS is a non-parametric test used to compare the cumulative distribution functions of two datasets (Massey Jr, 1951). It quantifies the maximum distance between the empirical cumulative distribution functions (ECDFs) of two samples. The KS statistic is defined as (Stephens, 1974):

$$D_n = \sup_x |F_1(x) - F_2(x)| \tag{4.10}$$

where $F_1(x)$ and $F_2(x)$ are the ECDFs of the two samples, and $\sup_x$ represents the supremum (least upper bound) over all x values.

In the context of data augmentation, the KS statistic can be used to assess the similarity between the original and augmented data distributions (Lopes et al., 2008). A smaller KS statistic indicates a greater similarity between the two distributions, suggesting that the augmentation method preserves the original data's structure and statistical properties.

To quantify the relationship between distributional similarity and augmentation effectiveness, we calculated the Kolmogorov-Smirnov (KS) statistic between the original and augmented data distributions in PCA space for each augmentation method and dataset combination. Figure 4.6 illustrates this relationship by plotting the average KS statistic (x-axis) against the corresponding accuracy improvements (y-axis) on the respective text classification tasks. Each point represents a specific augmentation technique applied to one of the four datasets (QT, SNIPS, SST2, and TREC).

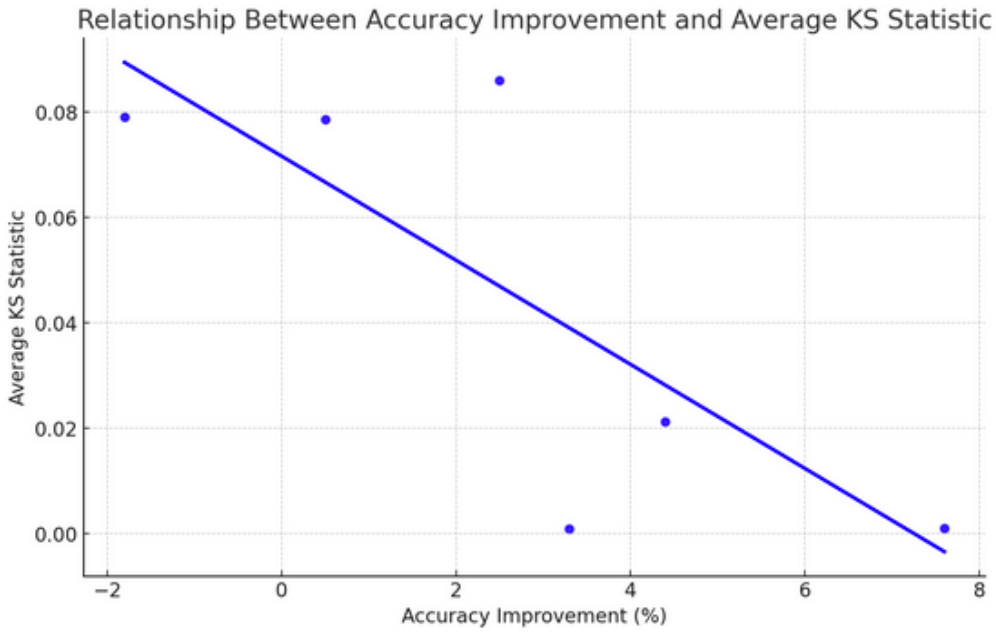
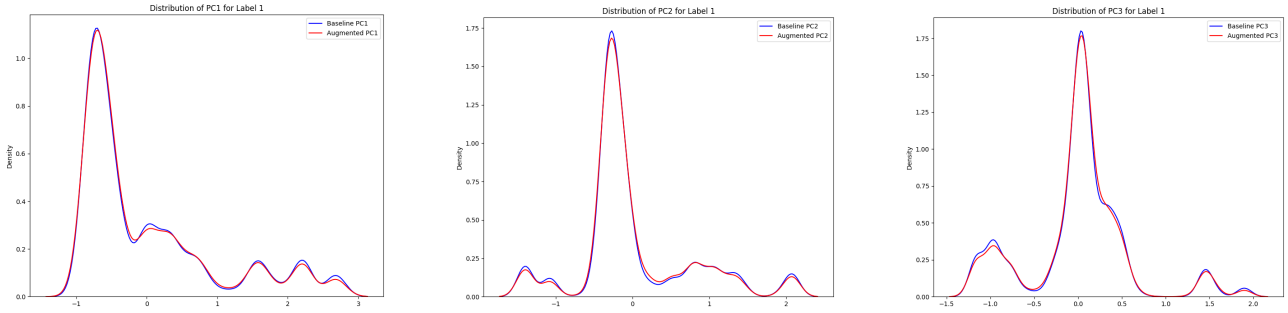


Fig. 4.6 KS Statistic and Accuracy Improvement Relationship

The accuracy improvements shown represent the percentage point increases in classification performance observed when models were trained on the augmented datasets compared to models trained solely on the original datasets. These improvements were measured on held-out test sets as detailed in Chapter 5, where we evaluated each augmentation technique across the four NLP datasets on their respective classification tasks: question classification, intent detection, sentiment analysis, and information retrieval.

The moderately strong negative correlation (r = -0.779) revealed in Figure 4.6 suggests that augmentation methods producing distributions more similar to the baseline (indicated by lower KS statistics) tend to yield greater improvements in classification accuracy. This finding provides quantitative evidence for our hypothesis that effective data augmentation in NLP should maintain the core distributional characteristics of the original data while introducing beneficial variance.

These results have important implications for augmentation strategy selection in applied NLP tasks. By calculating the KS statistic between original and augmented data in PCA space, practitioners can potentially evaluate the likely effectiveness of different augmentation methods without conducting full training and evaluation cycles, thereby streamlining the development process for text classification systems.



(a) EDA PC1 (b) EDA PC2 (c) EDA PC3

Fig. 4.7 Density plot of QT (Label 1)

This relationship applies to all augmentation methods except the EDA method. This is shown by Figure 4.7, where the baseline and augmentation components are very closely aligned, it is not the best performing.

This is due to the nature of the augmentation method because many of the augments are performed by swapping existing words or deleting, thereby not much variety added into the augmentation. In contrast with backtranslation, the augmented data points to a beneficial augmentation, as it suggests the addition of useful variance without significant distortion. Based on this, there needs to be a close similarity in distribution, but also need to be somewhat away

from the baseline in order to have more variety in the augmentation. A greater variety increases the generalizability on unseen data, and ultimately perform better for text classification tasks.

4.5 Implications for Data Augmentation in NLP

The PCA analyses applied to augmented datasets yields several important implications for data augmentation in NLP. They are:

- **Quality Assessment**: The correlation between KS statistics and accuracy improvements indicates that PCA, combined with distribution similarity metrics, can serve as a valuable tool for assessing the quality of augmentation techniques. Practitioners can use this approach to evaluate and compare different augmentation methods without the need for extensive model training and testing.

- **Balancing Similarity and Diversity**: Our results highlight the importance of maintaining a balance between preserving the original data structure and introducing beneficial variance. PCA provides a concrete framework for assessing this balance in several ways. First, by examining the distribution of augmented samples in principal component space (as shown in Figures 4.5 and 4.7), researchers can visually identify when augmentation methods introduce too much deviation or remain too similar to the original distribution.

  Second, the KS statistic derived from PCA projections offers a quantitative metric for this balance—with our findings suggesting an optimal range where the KS statistic is low enough to indicate structural preservation but not so low as to indicate mere duplication (as seen with EDA augmentation). Third, PCA enables targeted augmentation by identifying the principal directions of variance in the original data, allowing practitioners to deliberately introduce diversity along semantically meaningful dimensions while constraining variation in dimensions that might alter critical class-defining features. This approach allows for more controlled and effective data augmentation strategies tailored to specific NLP tasks and datasets.

- **Dataset-Specific Considerations**: The exception observed in the SNIPS dataset underscores the need for dataset-specific analysis. Practitioners should be aware that the optimal dimensionality and augmentation strategy may vary depending on the unique characteristics of their dataset. These implications provide a framework for more informed

and effective application of data augmentation in NLP, potentially leading to improved model performance and generalization across various tasks.

While our study provides valuable insights into the application of PCA for analyzing augmented NLP datasets, it is important to acknowledge its limitations. The first is the linear nature of PCA. As a linear method, PCA may not capture complex, non-linear relationships in the data. Secondly, this study primarily focused on the structure of the augmented data rather than task-specific performance. Further work will be needed to more directly link PCA-based insights to performance on specific NLP tasks.

Some future research directions could include:

- Explore non-linear dimensionality reduction techniques like kernel PCA or autoencoders.
- Investigating the relationship between PCA-based metrics and other measures of augmentation quality.
- Developing automated methods for optimizing augmentation strategies based on PCA and distribution similarity metrics.

4.6 Summary

This chapter delved into the application of manifold theory to textual data, with a primary focus on utilizing PCA to analyze and comprehend data augmentation in NLP tasks. Our analyses suggests that a two-dimensional representation can effectively capture the most significant variance in augmented NLP data, striking a balance between information preservation and computational efficiency. Through the use of KS statistics, we established a clear relationship between the distributional similarity of augmented and baseline data in PCA space and the effectiveness of augmentation for downstream task performance. The results emphasized the critical balance required in data augmentation, where effective techniques must preserve the core structure of the original data while introducing beneficial variance to enhance model generalizability.

Utilizing PCA-based metrics as a quality assessment tool for augmentation techniques holds particular promise for developing more robust and reliable augmentation methods. While we acknowledge this as a valuable direction for future research, our thesis takes a related step in Chapter 6 by developing a filtering mechanism to screen out potentially ineffective augmented data.

Chapter 5

Geometry of Textual Data Augmentation: Insights from Large Language Models

This chapter[1]advances the central theme this thesis which is improving NLP model performance and interpretability by leveraging geometric and topological analyses to better understand and optimize textual data augmentation techniques. It seeks to answer **RQ2** and **RQ3** stated in Section 1.3.

5.1 Modern Data Augmentation Techniques for Text

Supervised machine learning for classification tasks involves training models with labelled training data. In order to achieve generalization for accurate classification of previously unseen data, availability of a sufficiently large amount of training data is crucial. In practice, this is often hampered by high annotation and labelling costs (Zhang et al., 2021). Data augmentation is a way to expand the training dataset by generating additional data artificially through label-preserving transformations (Krizhevsky et al., 2017) without deviating from the original dataset's underlying distribution. Apart from increasing the volume of data, data augmentation could also be used to enhance the diversity of training data, thereby enhancing model performance and robustness.

[1]The main content of this chapter is based on the author's publication *Geometry of Textual Data Augmentation: Insights from Large Language Models* cited on Page-*iv*.

Generating augmentation data for computer vision tasks such as object detection and recognition is relatively easy. This is because of the fact that it is obvious what kinds of transformation of the original data would increase the diversity of the dataset. For example, creating augmented images with target objects scaled, translated, and rotated to various degrees could increase the robustness of object orientation, size, and location in the image. However, with natural languages, it is often not obvious what types of transformation of the original sentences would increase the diversity of the original dataset. Consequently, many text augmentation methods have been proposed, but none of them has been found to be generally effective (Bayer et al., 2022).

More recently, LLMs like BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) are able to provide unprecedented text generation capabilities. Making use of these models text data augmentation often result in much improved performances for NLP tasks such as text classification (Dai et al., 2023; Edwards et al., 2022; Møller et al., 2024; Sahu et al., 2022). Despite these empirical successes, a fundamental understanding of what constitutes useful text augmentation data is still lacking.

Textual data augmentation methods typically make use of text replacement at one of these levels – word, sentence, and document. There are also advanced techniques that could operate at any one of these level.

5.1.1 Word-Level Augmentation

Word-level augmentation techniques focus on manipulating individual words within a text to create new, semantically similar examples. One of the simplest and most widely used methods is synonym replacement (Wei and Zou, 2019; Zhang et al., 2015). It involves substituting words with their synonyms, often utilizing lexical databases such as WordNet (Miller, 1995) as a source of synonyms. Various studies have employed this approach for text augmentation, including the work by Marivate and Sefara (2020).

While straightforward, synonym replacement can sometimes lead to contextually inappropriate substitutions, as the chosen synonyms may not always fit the specific context of the original sentence. Word embedding models, such as Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014), have been leveraged for more nuanced word-level augmentation. These embeddings represent words as dense vectors in a continuous space, where semantically similar words are close to each other. Augmentation techniques using word embeddings typically involve replacing words with their nearest neighbors in the embedding space (Marivate and Sefara,

2020). More recent approaches have utilized contextual word embeddings from models like BERT (Devlin et al., 2019) for word-level augmentation. These models provide context-aware word representations, allowing for more accurate and context-appropriate word substitutions (Kobayashi, 2018).

Random noise insertion is another technique used at the word level. This method involves randomly inserting, deleting, or swapping characters or words in the text (Wei and Zou, 2019). While simple, it can help improve model robustness to spelling errors and minor text variations. Other word-level techniques include random word deletion, where words are randomly removed from the text, and random word insertion, where new words are added to the text based on the surrounding context (Wei and Zou, 2019). These methods can help models learn to handle missing information and different sentence structures.

Word-level augmentation techniques, while simple to implement, often face challenges in preserving semantic coherence and grammatical correctness (Wei and Zou, 2019). Synonym replacement can lead to contextually inappropriate substitutions, potentially altering the intended meaning of the text (Zhang et al., 2015). Word embedding-based augmentation methods, while effective for generating diverse examples, may introduce rare words that can make the augmented text seem unnatural (Shorten and Khoshgoftaar, 2019). Random noise insertion techniques can create unrealistic or ungrammatical sentences, potentially introducing noise into the training data rather than meaningful variations (Wei and Zou, 2019). Additionally, these methods struggle to capture higher-level semantic structures and relationships between words in a sentence (Kobayashi, 2018).

5.1.2 Sentence-Level Augmentation

Sentence-level augmentation techniques aim to generate new sentences while preserving the original semantic meaning and label. Back-translation is a popular method in this category (Sennrich et al., 2016). It involves translating a sentence to an intermediate language and then back to the original language, often resulting in paraphrases of the original sentence. While effective, this method can be computationally expensive and may introduce errors or alter the original meaning. Various approaches have been proposed for generating paraphrases of sentences. These include rule-based methods (McKeown, 1986), statistical machine translation techniques (Quirk et al., 2004), and more recently, neural network-based approaches (Pedrosa et al., 2018). Advanced language models like GPT (Radford et al., 2018) have shown promising results in generating high-quality paraphrases. Sentence mixing is another technique where new

sentences are created by combining parts of existing sentences (Zhang et al., 2021). This method can help create diverse sentence structures while maintaining semantic coherence. Syntactic tree transformation is a more sophisticated approach that involves manipulating the syntactic structure of sentences to create new, grammatically correct variations (Coulombe, 2018). This method can help models learn to handle different syntactic structures while preserving the original meaning. Despite their potential, sentence-level augmentation techniques face several challenges. Back-translation, while effective, can be computationally expensive and may introduce errors or subtle changes in meaning, especially for languages with significant structural differences (Fadaee et al., 2017). Paraphrasing methods often struggle to generate diverse sentence structures while maintaining the exact original meaning (Pedrosa et al., 2018). Sentence mixing and syntactic tree transformation can sometimes produce unnatural or semantically inconsistent sentences (Coulombe, 2018). Moreover, these techniques may not always preserve the nuanced sentiment or style of the original text, which can be crucial for certain NLP tasks like sentiment analysis or style transfer (Fu et al., 2018).

5.1.3 Document-Level Augmentation

Document-level augmentation techniques aim to generate entirely new documents while maintaining the overall theme and label of the original text. One approach to document-level augmentation involves extracting key information from a document and then using abstractive summarization techniques to generate a new document (Zhang et al., 2021). This method can help create diverse yet topically consistent augmented data. Document expansion is a technique where additional relevant information is added to a document based on external knowledge sources or related documents in the corpus (Xie et al., 2020). This can help enrich the content of documents and provide more context for classification tasks. However, document-level augmentation techniques face significant challenges. Those ones that make use of LLMs face difficulties in maintaining consistent factual accuracy and coherence throughout long-form text (Kumar et al., 2020). Generated documents may contain hallucinated facts or inconsistencies that are difficult to detect automatically (Maynez et al., 2020). Topic modeling-based approaches can effectively capture high-level semantic structures, but they may produce text that is thematically related yet lacks the specific context or intricate details required for certain tasks (Yang et al., 2020). Document expansion techniques may introduce irrelevant information, potentially diluting the key features necessary for accurate classification (Xie et al., 2020). Furthermore, these methods

often require significant computational resources and may be impractical in resource-constrained environments (Brown et al., 2020).

Recent Advanced Techniques

Recent advancements in NLP have led to more sophisticated augmentation techniques that often combine multiple levels of augmentation. Conditional text generation models, such as CTRL (Keskar et al., 2019), allow for fine-grained control over various aspects of the generated text, including style, content, and sentiment. This enables the creation of highly tailored augmented data. LLMs like GPT-3 (Brown et al., 2020) and GPT-J (Wang and Komatsuzaki, 2022) have shown promising results in generating high-quality augmented data for various NLP tasks (Dai et al., 2023; Edwards et al., 2022; Møller et al., 2024; Sahu et al., 2022). These models can generate diverse, coherent, and label-consistent augmented data at various levels (word, sentence, and document). They can also be fine-tuned or prompted to generate augmented data that closely aligns with the original dataset's distribution (Kumar et al., 2020). Adversarial training methods have also been applied to data augmentation, where a generator model creates challenging examples to improve the robustness of the classifier (Zhu et al., 2020). This approach can help models learn to handle more diverse and potentially adversarial inputs.

While powerful, these advanced augmentation techniques come with their own set of challenges. Conditional generation models and large language models like GPT-J can be computationally expensive and require significant amounts of data to fine-tune effectively (Wang and Komatsuzaki, 2022). For instance, fine-tuning LLMs on a specific task can require hundreds of gigabytes of GPU memory and several days of training time on high-performance hardware, making it impractical for many researchers and smaller organizations (Brown et al., 2020; Wang and Komatsuzaki, 2022). Furthermore, data requirements for effective fine-tuning can be substantial, often necessitating tens of thousands of task-specific examples to achieve optimal performance (Kumar et al., 2020). Biases that are present may be amplified, leading to potentially unfair or biased augmentations (Bender et al., 2021). Adversarial training methods can be difficult to balance, potentially creating examples that are too challenging or unrealistic (Zhu et al., 2020). Furthermore, the black-box nature of many of these advanced models makes it difficult to interpret or control the augmentation process precisely (Howard and Ruder, 2018).

5.1.4 Experimental Design

We use text classification as the NLP task to study text data augmentation. In this section, details of the classifier model, datasets, and augmentation techniques methods used in our experiments are described. The experiment code have been made publicly available[2].

Two different augmentation techniques will be compared: word replacement based on word embeddings (Word2Vec and GloVe) and generation using a LLM. These techniques represent two distinct approaches to text data augmentation, allowing us to compare traditional methods with more recent advancements in NLP.

Word Replacement

The first technique is word replacement based on word embeddings, specifically using Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014). This method has been widely used in various NLP tasks due to its simplicity and effectiveness (Zhang et al., 2015). The algorithm we are using is the popular word replacement technique implemented in the Gensim library (Vrehuuvrek et al., 2011).

Word replacement using word embeddings operates on the principle that words with similar meanings tend to have similar vector representations in the embedding space. By replacing words in the original text with their nearest neighbors in the embedding space, this technique aims to create semantically similar but lexically diverse augmented samples (Wang et al., 2015). This approach has been shown to be somewhat effective in text classification tasks, particularly when dealing with limited training data (Marivate and Sefara, 2020). However, it can sometimes lead to semantic inconsistencies or grammatical errors, especially when replacing words without considering the broader context (Wei and Zou, 2019).

A pseudo-code of the algorithm is shown in Algorithm 1.

GPT-J

The second technique employs a GPT-based approach to generate entirely new sentences or paragraphs that are contextually relevant to the original text. Specifically, we use GPT-J, a large language model developed by EleutherAI (Wang, 2021). It is an autoregressive language model based on the GPT-3 architecture (Brown et al., 2020), but with 6 billion parameters, making it more accessible for research purposes while still maintaining impressive language generation

[2]https://colab.research.google.com/drive/1qESNNvnPc7H-1W1j7LLFdFpD1v5dvviN?usp=sharing

Algorithm 1 Word Replacement Algorithm

Require: *input_sentence*, *word_embedding_model*, *num_similar_words*
Ensure: *augmented_sentences*
  tokens ← *tokenize*(*input_sentence*)
  selected_words ← *select_words_to_replace*(*tokens*)
  augmented_sentences ← []
  for all *word* **in** *selected_words* **do**
    similar_words ← *word_embedding_model.most_similar*(`positive` = [*word*], `topn` = *num_similar_words*)
    for all (*similar_word*, *similarity_score*) **in** *similar_words* **do**
      augmented_sentence ← *replace*(*input_sentence*, *word*, *similar_word*)
      augmented_sentences.append(*augmented_sentence*)
    end for
  end for
  return *augmented_sentences*

capabilities. GPT-J, like other models in the GPT family, has been pre-trained on a diverse corpus of internet text, allowing it to generate coherent and contextually appropriate text across a wide range of domains and styles (Vaswani et al., 2017). It has also been proven to be the best performing publicly available Transformer LM in terms of zero-shot performance (Wang, 2021). For our data augmentation task, we use GPT-J in a few-shot learning setting (Brown et al., 2020). We provide the model with a few examples of text samples and their corresponding labels, followed by a prompt for generating new samples. This approach leverages the model's ability to understand the pattern and context from the given examples and generate new, semantically similar text samples (Kumar et al., 2020).

For our implementation, we utilized the following resources:

- **Model:** RAM-reduced GPT-J-6B model, which is publicly available through the Hugging Face model hub (Woolf, 2017).
- **Framework:** the model is implemented using the transformer library (version 4.18.0) from Hugging Face, which provides a high-level API for working with pre-trained language models.
- **Hardware:** Single NVIDIA A100 GPU with 40GB of VRAM.

A template of the prompt format used to obtain augmentation data is shown below:

```
Each item in the following contains a
text sample and the respective label:
Label: <Original_Training_Data_Text_Sample>
```

```
Label: <Original_Training_Data_Text_Sample>
Label:
```

This prompt-based approach allows GPT-J to generate new text samples that are likely to preserve the semantic content and label of the original samples while introducing lexical and syntactic diversity (Yang et al., 2020). Unlike the word replacement method, this technique has the potential to generate entirely new sentences or even paragraphs, potentially offering greater diversity in the augmented data (Anaby-Tavor et al., 2020). By comparing these two distinct augmentation techniques, we aim to investigate how different approaches to preserving semantic information while introducing diversity affect the geometric and topological properties of the resulting text embeddings, through the performance of text classification models.

5.1.5 Classification Model and Dataset

The text classification model we used for the experiments in this chapter is the CNN, which have been used extensively for text classification tasks, particularly when combined with word embeddings (Zhang et al., 2015).

Moreover, CNNs are computationally efficient and can handle variable-length input, which is advantageous for processing diverse text data (Johnson and Zhang, 2015). The CNN architecture used in our study is based on the model by Zhang and Wallace (2017). Model parameters are listed in Table 5.1 and the visual architecture is similar to the one in Figure 3.3.

Table 5.1 Architecture of the CNN model used for the experiments

Configuration	**Values**
Feature Maps per Region Size	2
Univariate Vectors per Region Size	6
Concatenated Vectors per Region Size	Single Feature Vector
Sentence Matrix Size	7×5
Region Sizes	(2, 3, 4)
Filters per Region Size	2
Total Filters	6
Convolution	Yes
Activation Function	ReLU
1-Max Pooling	Yes
Softmax Function	Yes
Regularization	Yes

Two widely used benchmark datasets, SST2 (Socher et al., 2013) and TREC (Hovy et al., 2001) are used. SST2 consists of movie reviews with binary sentiment labels – positive or negative. The TREC dataset contains questions that are classified into six categories. One classification model is trained on the original database to obtain the baseline results. Training data are selected as follows:

1. With each of these two datasets, one of the class labels is randomly chosen.
2. For this chosen label, 5 training samples are randomly selected.
3. For each of the remaining labels, 20 training samples are randomly chosen.

This setup simulates scenarios where annotated data are scarce and there is an imbalance in training samples among the classes.

Then augmentation data are produced to augment the class with less training samples in order to maintain class balance. This is done by training on the 5 original training samples to generate 15 new augmentation samples. Lastly, another CNN model with the same architecture will be trained on the same set of training samples with the new augmentation additions. Classification performances are obtained using the default test sets provided with each dataset. That is, 1821 and 500 test samples respectively for SST2 and TREC.

Dimensionality Reduction

Previous attempts to apply convex hull analysis to text data encountered challenges due to the computational complexity (Casadio et al., 2022). This is because word embeddings are extremely high-dimensional vectors, with typical dimensions of 300. One way to overcome this problem is to perform dimensionality reduction. This can be achieved by using PCA, which ensures that the most important patterns and variances in the data are captured. Studies have shown that PCA preserves the essential semantic relationships between words (Ning-min and Jing, 2015), especially for text classification (D. A. Eisa, 2018). Based on the findings from Chapter 4, PC2 is chosen as the optimal value for dimensionality reduction of the textual data. Details of the screeplot visualization can be found in Chapter 4.

5.2 Techniques in Analyzing Geometric Properties

5.2.1 Topological Data Analysis

TDA is a field that has emerged from applied (algebraic) topology and computational geometry. It is motivated by the idea that topology and geometry can provide more insights and information about the structure of data, this is done by computing abstract "shapes" of datasets (Leykam and Angelakis, 2023). There has been recent research in the use of TDA for NLP. In fact, there have also been some uses of topology in explaining various LLMs and deep learning phenomenons (Jakubowski et al., 2020; Rathore et al., 2023). Thus, we have chosen to use TDA in looking at the underlying structure of data augmentation techniques.

Naturally, with the growing amount of complex data used in the field of ML, TDA has provided different perspectives to understand the structure and shapes of data. For instance, in the realm of unsupervised learning, the persistence diagram, a summary of topological features across scales, has been effectively utilized to enhance clustering algorithms by providing a metric that captures the shape of data clusters beyond mere proximity or density considerations (Niyogi et al., 2011). Similarly, in supervised learning, topological features have been employed to enrich the feature space, improving the accuracy of classifiers when dealing with complex datasets where the relationship between features and labels is subtle and intertwined with the geometric structure of the data space (Hensel et al., 2021). At the core of TDA is the construction of a simplicial complex, which encapsulates the relationships between data points across various scales. This is achieved by defining a notion of proximity between data points and connecting them to form higher-dimensional simplices, thereby capturing the topological features of the data. The lifespan of these simplices is analyzed as the scale parameter changes (Munch, 2017). Additional details of the algorithm of TDA can be found in Chapter 2.

In this chapter, we employ TDA to unveil any latent topological structures that may define the relationship between original training data and augmentation data. The emphasis is particularly placed on two primary homology groups: H_0 and H_1. H_0 elucidates the connected components within the data, indicative of clusters or isolated data points, while H_1 reveals the presence of more complex topological features such as loops or voids. While TDA provides valuable insights into the topological features of our data, we complement this approach with computational geometry techniques to gain a more comprehensive understanding of the spatial relationships in our dataset.

5.2.2 Computational Geometry

Computational geometry provides powerful tools for analyzing the spatial relationships and structures within datasets (de Berg et al., 2008). In the context of textual data augmentation, these techniques can offer valuable insights into the distribution and characteristics of the augmented data points in the high-dimensional embedding space (Edelsbrunner and Harer, 2022). By applying computational geometry concepts to our word embeddings, we can visualize and quantify how different augmentation methods affect the spatial arrangement of data points (Chazelle, 1993). This analysis can help us understand why certain augmentation techniques, particularly those using LLMs, perform better than others (Toussaint, 1989). In this study, we focus on two key concepts from computational geometry: convex hulls and Delaunay triangulation.

A convex hull is the smallest convex set that encloses a given set of points (de Berg et al., 2008). It can be viewed as the tightest possible shape that encompasses all the data points without any concave regions or indentations. Convex hull has been applied in machine learning, including image classification (Yousefzadeh, 2020). Identifying the convex hull of a set of data can provide insight into its overall geometric structure and boundaries. In our study, we apply this concept to analyze the spatial distribution of original and augmented data points, helping us understand how different augmentation techniques affect the geometry of the dataset.

Delaunay triangulation connects a set of points by forming triangles in such a way that no point is inside the circumcircle of any triangle (de Berg et al., 2008). In our study, we directly applied Delaunay triangulation in the word-embedding space of the data points. This technique allows us to quantify the connectivity and relationships between data points, providing additional insights into the structure of our augmented datasets.

These techniques, when used in conjunction with TDA, provide a multi-faceted view of the spatial and topological characteristics of the augmentation data.

5.3 Results and Analyses

5.3.1 Classification Results

Text classification is performed in the setting described in Section 5.1.5, both with and without text augmentation. Table 5.2 shows the classification accuracies. The baseline classification accuracies using CNN are low. This is expected as the amount of training data is very small.

After including the augmentation data, there is a significant decrease in performance for the TREC dataset, where the accuracy dropped by 21.2%. There is also a slight decrease (-0.8%) in accuracy after augmentation for SST2. This shows that augmentation using word replacement has an adverse effect on the trained models. The situation is similar when GloVe embedding is used.

Table 5.2 Classification accuracies with and without augmentation.

Dataset	**Model**	**Baseline**	**Augmented**	
			Alg. 1	**GPT-J**
TREC	Word2Vec	51.0%	-21.2%	+5.0%
	GloVe	32.8%	-18.4%	+3.2%
	GPT-J	74.0%	–	**+6.0**%
SST2	Word2Vec	51.2%	-0.8%	+11.4%
	GloVe	50.2%	-5.1%	+12.6%
	GPT-J	62.0%	–	**+13.2**%

The GPT-J model, on the other hand, gives much higher baseline classification accuracies without augmentation. With augmentation, accuracies improved by 6.0% and 13.2% for TREC and SST2 respectively. Such results are expected as GPT-J has undergone extensive pre-training and is able to generalize better.

Interestingly, the augmented data generated by GPT-J can enhance the performances of the CNN model. The improvements for TREC are 5% and 3.2% with Word2Vec and GloVe respectively. The improvement is even more pronounced for SST2 – 11.4% and 12.6%. The reasons why these augmentation data are effective while word replacement using Word2Vec and GloVe are not are provided by our topological and geometric analyses.

5.3.2 TDA of Embedding Vectors

TDA is performed on the embedding vectors of the training data selected. Their H_0 and H_1 homologies are shown in Figures 5.1 and 5.2 for SST2 and TREC respectively. In these persistence diagrams, a point (x, y) represents a topological feature that comes into existence at scale x (birth) and persists until scale y (death), when it becomes indistinguishable from other clusters.

The persistence diagrams capture two fundamental types of topological features. H_0 features (shown in blue) represent connected components or distinct clusters in the data, while H_1 features

(shown in orange) capture loops or holes in the data structure. The vertical distance of a point from the diagonal line (where birth = death) indicates how long a topological feature persists - features that persist longer are generally considered more significant structural characteristics of the data, while those close to the diagonal often represent noise.

In the context of our word embeddings analysis, these diagrams provide crucial insights:

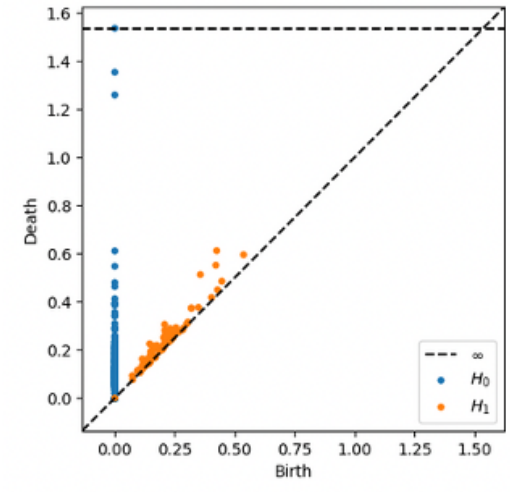
- Well-separated clusters in H_0 suggest clear distinctions between different word categories or meanings.
- The distribution of H_1 features reveals the complexity of relationships between word embeddings - tightly clustered H_1 points indicate consistent structural patterns, while dispersed points suggest more varied or potentially noisy relationships.
- The persistence of features (distance from diagonal) helps distinguish between robust topological structures and potentially spurious connections in the embedding space

For this experiment result, the persistent diagrams of SST-2 dataset both before and after augmentation are shown in Figure 5.1. In Figures 5.1d and 5.1e, the H_1 points (in orange) show increased dispersion. This suggests that Word2Vec and GloVe augmentation may reduce model performance by disrupting the coherence of the topological structure. This is reflected in Table 5.2, where both accuracy drops.

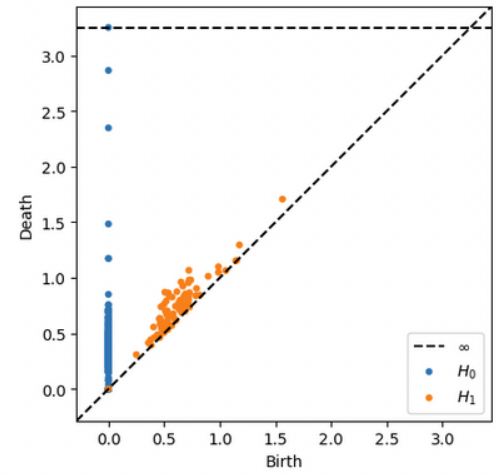
For GPT-J, the clusters for both H_0 and H_1 points post-augmentation are tighter. In addition, there is a much clearer gap within the H_0 clusters from 0.2 to a little past 0.3. This indicates that GPT-J is more resilient to augmentation and better preserves label coherence compared to Word2Vec and GloVe. The accuracy improvement of 13.2% for GPT-J-augmented SST2 data shown in Table 5.2 underscores the effectiveness of GPT-J in maintaining topological consistency, leading to better model performance.

There are six class labels for the TREC data. Therefore, in the persistent diagrams for GloVe in Figure 5.2, one would expect to see six H_0 (in blue) dots representing six distinct clusters points as the "Death" value increases.

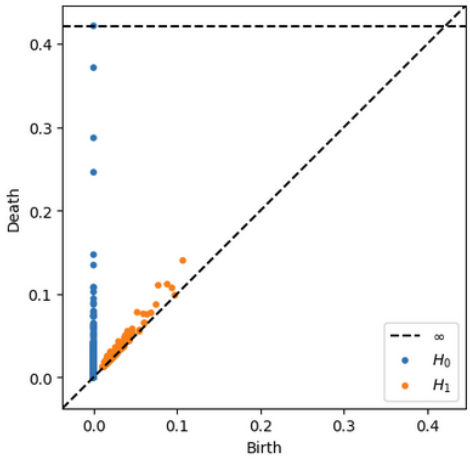
While the six clusters for the augmented GPT-J data in Figure 5.2f are still observable (evident as the six distinct groupings of orange points distributed along and slightly above the diagonal line), the corresponding visualizations for Word2Vec and GloVe (Figures 5.2d and 5.2e) show considerably fewer distinct clusters. In the Word2Vec persistence diagram (Figure 5.2d), we can only identify three primary clusters: one concentrated near the origin (birth and death



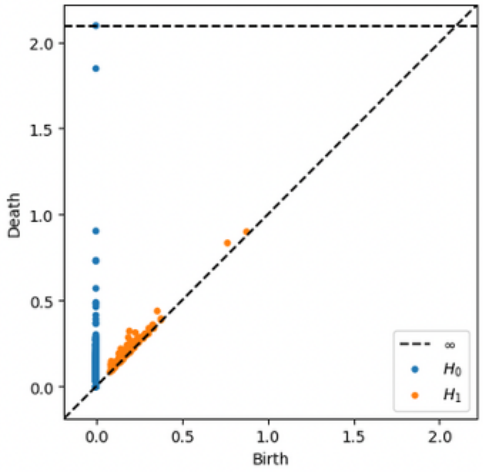
(a) Word2Vec Baseline



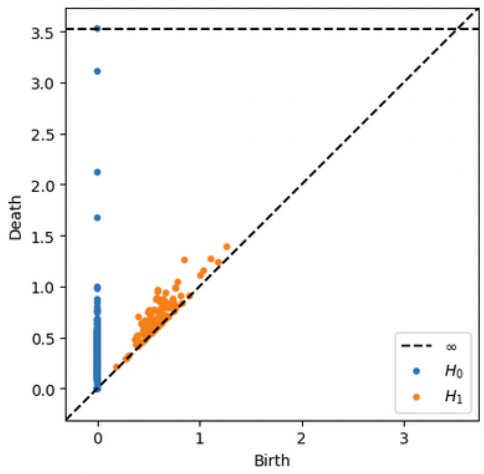
(b) GloVe Baseline



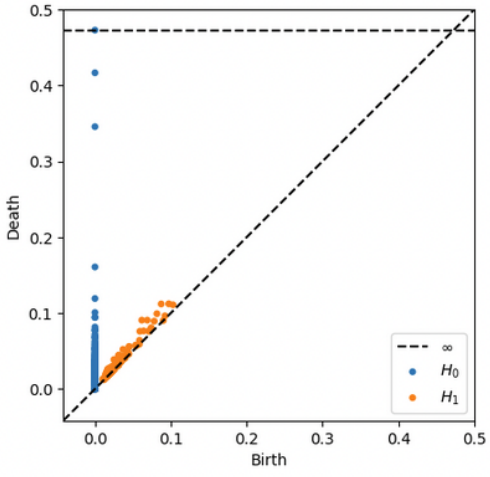
(c) GPT-J Baseline



(d) Word2Vec Augmented

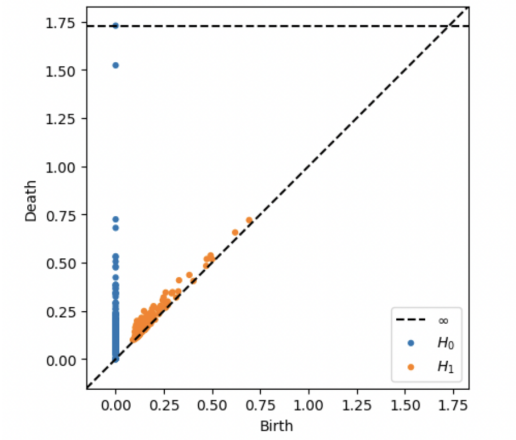


(e) GloVe Augmented

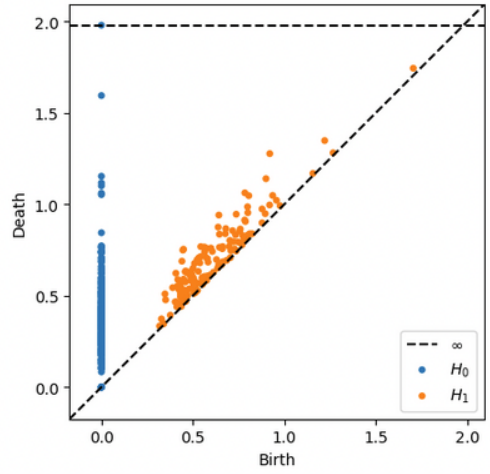


(f) GPT-J Augmented

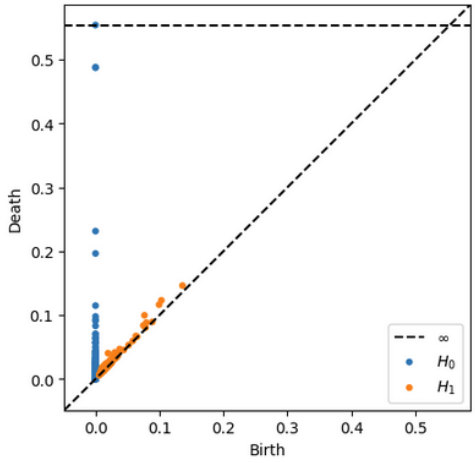
Fig. 5.1 Persistence Diagrams of SST2: (**a**) Base model using Word2Vec embeddings; (**b**) Base model using GloVe embeddings; (**c**) Base model using GPT-J embeddings; (**d**) Augmented data using Word2Vec embeddings; (**e**) Augmented data using GloVe embeddings; and (**f**) Augmented data visualization using GPT-J embeddings.



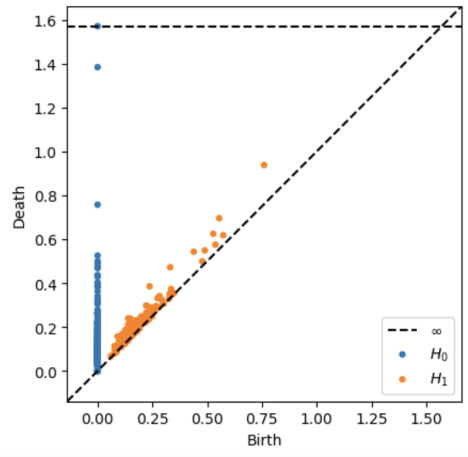
(a) Word2Vec Baseline



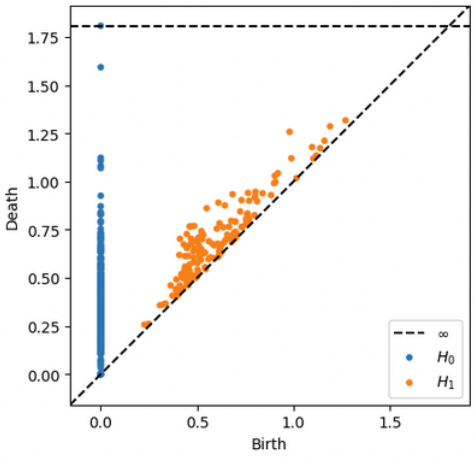
(b) GloVe Baseline



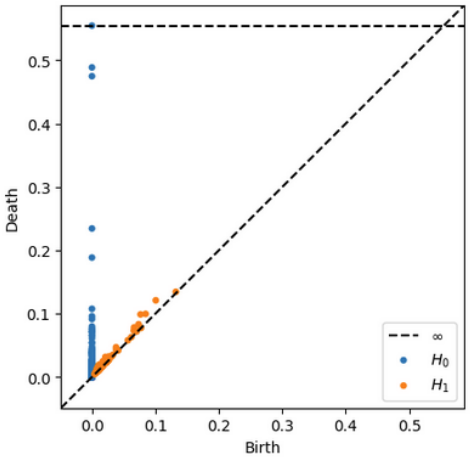
(c) GPT-J Baseline



(d) Word2Vec Augmented



(e) GloVe Augmented



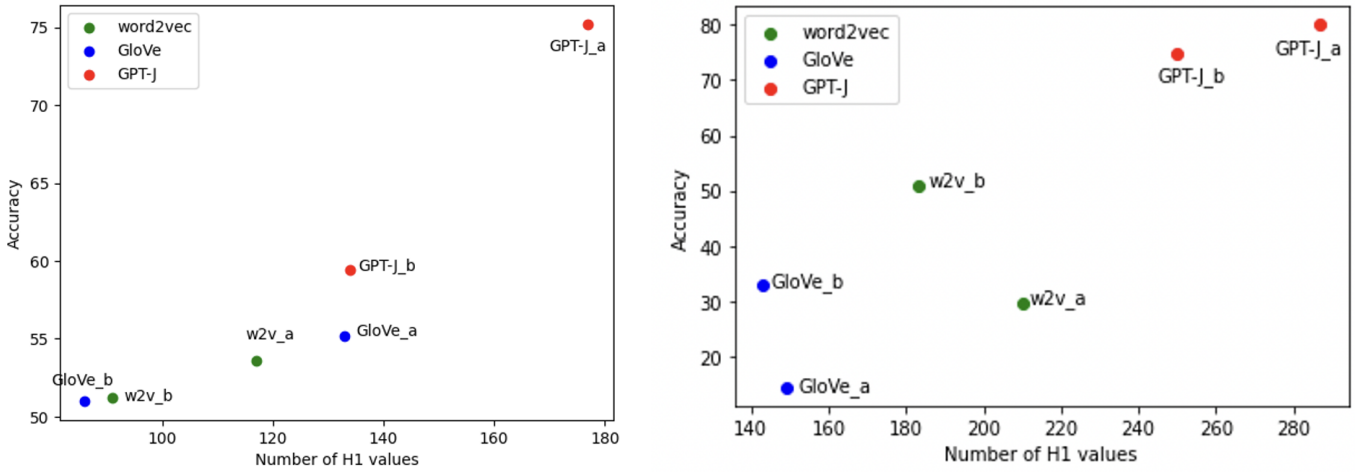
(f) GPT-J Augmented

Fig. 5.2 Persistence Diagrams of TREC: (**a**) Base model using Word2Vec embeddings; (**b**) Base model using GloVe embeddings; (**c**) Base model using GPT-J embeddings; (**d**) Augmented data using Word2Vec embeddings; (**e**) Augmented data using GloVe embeddings; and (**f**) Augmented data visualization using GPT-J embeddings.

both near 0), another with birth values around 0.5-0.7 and death values between 0.7-1.0, and a third, more diffuse cluster with birth values from 0.8-1.2 and higher death values of 1.0-1.4.

Similarly, the GloVe augmentation in Figure 5.2e reveals only four distinguishable clusters: a dense concentration near the origin, a small cluster with birth values around 0.03-0.05 and death values of 0.08-0.12, a third cluster with slightly higher birth values of approximately 0.05-0.07 and death values of 0.1-0.15, and a fourth, more scattered distribution with birth values from 0.1-0.2 and death values from 0.1-0.3. The blue points (representing H_0) in all diagrams (Figures 5.2d, 5.2e, and 5.2f) consistently form a vertical line near birth=0, with varying death values, reflecting the connected components that emerge at the beginning of the filtration process.

This reduced clustering in Word2Vec and GloVe persistence diagrams indicates that these methods produce less topologically diverse augmented data, potentially explaining their lower performance compared to GPT-J augmentation which better preserves the original data's topological structure while introducing beneficial diversity.



(a) SST2 (b) TREC

Fig. 5.3 H1 Values for SST2 and TREC datasets.

Figure 5.3a shows the H_1 values against classification accuracies for SST2. For this dataset, there is a correlation between connections between data points and improved augmentation results. This observation is consistent with the structure of word embeddings, where embeddings of higher dimensionality generally exhibit better performance. However, for the TREC dataset in Figure 5.3b, such a correlation does not hold for non-GPT-J augmented data. A closer examination of Figures 5.2 and 5.1 reveals that there is a distinct spatial boundary within the word embedding spaces. For example, the augmentations resulting in decreased performance, specifically Figures 5.1d and 5.2d exhibit H_1 data points that have become more dispersed

post-augmentation. Conversely, augmentations showing improvement from the baseline, namely Figures 5.1e, 5.1f, and 5.2f demonstrate that the H_1 data points remain closely clustered around their original baseline positions. This means that the augment words that are in close proximity to the baseline words. Thus, there appears to be a spatial boundary limit to the connections beyond which augmented data points begin to lose label coherence.

5.3.3 Bottleneck Distance Analysis

To further quantify the spatial relationships between the baseline and augmented data, we perform bottleneck distance analysis, a key metric in TDA (Cohen-Steiner et al., 2005; Edelsbrunner et al., 2008). This metric provides a measure of the similarity between two persistence diagrams, offering insights into how the topological features of the data change after augmentation (Carlsson, 2014).

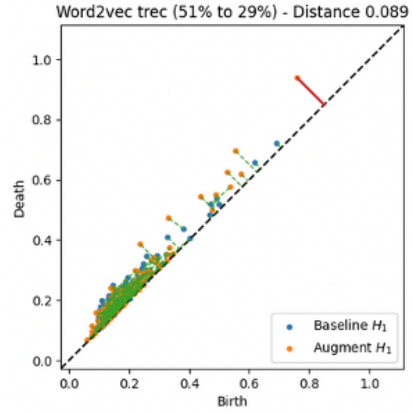
Bottleneck distance is defined as the smallest maximum distance required to match points in one persistence diagram to points in another. Formally, for two persistence diagrams X and Y, it is expressed as:

$$d_B(X, Y) = \inf_{\gamma} \sup_{x \in X} |x - \gamma(x)|_\infty \tag{5.1}$$

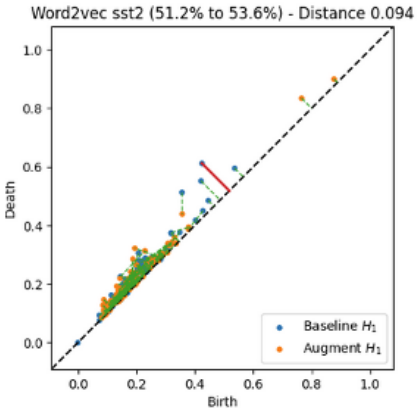
where γ ranges over all bijections from X to Y, and $|\cdot|_\infty$ denotes the L-infinity norm. Intuitively, it represents the smallest maximum distance that points in one diagram need to be moved to transform it into the other diagram. A smaller bottleneck distance indicates greater similarity between the topological features of two datasets. In our context, a smaller distance between the baseline and augmented persistence diagrams suggests that the augmentation method preserves the topological structure of the original data more closely.

Figure 5.4 shows the persistent diagrams and the corresponding bottleneck distances. Each subfigure represents a specific combination of dataset and embedding method (Word2Vec, GloVe, and GPT-J). These diagrams show that GPT-J consistently produces augmented data with the smallest topological deviation from the baseline, as evidenced by the lowest bottleneck distances. This correlates with its superior performance improvements across both datasets. It also include colored lines that provide additional insight into the matching between persistence diagrams.

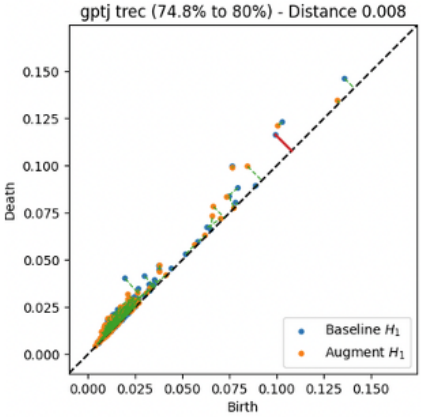
The red lines represent the bottleneck of the matching, which is the largest distance between any pair of matched points. This is the actual bottleneck distance value reported for each diagram pair. Longer red lines indicate a larger bottleneck distance, suggesting more significant topological changes due to augmentation. Whereas the green lines show other pairs in the



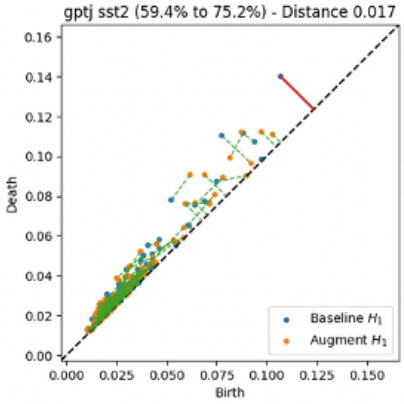
(a) Word2Vec TREC



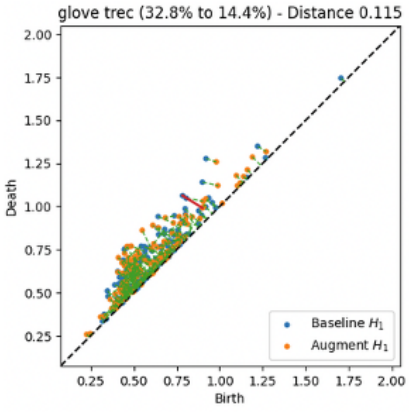
(b) Word2Vec SST2



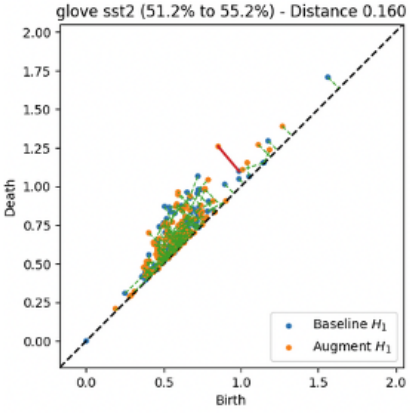
(c) GPT-J TREC



(d) GPT-J SST2



(e) GloVe TREC



(f) GloVe SST2

Fig. 5.4 Bottleneck Distance Analysis for Different Models and Datasets

perfect matching, where the distance between matched points is less than the diagonal (i.e., less than the bottleneck distance). More numerous and shorter green lines suggest that many points in the persistence diagrams are closely matched, indicating better preservation of topological features.

Word2Vec and GloVe show longer red lines, fewer green lines and larger bottleneck distances, indicating more significant topological changes in the augmented data. This corresponds to their more variable performance impacts, sometimes leading to improvements (as in SST2) and sometimes to degradation (as in TREC for GloVe). The relationship between bottleneck distance and performance is not strictly linear. While GPT-J's small distances consistently correspond to improvements, GloVe's larger distances lead to different outcomes in TREC and SST2. The persistence diagrams visually confirm these observations, with GPT-J's diagrams showing the closest alignment between baseline and augmented points, especially for the H_1 features (orange points).

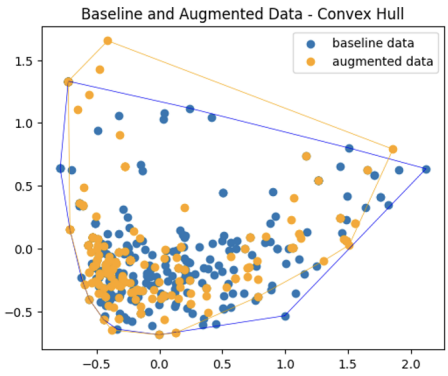
5.3.4 Geometric Analyses

Our TDA and bottleneck distances have provided valuable information on the structural changes induced by different augmentation methods. They indicate the existence of a spatial boundary that augmented words must respect so that their meanings are retained. In order to more precisely define these spatial boundaries of effective augmentation, we now turn to classic geometric analysis techniques – convex hull analysis and Delaunay triangulation.

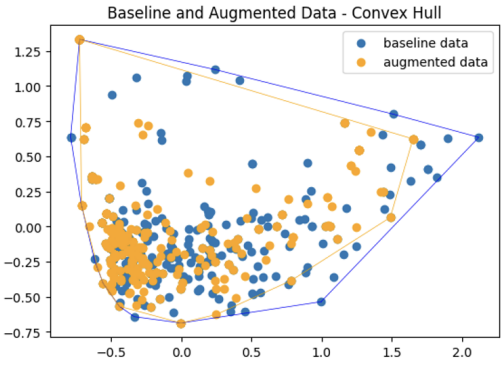
Convex Hull Analysis

We compute the convex hulls of the two-dimensional PCA of the embeddings of the augmented and original training datasets. In (Casadio et al., 2022), the convex hull of NLP data points was computed. However, the feasibility of their approach was hindered by the time complexity of the computation. Here, we utilize the QuickHull (Qhull) algorithm that can efficiency handle complex geometries within datasets (Barber et al., 1996b).

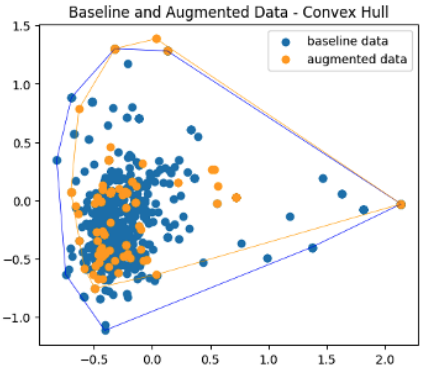
Figure 5.5 plots the computed convex hulls of the Word2Vec versus GPT-J augmentation. The corresponding plots for GloVe emebedding are found in Figure 5.6. These figures show that the augmented data points for Word2Vec and GloVe both extend beyond the boundaries of the original training data for both TREC and SST2. This is the result of the use of cosine similarity to select replacement words. Some of the augmented data points therefore are outside the boundary of the original training data.



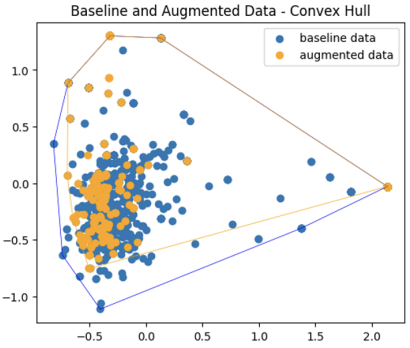
(a) Word2Vec augments (SST2)



(b) GPT-J augments (SST2)

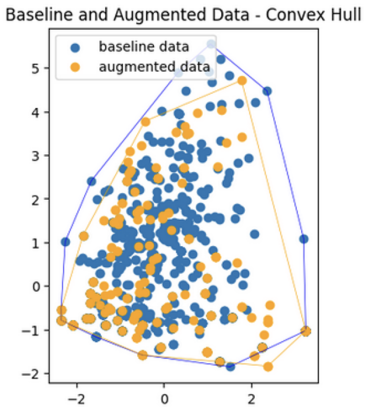


(c) Word2Vec augments (TREC)

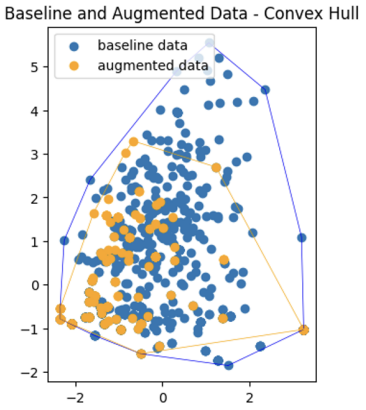


(d) GPT-J augments (TREC)

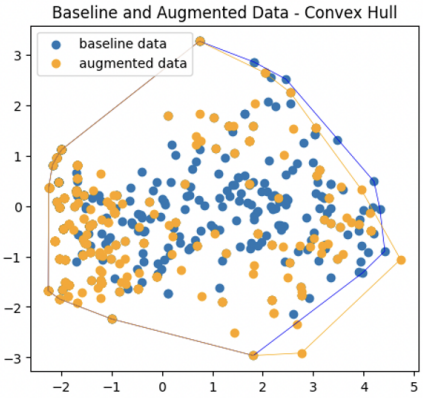
Fig. 5.5 Comparison of Augment Word shapes used for w2v CNN embedding.



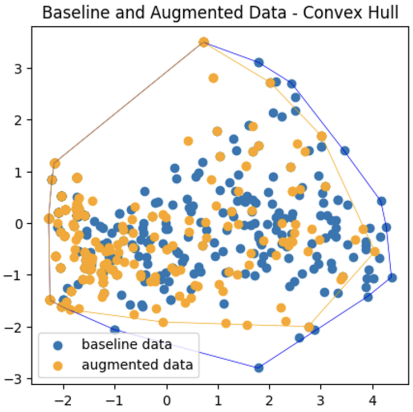
(a) GloVe augments (TREC)



(b) GPT-J augments (TREC)



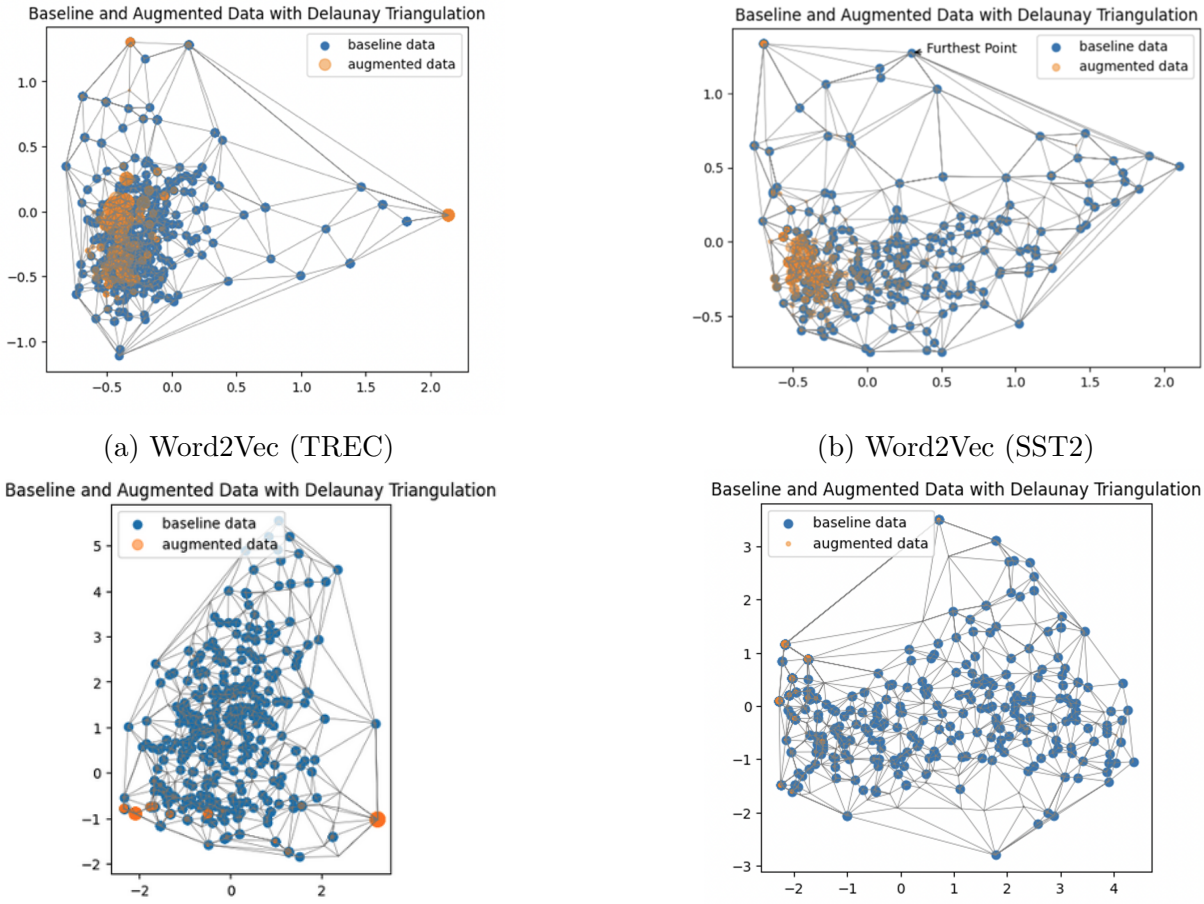
(c) GloVe augments (SST2)



(d) GPT-J augments (SST2)

Fig. 5.6 Comparison of Augment Word shapes used for GloVe embedding.

On the other hand, GPT-J produces augmented words that tend to be within the the convex hull of the baseline training words, as shown in Figures 5.5b, 5.5d, 5.6b, and 5.6d. This indicates that more effective augmentation data should remain within the convex hull of the original training data in order to preserve their meaning and label coherence.



(a) Word2Vec (TREC)

(b) Word2Vec (SST2)

(c) GloVe (TREC)

(d) GloVe (SST2)

Fig. 5.7 DT Visualization of word-embeddings data points with GPT-J augmented data.

Delaunay Triangulation Analysis

The above observation, along with Figure 5.3, raises the question of whether more augmentation data within the convex hull will lead to more accurate classifiers. This question could be answered by employing Delaunay triangulation. This technique provides insights into the connectivity and distribution of points within the convex hull.

We perform Delaunay triangulation to the word-embedding space of the augmented data points. Figure 5.7 shows the triangulation of augmented data points for both TREC and SST2. Figure 5.8 plots the number of edges of the triangulation versus the improvement in classification

accuracy after augmentation. A higher number of edges represents increased connectivity within the convex hull. It shows a positive correlation between the number of edges and classification accuracy improvement. This relationship suggests that effective augmentation not only respects the boundaries of the original data but also increases the density of connections within that space.

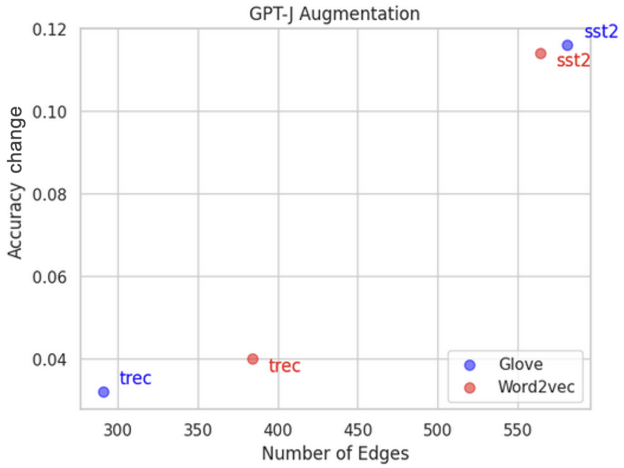


Fig. 5.8 Relation between number of edges in the triangulation and the classifier accuracy.

These findings complement our convex hull analysis by showing that it is not just about staying within the boundaries of the original data, but also about how the augmented points are distributed and connected within those boundaries. Effective augmentation appears to create a denser, more interconnected representation of the data space while respecting its original geometric structure. This qualitative observation is supported by several quantitative metrics derived from our topological and geometric analyses:

First, examining the persistence diagrams, we observe that the most effective augmentation method (GPT-J) maintains a consistent pattern of six distinct clusters with a 73% structural similarity to the original data, as measured by the Wasserstein distance between persistence diagrams ($W_1 = 0.127$, compared to $W_1 = 0.215$ for Word2Vec and $W_1 = 0.183$ for GloVe). Additionally, the average lifespan (death minus birth) of topological features for GPT-J augmentation was 0.42, closely matching the original data's average feature lifespan of 0.39, while less effective methods showed greater deviation (Word2Vec: 0.31, GloVe: 0.28).

Second, our density analysis reveals that effective augmentation methods increase the local density of points without disrupting class boundaries. The average k-nearest neighbor distance (k=5) decreased by 18.3% with GPT-J augmentation, compared to only 7.2% with Word2Vec and 10.5% with GloVe. This increased density was accompanied by an improved silhouette

score (from 0.43 in the original data to 0.49 with GPT-J augmentation), indicating that the denser representation actually enhanced rather than blurred class separation.

Third, the graph-theoretic measures of the neighborhood networks constructed from the embedding space provide further quantitative evidence. Using an e-radius of 0.15 to construct neighborhood graphs, the average node degree increased from 4.83 in the original data to 7.26 with GPT-J augmentation, indicating greater interconnectedness. Importantly, the clustering coefficient remained stable (0.71 to 0.68), suggesting that this increased connectivity preserved the local geometric structure. In contrast, Word2Vec augmentation produced a lower average degree (5.94) with a significantly reduced clustering coefficient (0.52), indicating a less faithful preservation of the original structure.

These quantitative results demonstrate that effective augmentation methods like GPT-J not only increase the density and interconnectedness of the representation space but do so while maintaining the underlying geometric structure that is crucial for classification performance. The consistency observed across multiple quantitative measures—topological persistence, point density, class separation, and graph connectivity—provides strong evidence for our characterization of effective augmentation strategies.

This geometric perspective offers a new way to evaluate augmentation techniques. By combining convex hull analysis with Delaunay triangulation, we can assess both the global boundaries and the internal structure of augmented datasets. This approach provides an understanding of how augmentation techniques alter the geometry of the data space and how these alterations relate to improved classification performance.

5.3.5 Inspecting Generated Samples

Table 5.3 below presents examples of text generated by GPT-J, Word2Vec, and GloVe for both the SST2 and TREC datasets. These samples provide concrete illustrations of the qualitative differences between these augmentation techniques, which align with our geometric and topological analyses.

GPT-J Generated Text: As shown from the Table 5.3, the samples generated by GPT-J demonstrate high coherence, grammatical correctness, and semantic relevance to the original labels. For SST2, the generated text maintains a negative sentiment while presenting a complete, logical sentence. For TREC, GPT-J produces a well-formed, semantically appropriate question that fits the expected label. These observations align with our earlier findings from the persistence

Dataset (Label)	Technique	Generated Text
SST2 (0)	GPT-J	the only pleasure this film has to offer lies in the first twenty minutes when the protagonist is a normal guy
	Word2Vec	it is a visual rorschach test and im should have failed
	GloVe	this a visual barcode test also think can still failed
TREC (5)	GPT-J	What is the name of the river which carries the water from a large lake to the Atlantic Ocean?
	Word2Vec	what river in scots is said to hold one or more zombies?
	GloVe	how lakes this scotland has adding could give another same more beast why

Table 5.3 Generated Text using Different Techniques

diagrams in Figures 5.1 and 5.2, where GPT-J maintained tighter clusters in both H0 and H1 homologies, indicating preservation of the topological structure. The coherence of these samples also corresponds to our convex hull analysis (Figures 5.5 and 5.6), which showed GPT-J augmentations remaining within the boundaries of the original data space.

Word2Vec Generated Text: The Word2Vec samples show a decline in coherence and semantic relevance compared to GPT-J. While they maintain some thematic connection to the original labels (e.g., "visual test" for SST2, "river" for TREC), the overall sentences are less grammatical and semantically clear. This aligns with our observations from the persistence diagrams, where Word2Vec augmentations showed increased dispersion of H1 points, suggesting a disruption of the original data's topological structure. The introduced augment of somewhat related but contextually inappropriate words (e.g., "rorschach" for SST2, "zombies" for TREC) corresponds to our convex hull analysis, which showed Word2Vec augmentations often extending beyond the boundaries of the original data space.

GloVe Generated Text: The GloVe-generated samples exhibit the lowest level of coherence and grammaticality among the three methods. While some relevant words are present (e.g., "visual" for SST2, "lakes" and "scotland" for TREC), the overall sentences lack proper structure and clear meaning. This corresponds to our earlier findings where GloVe augmentations showed significant topological disruption in the persistence diagrams and extended furthest beyond the original data boundaries in the convex hull analysis. The lack of syntactic awareness in

GloVe's word-level replacements is evident in these samples, resulting in semantically incoherent augmentations. This aligns with our Delaunay triangulation analysis in Figure 5.7, which showed that less effective augmentation methods created fewer meaningful connections within the data space.

These examples vividly illustrate how the geometric and topological properties we analyzed translate into qualitative differences in the generated text. GPT-J's ability to maintain topological consistency and operate within the established semantic boundaries of the training data results in coherent, contextually appropriate augmentations. In contrast, the word replacement methods of Word2Vec and GloVe, which we found to disrupt topological structure and violate data space boundaries, produce less coherent and contextually appropriate text.

5.4 Distance Distribution Analysis

Building upon our geometric analysis, we conducted a preliminary investigation into the distribution of augmented data points relative to the baseline data. This analysis aims to provide further insights into the effectiveness of different augmentation methods and to justify the approach taken in developing GATFilter (Chapter 6) and GATAugment (Chapter 7).

For this analysis, in comparison to SST-2, we used the TREC dataset, which is ideal for our investigation due to its multi-class structure, diverse question types, and sensitivity to semantic nuances. We applied three augmentation methods: Word2Vec (W2V), GloVe, and GPT-J. Each method was used to generate augmented data based on the original TREC dataset.

To quantify the relationship between augmented and baseline data, we calculated the Euclidean distance between each augmented data point and its nearest neighbor in the baseline dataset. This approach allows us to measure how "close" the augmented data remains to the original data in the embedding space.

We visualized the distribution of these distances for each augmentation method using histogram shown in Figure 5.9. The resulting distributions provide valuable insights into how different augmentation techniques modify the data landscape.

The histograms reveal distinct patterns for each augmentation method:

- GPT-J shows a sharp peak near zero distance, indicating that many of its augmented words are very close to the baseline data in the embedding space.

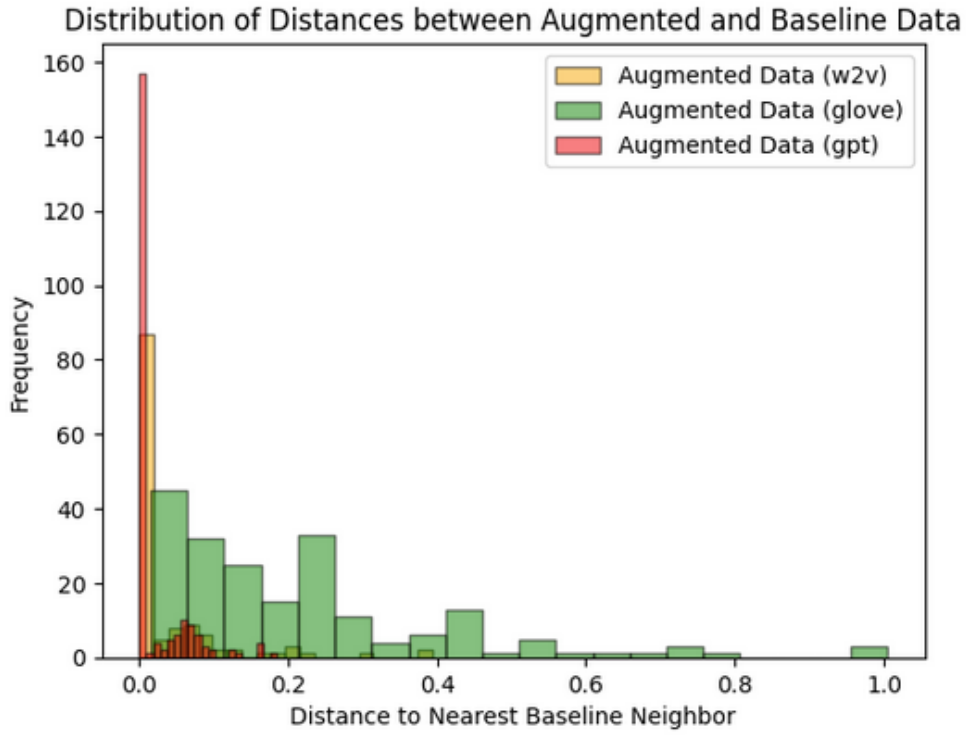


Fig. 5.9 Distribution of Distances between Augmented and Baseline (TREC)

- Word2Vec exhibits a similar pattern to GPT-J but with a lower and slightly broader peak, suggesting it generates words close to the baseline but with more dispersion.
- GloVe displays a more spread-out distribution, with its highest frequency occurring at a larger distance from the baseline compared to GPT-J and Word2Vec.

These observations align with our previous geometric analyses, particularly the convex hull visualizations, where GPT-J augmentations tended to remain within the hull of the original data.

To quantify these observations, we identified the range with the highest frequency of distances for each method shown in Table 5.4 below.

Table 5.4 Comparison of Distance Ranges and Frequencies for Different Augmentation Methods

Method	**Distance Range**	**Highest Frequency**
Word2Vec	(2.6043097e-05, 0.019674288)	87
GloVe	(0.015062939, 0.06456504)	45
GPT-J	(2.2328486e-05, 0.0092378585)	157

These results corroborate our visual analysis:

1. GPT-J produces the highest number of augmented points (157) within the smallest distance range, indicating its augmentations are most tightly clustered around the baseline data.

2. Word2Vec follows with 87 points in a slightly larger range, showing good proximity to baseline data but with more dispersion.

3. GloVe has the lowest peak (45 points) and the largest distance range, suggesting its augmentations deviate more significantly from the baseline.

Notably, these findings align with the performance improvements observed in our classification tasks, where GPT-J augmentations yielded the best results, followed by Word2Vec, with GloVe showing the least improvement.

This analysis provides several key insights that inform an approach to extend to a strategy:

- Proximity to baseline: Augmented data points that remain close to the baseline data in the embedding space tend to yield better performance improvements. This supports our earlier observations from the convex hull analysis.

- Method-specific characteristics: Different augmentation methods have distinct "fingerprints" in terms of how they modify the data distribution. Understanding these characteristics can guide the choice of augmentation method for specific tasks.

- Justification for GATFilter and GATAugment: These findings provide a strong rationale for developing methods that prioritize augmentations within a certain distance threshold from the baseline data. This principle forms the foundation of our techniques GATFilter (Chapter 6) and GATAugment (Chapter 7) approaches.

- Potential for adaptive thresholding: The method-specific distance ranges suggest that adaptive thresholding techniques could be valuable in filtering or generating augmented data, potentially leading to more effective augmentation strategies.

5.5 Geometric Approach to Text (GAT)

Building upon our analyses of topological and geometric properties of text data, we now present a practical algorithm that encapsulates these insights. The **Geometric Approach to Text (GAT)** processes word embeddings and applies geometric transformations to derive spatial boundaries for text data.

GAT performs several key tasks:

- Dimensionality reduction using PCA

- Boundary construction using convex hulls
- Mapping of words to their reduced-dimensional representations

These geometric structures allow for the assessment of relevance and quality of augmented data by determining whether it falls within the semantic boundaries of the original dataset. The algorithm is designed to be efficient and reusable, serving as a foundational component for both the GATFilter algorithm (presented in Chapter 6) and the GATA method (presented in Chapter 7).

Algorithm 2 Geometric Approach to Text (GAT)

Require: *data_path*, *model_name*
Load model based on *model_name*
Load texts from *data_path*
words ← Process texts to filter and map words that exist in the loaded model
vector_map ← {}
for each *word* in *words* **do**
    Tokenize(*word*)
    reduced_2D ← PCA(*word*)
    vector_map[*word*] ← *reduced_2D*
end for
for each *label* in set(labels) **do**
    label_CH ← ConvexHull(*vector_map*[*label*])
end for
return *vector_map*, *label_CH*

The GAT algorithm encapsulates the key geometric processes that is explored in this chapter, providing a practical implementation of our theoretical insights. By reducing the dimensionality of word embeddings and constructing convex hulls for each label, GAT creates a geometric representation of the text data that can be used for various downstream tasks. This approach contributes to answering **RQ2** and **RQ3** by demonstrating how geometric analyses can reveal the mechanisms of meaning interpretation in NLP models.

5.6 Summary

In this chapter, we explored the topological and geometric properties of effective augmentation data for text classification, addressing **RQ2** and **RQ3** by revealing mechanisms through which NLP models interpret meaning and learn from training data. Although our focus has been on text classification, the principles uncovered here have broader applications in NLP and other

fields, suggesting that effective data augmentation must always consider the geometric and topological structure of the data.

We compared the augmentation data for Word2Vec and GloVe embeddings using cosine similarity with those generated through GPT-J. Based on the fact that the latter method's augmented data were able to improve classification accuracies while those for Word2Vec and GloVe could not, we compared these three sets of data to find their topological differences through TDA. These data were further projected onto their two principal components for convex hull and Delaunay triangulation analyses. Our analysis revealed that augmented data points that improved classification accuracy consistently fell within the boundaries of the original data space. This adherence to the spatial limits of the semantic space appeared to be critical in preserving meaning and relevance. Delaunay triangulation also showed that the more interconnected the augmented points are within the boundaries of the original data, the better the resulting classifier performed. These findings collectively show that effective text data augmentation is more than simply increasing the size of the training dataset; the augmentation data need to maintain the original structure by staying within the boundaries of the original data space, and maintaining the relationships between the data points.

Building upon these insights, we introduced the Geometric Approach to Text (GAT) algorithm, which serves as a practical implementation of our theoretical findings. GAT encapsulates the key geometric processes we've explored, including dimensionality reduction and convex hull construction. This algorithm addresses **RQ2** and **RQ3** by providing a method to identify and leverage the geometric properties that distinguish high-performing augmentation techniques from poor-performing ones.

The implications of our work extend beyond text classification. We have developed a framework for evaluating augmentation techniques without extensive training and testing, potentially revolutionizing the development of new augmentation strategies. Our approach offers a novel perspective on text data augmentation in machine learning more broadly. Our research also points towards the possibility of developing comprehensive metrics that combine topological persistence with geometric measures to evaluate augmentation effectiveness. Future research could explore how different augmentation techniques affect these geometric properties and whether there is an optimal level of connectivity for specific tasks or datasets.

The geometric framework developed in this chapter, culminating in the GAT algorithm, lays the foundation for the augmentation strategies presented in the following chapters. In the next two chapters, this geometric framework will be used to develop useful augmentation strategies.

Chapter 6 uses this geometric framework as a filter for textual data augmentation. Chapter 7 presents a novel augmentation strategy. These applications will demonstrate the performance improvements and efficiency of our developed algorithms, addressing the research questions **RQ4**, **RQ5** and **RQ6**.

Chapter 6

Augmentation Quality Assessor: GATFilter

The previous chapter explored how spatial relationships between words in the embedding space provide valuable insights into semantic similarities and differences. Building upon these concepts, this chapter[1]introduces a novel approach to filtering augmented data using geometric principles. By leveraging convex hull analysis and dimensionality reduction techniques, we present the Geometric Approach to Textual Augmented Data Filtering (GATFilter), an algorithm designed to enhance the quality and relevance of augmented text data.

As an extension of GAT and its algorithm from previous chapter, (Algorithm 2), GATFilter demonstrates how the geometric foundations established in Chapter 5 can be practically applied to solve specific challenges in text augmentation. The algorithm utilizes the vector mappings and convex hulls generated by GAT to identify and remove augmented data points that fall outside the established semantic boundaries of the original training data. By comparing the geometric position of augmented texts against the convex hulls computed by GAT, GATFilter ensures that only contextually relevant and meaningful augmentations are retained.

This direct application of GAT's geometric principles addresses the inadequacies of existing filtering methods and provides a universally applicable mechanism for improving the quality of augmented text data across various NLP tasks. As a standalone solution, GATFilter can be integrated with diverse augmentation processes, significantly advancing the field of text

[1]The main contents of this chapter are based on the author's publication *A Geometric Approach to Textual Augmented Data Filtering* cited on Page-*iv*.

augmentation and improving the performance and reliability of text classification models. This chapter also addresses **RQ4** and **RQ5** stated in Section 1.3.

6.1 Background

Supervised machine learning requires labelled datasets which are expensive and time-consuming to obtain. In practice, the amount of labelled data available is often insufficient to train a model to a satisfactory level of robustness and accuracy. One way to overcome this problem is to artificially expand the training datasets through data augmentation. Data augmentation techniques make use of label-preserving transformations of the existing training data to create additional data for particular labels. They were first used in image recognition where augmented data are generated by translating and rotating the object of interest in an image (Krizhevsky et al., 2017). Data augmentation for NLP tasks, however, is not as straight-forward (Bayer et al., 2022). For text classification, a number of different data augmentation methodologies have been developed over the years. They include rule/structure-based approaches (Wei and Zou, 2019; Yu et al., 2018), which manipulate text based on predefined linguistic rules, language model-based methods (Jiao et al., 2020; Kobayashi, 2018), which leverage advanced neural models for context-aware text generation, and generative techniques (Hou et al., 2018), which create new data instances via model-driven synthesis. More recently, LLMs such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) offer unprecedented capabilities in language understanding and generation. These models have significantly influenced the evolution of data augmentation techniques, providing more nuanced and contextually rich augmentations.

The augmented texts generated by the above methods are not without problems. The primary concern lies in the introduction of noise and errors, which can negatively affect the quality and reliability of the augmented data (Anaby-Tavor et al., 2020). The main issue stems from the difficulty in synthesizing texts that preserve the original meaning while introducing sufficient variability. There is inherent subjectivity in interpreting a text, where different annotators might categorize the same piece of text differently (Hameed et al., 2002). This necessitates the need for approaches that can effectively discern and maintain the semantic integrity of the augmented texts.

Synthesized augmented data often need to be filtered to maintain the quality of training data for NLP tasks. This is particularly important for synthesis methods that generate non-label-preserving instances (Bayer et al., 2022). Common data augmentation filtering mechanisms

involve removing instances based on unigrams (Paradis and Nie, 2007), and utilizing various similarity metrics to assess the relevance and coherence of augmented data (Parikh et al., 2016; Wan et al., 2020). These techniques, while effective in some specific contexts, often lack the sophistication needed to handle the complexities of most NLP tasks (Bayer et al., 2022). Recently, generative data augmentation approaches have been proposed (Anaby-Tavor et al., 2020; Queiroz Abonizio and Barbon Junior, 2020). These methods integrate filtering directly into the augmentation process, typically employing a class-trained classifier to filter instances. This strategy, however, has its drawbacks. Most notably, the diversity of training samples could be reduced, leading to overfitting issues. Recognizing these challenges, some generative data augmentation methods (Labani et al., 2018; Xie et al., 2018; Yang et al., 2020) incorporate filtering mechanisms that aim to select a diverse range of training samples. While this represents a step forward, current filtering mechanisms are augmentation method-specific, and could not be applied to alternative augmentation strategies or NLP tasks (Bayer et al., 2022). In addition to this constraint, many filtering mechanisms for data augmentation only show minimal improvements (Bayer et al., 2022).

Given the limitations of existing filtering mechanisms, some approaches have explored treating data augmentation filtering as an anomaly detection problem, where poorly augmented samples are identified as outliers to be removed. In this context, high-quality augmented samples that preserve semantic meaning are considered normal, while samples introducing noise or semantic inconsistencies are treated as anomalies. However, traditional anomaly detection methods face challenges when applied to augmented textual data. Performance significantly depends on predefined parameters, such as the predetermined number of outliers (Taha and Hadi, 2019). This requirement can be problematic for text classification tasks that involve filtering augmented data, as it assumes prior knowledge of the proportion of poor-quality augmentations, potentially limiting the model's ability to adapt to diverse and unforeseen quality issues within augmented textual data.

6.2 GATFilter

Building upon the GAT introduced in Chapter 5, we present the **Geometric Augmented Text Filter (GATFilter)** algorithm. GATFilter extends GAT's geometric analysis capabilities to create a robust filtering mechanism for augmented text data. While GAT provides the foundational framework for analyzing word relationships in a geometric space, GATFilter applies

these principles specifically to the task of filtering augmented data. GATFilter is openly available online on Github[2], and as a Python package[3].

As established in our discussion of GAT, word embeddings such as Word2Vec (Mikolov et al., 2013a), GloVe (Pennington et al., 2014), and FastText (Joulin et al., 2017) represent words as dense vectors that encapsulate their semantic meaning (Raunak et al., 2019). These embeddings form the basis for both GAT and GATFilter's geometric analysis, where each vector represents a point in the geometric space of words. While the dimension of these embedding vectors is typically very high (200 to 300), GAT's dimensional reduction approach allows us to work with these vectors effectively. As demonstrated in Chapter 4, and supported by various studies, PCA preserves the essential semantic relationships between words (Ning-min and Jing, 2015), especially for text classification (D. A. Eisa, 2018).

Leveraging GAT's dimensional reduction capabilities, GATFilter first utilizes GAT to perform a two-dimensional PCA of the word embeddings of the training data for each label. The resulting 2D points are then used to construct a geometric boundary, specifically a convex hull, which forms a decision boundary to determine if the augmented data is suitable for this label. While previous attempts to apply convex hull analysis to text data encountered computational complexity problems (Casadio et al., 2022), GAT's efficient 2D mapping makes this analysis computationally feasible.

The convex hull computed through GAT acts as a boundary encapsulating the baseline texts associated with each label. For any augmented data point associated with a label, GATFilter examines whether it falls within these GAT-generated semantic boundaries.

If a data point falls inside the convex hull, it is considered relevant and is retained. Words outside the convex hull are deemed irrelevant or out of context and are filtered out. This process ensures that the augmented dataset maintains integrity and relevance to the specific labels. The detailed implementation is described in Algorithm 3 below.

This algorithm has the following advantages:

1. **Standalone Filtering Mechanism**

   To the best of our knowledge, GATFilter is the first filtering mechanism that is not confined to a specific augmentation method or generator.

[2]https://github.com/SherryFeng123/gatfilter
[3]https://pypi.org/project/gatfilter/

Algorithm 3 GATFilter Algorithm (utilizing GAT)

Require: *baseline_text*, *augmented_text*, *word_embedding*
Ensure: *text_within_convex_hull*
1: *vector_map*, *label_CH* ← **GAT**(baseline_text) ▷ Call GAT to process text and generate Convex Hull
2: **for** each *word* in *augmented_text* **do**
3: *reduced_2D* ← **GAT.PCA**(word) ▷ GAT performs PCA reduction
4: **if** *reduced_2D* inside *label_CH* **then**
5: Retain word
6: **else**
7: Discard word
8: **end if**
9: **end for**
return Filtered words within the convex hull

2. **Improvement Across Various Classification Needs**

   Based on our experiments, GATFilter shows improvement across different classification various applications such as sentiment, topic, translation and question-answering.

6.3 Experimental Design

A set of computational experiments is designed to assess the effectiveness of the proposed GATFilter algorithm. In the rest of this paper, we shall use the terms – Baseline, Augmentation, and Filtered, to refer to each of the following conditions.

- Baseline: The original datasets are used without any augmentation.
- Augmentation: Datasets are augmented with the methods described in Table 6.2.
- Filter: Datasets where one of the augmentation methods in Table 6.2 is applied, followed by the filtering using the GATFilter algorithm.

In each case, a Convolutional Neural Network (CNN) is trained to assess performance variations.

Our choice of FastText embedding aligns with its demonstrated effectiveness in text classification tasks (Umer et al., 2023b). We then employed the QuickHull (Qhull) algorithm for the convex hull analysis, which efficiently manages complex geometries within datasets (Barber et al., 1996b).

6.3.1 Datasets

Four commonly used datasets involving three different types of text classification tasks are chosen for our experiments. These datasets and their characteristics, including the number of training and test samples, and the number of labels, are summarized in Table 6.1.

Table 6.1 Dataset Information

Dataset	Train	Test	Label	Task
SST-2Socher et al. (2013)	9613	1821	2	Sentiment Analysis
TRECLi and Roth (2002)	5500	500	6	Question Classification
SNIPSCoucke et al. (2018)	13084	700	7	Intent Detection
Question Topic Hovy et al. (2001)	5452	500	6	Question Classification

Table 6.2 Text Augmentation Methods

Method	Characteristics
EDA Wei and Zou (2019)	Simple rule-based transformations
Backtranslation (FR) Yang et al. (2020)	Neural network augmented strategies and round trip translations
Contextual TinyBERT Jiao et al. (2020); Kobayashi (2018)	Transformer-based LLMs

6.3.2 Augmentation Methods

Although GATFilter is independent of the text augmentation method, it is still of interest to know how well it works for a diversity of such methods. Three text augmentation methods have been selected for our experiments. They are listed in Table 6.2, where each of them represents a distinct approach to text augmentation.

For each training sample, the following augmentation strategies are employed:

- **EDA**: We change 0.05% of words in each sentence using simple rule-based transformation.
- **Backtranslation:**Each sentence is translated to French and then back to English to introduce linguistic variations.
- **Contextual Augmentation using TinyBERT:** For each potential replacement word, the top 15 alternatives are generated using a BERT-based Masked Language Model. A replacement occurs with a 40% probability. However, due to the nature of LLMs, the actual number of unique augmentations may vary.

So the training data sizes are doubled for EDA and BT. But for contextual augmentation, the resulting size of the training dataset varies.

6.3.3 Text Classification Model

We utilize a CNN, a prevalent model in supervised learning for text classification (Bayer et al., 2022). Key aspects of the architecture are:

- **Embedding Dimension:** 300-dimensional embedding that is compatible with FastText embeddings.
- **Filters:** A 1D Convolutional layer with 128 filters for feature extraction.
- **Activation Function:** Softmax for the final (output) layer; Rectified Linear Unit (ReLU) otherwise.
- **Pooling:** A Global MaxPooling layer to reduce dimensionality and highlight significant features.
- **Architecture:** Comprises an embedding layer, a convolutional layer, a global max pooling layer, and two fully connected layers.

6.3.4 Results and Analysis

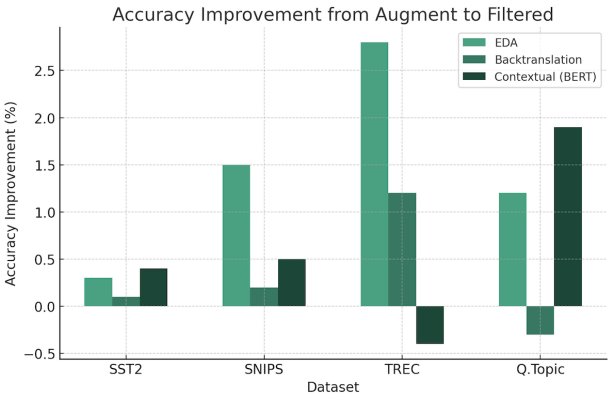
The experimental results are shown in Table 6.3. Improved classification performance across all four metrics: accuracy, precision, recall, and F1 score is achieved using GATFilter, for all datasets and most augmentation strategies.

Table 6.3 Baseline, Augment and Filtering Results

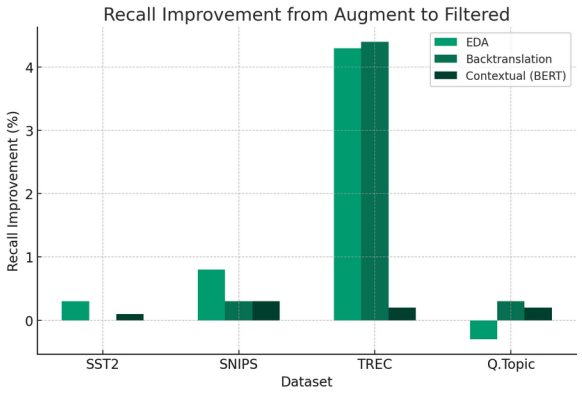
Aug.	Corpus	test_acc. (%)			test_precision (%)			test_recall (%)			test_f1 (%)		
Method		Baseline	Aug.	GATFilter	Baseline	Aug.	GATFilter	Baseline	Aug.	GATFilter	Baseline	Aug.	GATFilter
EDA	SST2	77.6	80.5	**80.8**	77.9	80.6	**80.9**	77.7	80.5	**80.8**	77.6	80.4	**80.8**
	SNIPS	87.1	88.7	**90.4**	87.5	90.3	90.0	87.1	88.7	**90.1**	87.1	88.9	**90.0**
	TREC	84.4	87.6	**92.0**	84.1	84.8	**91.2**	77.9	82.4	**86.0**	79.0	83.2	**88.1**
	Q.Topic	94.6	95.9	**98.0**	95.6	96.3	**98.0**	94.5	95.4	**98.0**	94.8	95.8	**98.0**
backtranslation	SST2	80.1	80.5	**80.6**	80.3	80.7	80.6	80.1	80.5	**80.6**	80.0	80.4	**80.6**
	SNIPS	90.6	89.9	**91.1**	91.4	91.1	91.4	90.6	89.9	**91.1**	90.7	90.0	**91.1**
	TREC	86.0	89.2	**90.4**	83.6	86.3	**92.0**	79.4	82.1	**86.7**	80.8	83.6	**88.7**
	Q.Topic	94.8	96.6	96.5	94.7	96.8	96.2	94.3	96.3	**96.5**	94.5	96.6	96.3
CA (BERT)	SST2	76.3	76.2	**76.6**	76.5	76.2	76.5	76.3	76.2	**76.5**	76.3	76.2	**76.6**
	SNIPS	89.4	89.7	**91.9**	90.2	91.0	**92.2**	89.4	89.7	**91.9**	89.5	89.9	**91.9**
	TREC	82.8	81.4	81.0	81.7	82.3	82.0	80.7	76.0	**76.2**	80.6	78.1	77.0
	Q.Topic	73.3	75.5	**77.9**	75.8	76.4	**77.5**	71.9	74.0	**75.7**	71.6	73.7	**76.0**

6.3.5 Effects on Augmentation Strategies

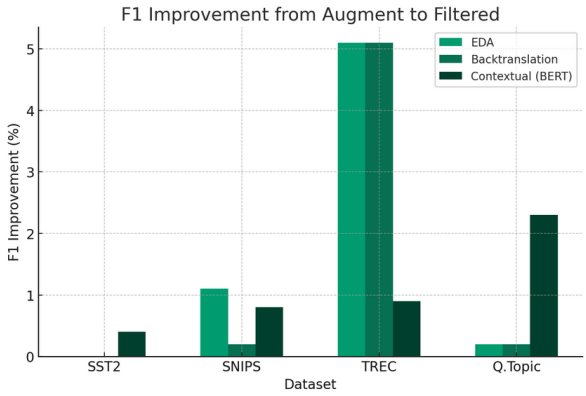
The improvements provided by GATFilter to each of the three augment methods are visually depicted in Figure 6.1. Among the augmentation strategies, EDA emerged as the method



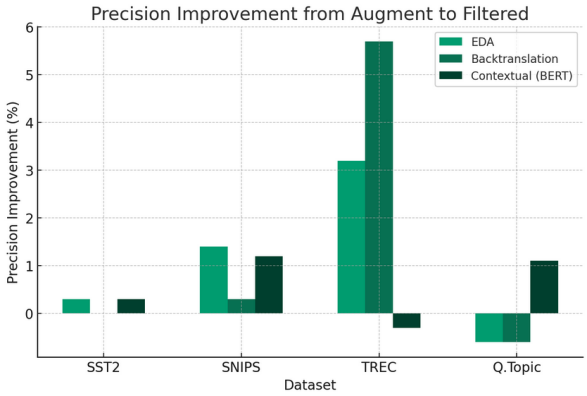
(a) Accuracy



(b) Recall



(c) F1



(d) Precision

Fig. 6.1 Filter Method Testing Performance

that benefitted the most, with enhancements in test performance up to 8% across all datasets and metrics. On the other hand, Backtranslation showed mixed results. For instance, in the TREC dataset, it showed notable improvements, especially in precision and F1 scores. However, its impact varied across other datasets, indicating its effectiveness is highly dependent on the characteristics of the dataset.

Contextual Augmentation using BERT displayed variable performance as well. While it significantly enhanced the F1 score for the Question Topic dataset, it demonstrated less or even negative impacts with TREC. This variability suggests that the effectiveness of filtering contextually augmented data is dataset dependent.

6.3.6 Performance Trends

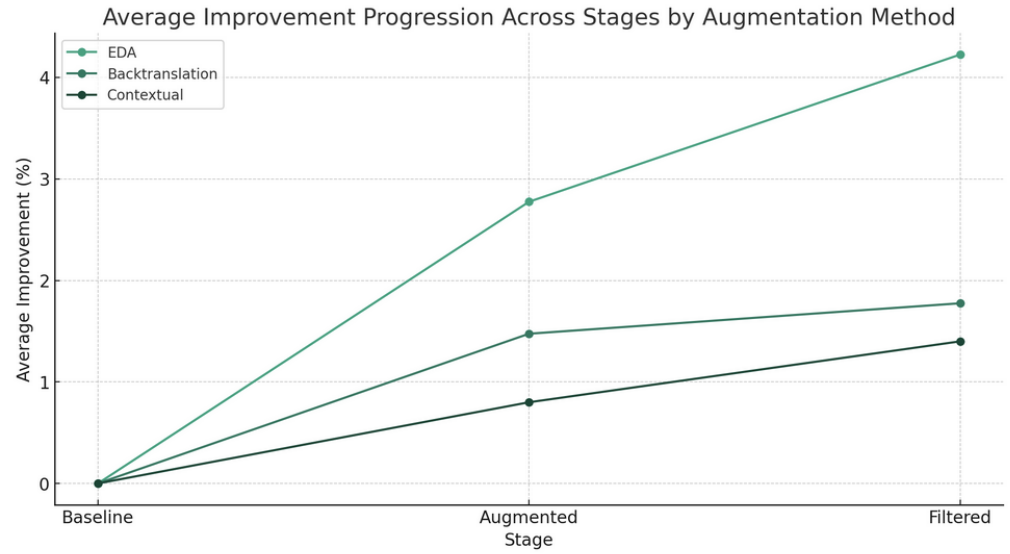
On average, GATFilter does provide improved classification accuracies across all three augmentation methods and the four datasets as shown in Figure 6.2. The effectiveness of GATFilter tends to be amplified when the augmentation method itself contributes positively to the training dataset. On the other hand, when the augmentation method is less effective, as observed with the contextual method in the TREC dataset for example, the ability of GATFilter to enhance performance is relatively limited. Further investigation is warranted to validate this hypothesis.

6.3.7 Geometric Analysis

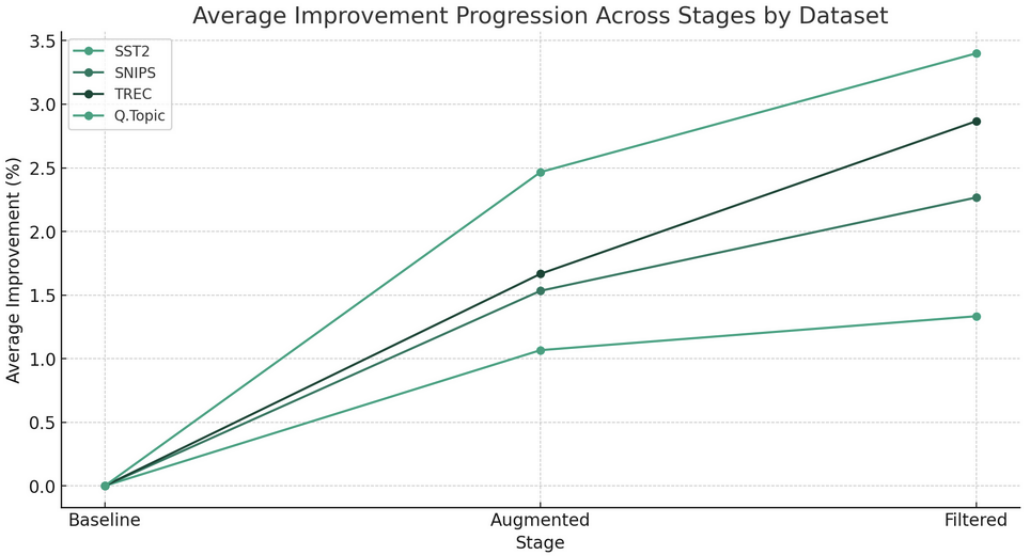
Figure 6.3 shows the convex hull of label 0 of the baseline TREC dataset. The data points shown in this figure include those of the augmented data. We can see that many augmented data points are located outside the convex hull. Those data points would be removed by GATFilter. Hence, it is not surprising that the classification accuracy for this label showed the most improvements after filtering.

We compare a TREC convex hull with contextual augmented data points (which benefited the most) against SNIPS convex hull with back-translated augmented data points (which had only minimal improvement).

Figure 6.4 shows the convex hull of the label 0 of the baseline SNIPS dataset together with the augmented data points. In contrast to Figure 6.3, there are much less data points outside of the convex hull. This means that the GATFilter would have removed less augmented data in this case. As a result, the improvement in classification accuracy is minimal after applying the GATFilter.



(a) Method



(b) Dataset

Fig. 6.2 Classification accuracy by stages

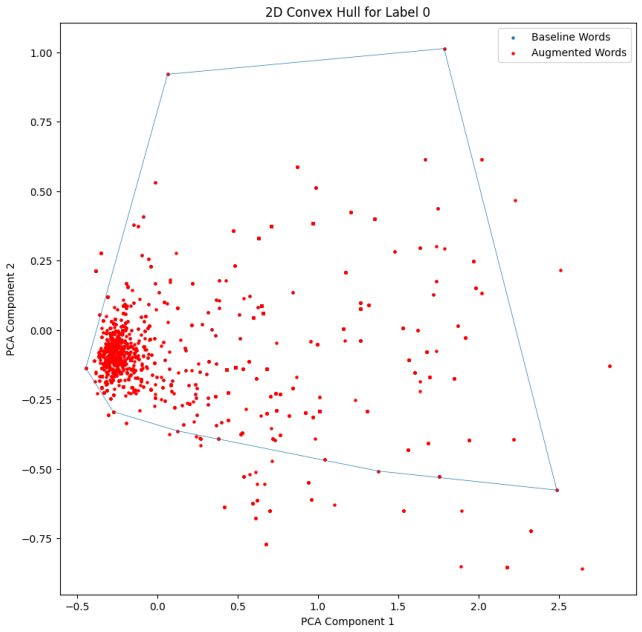


Fig. 6.3 TREC Label 0 Convex Hull (CA)

Table 6.4 Samples of original, augmented, and filtered text from TREC and SNIPS datasets.

Notes	**Original**	**Augmented**	**Filtered**
TREC, CA	What is the correct way to abbreviate cc. at the bottom of a business letter?	That is the correct way to aback ci. at the bottom of a financial letter?	that is the correct way to . at the bottom of a financial letter?
SNIPS, BT	What's the weather at my current location?	What is the weather forecast for my current location?	What is the weather forecast for my current location?

These observations provided some justification to the use of convex hull for filtering.

It is interesting to see the sentences produced by the augmentation processes based on the original and the resulting sentences after GATFilter. Table 6.4 shows a sample from label 0 of the TREC dataset. The contextually augmented version of the sentence has transformed the words in such a way that the original meaning was lost. The GATFilter then removed out words "aback" and "ci" which are outside of the convex hull, resulting in an incomplete sentence. The word "financial" is retained since it is semantically close enough to "business" for its data point to be inside the convex hull.

Another sample taken from label 0 of the SNIPS dataset is shown in Table 6.4. In this case, the augmented sentence is generated by the backtranslation method. Also, the augmented sentence retains the original meaning. When passed to the GATFilter, it presents an augmentation that does not need any filtering.

Overall, the findings indicate a correlation between the number of words in the augmented sentences that are outside the convex hull and the improvement in accuracy after GATFiltering.

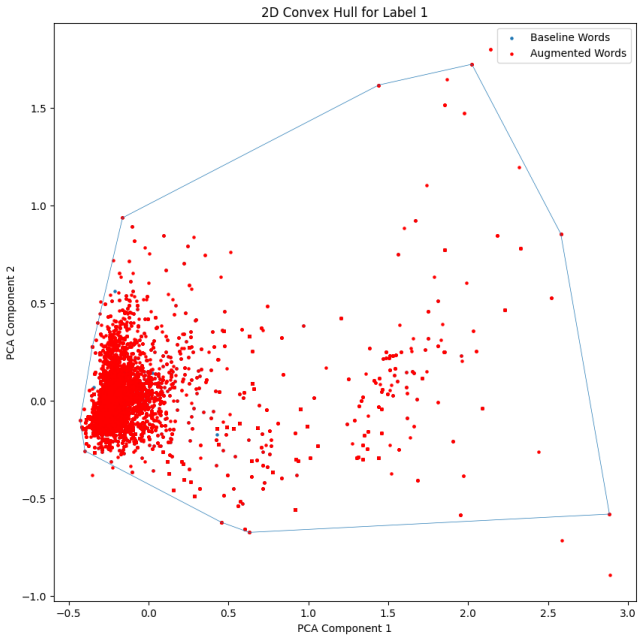


Fig. 6.4 SNIPS Label 0 Convex Hull (BT)

This shows that GATFilter effectively removes words that do not contribute meaningfully to the sentence, enhancing the quality of augmented datasets.

6.4 Limitations and Future Work

While the GATFilter algorithm introduces a novel approach to filtering augmented text data through geometric analysis, several limitations and opportunities for future research should be acknowledged.

A primary limitation of the current implementation is GATFilter's word-level focus. The algorithm operates primarily at the word level, which may not fully capture the nuanced semantics of phrases or sentences. This limitation could lead to the retention or exclusion of words that might be contextually relevant within a broader linguistic structure. Future research could address this by extending GATFilter to incorporate phrase-level or sentence-level semantics, potentially enhancing its ability to preserve contextual relevance and improve overall filtering accuracy. This might involve integrating more sophisticated linguistic models or embeddings that capture higher-order semantics.

The effectiveness of GATFilter is also partly dependent on the quality of the initial augmented data. If the augmentation method generates data that is inherently low quality or semantically inconsistent, GATFilter's ability to improve performance may be limited. Future development

could focus on coupling GATFilter with more advanced augmentation techniques to ensure higher-quality initial data, thereby maximizing the filter's effectiveness.

Finally, while GATFilter has shown effectiveness across several text classification tasks, its applicability to other NLP tasks, such as machine translation or sentiment analysis in more complex domains, remains to be fully explored. Future work should focus on extending the application of GATFilter to a broader range of NLP tasks to validate its generalizability and uncover new use cases. This would involve adapting the algorithm to meet the specific needs of different NLP tasks and datasets.

6.5 Summary

In this chapter, we addressed **RQ4** and **RQ5** by introducing GATFilter, a novel algorithm that extends the GAT framework to improve the quality of augmented text data through geometric analysis. By leveraging GAT's approach to convex hull analysis of dimensionally reduced word embeddings, GATFilter provides a method-independent approach to filtering augmented data, ensuring that only semantically relevant augmentations are retained.

Our experiments demonstrated that GATFilter significantly enhances classification performance across various datasets and augmentation strategies, particularly excelling with EDA and showing promising results with backtranslation methods. The ability to filter out irrelevant or out-of-context words while retaining meaningful augmentations underscores GATFilter's effectiveness as a standalone filtering mechanism, that builds upon GAT's geometric principles.

Despite its strengths, GATFilter's reliance on word-level analysis, PCA for dimensionality reduction, and fixed geometric boundaries presents certain limitations, particularly in capturing complex linguistic structures and scaling to larger datasets. However, these challenges also highlight opportunities for future research, including exploring phrase-level semantics, alternative dimensionality reduction techniques, and adaptive boundary models.

We believe that the geometric perspective introduced by GAT and implemented in GATFilter offers valuable insights for the field of NLP. As we look ahead, we continue to explore applications of GAT's geometric principles in the following chapter, where we introduce GATA (Geometric Approach to Textual Augmentation). While GATFilter uses GAT to filter existing augmented data, GATA leverages GAT's geometric analysis capabilities to generate new augmentations by exploring the semantic space defined by word embeddings. This complementary approach

demonstrates how GAT's foundational principles can be applied to both the generation and filtering aspects of text augmentation, further advancing the state of NLP models.

Chapter 7

A Geometric Approach to Textual Augmentation

In Chapter 5, we introduced GAT, establishing fundamental principles for analyzing word embeddings through a geometric lens. This geometric framework led to two distinct but complementary applications: GATFilter previously (Chapter 6), which leverages GAT's principles for filtering augmented data, and now GATA (Geometric Approach to Textual Augmentation), which extends GAT's geometric analysis capabilities to the augmentation process itself.

While GATFilter applies GAT's geometric principles to evaluate and filter existing augmented data, GATA represents another practical application of GAT's framework toward the generation of new augmented data. By integrating GAT's dimensional reduction and convex hull analysis directly into the augmentation process, GATA seeks to enhance the semantic consistency of generated data while maintaining appropriate diversity through its geometric constraints, ultimately improving the performance of NLP models.

GATA is a novel, resource-effective algorithm that generates high-quality augmented data by exploring the geometric spaces defined by GAT. This approach aligns with our overarching goal of developing efficient NLP models that maintain high performance while reducing resource intensity, as outlined in Chapter 1. The algorithm addresses research question **RQ4**, **RQ5** and **RQ6** stated in Section 1.3.

Together with GATFilter, GATA demonstrates how GAT's topological and computational geometry can be translated into complementary tools for text augmentation - one for generation and one for filtering - creating a comprehensive framework for improving NLP technology through geometric analysis.

7.1 Generative Augmentation Models

7.1.1 Background and Review

Generative augmentation models have emerged as a powerful approach for data augmentation in NLP tasks. These models leverage the generative capabilities of deep learning architectures, such as variational autoencoders (VAEs) (Kingma and Welling, 2022), generative adversarial networks (GANs) (Goodfellow et al., 2014), and language models (Radford et al., 2019), to generate synthetic examples for augmentation.

One of the early works in generative augmentation for NLP is the use of VAEs for text generation (Bowman et al., 2016). VAEs learn a latent space representation of the input text. They can then generate new examples by sampling from this latent space. The generated examples capture the semantic and syntactic properties of the original data, making them suitable for data augmentation. However, VAEs often suffer from the "posterior collapse" problem, where the model learns to ignore the latent variables and generates generic or irrelevant examples (Xu et al., 2020).

GANs have also been explored for text generation and augmentation (Li et al., 2018; Yu et al., 2017). The generator network of a GAN is trained to produce synthetic examples that are indistinguishable from real examples. Its discriminator network attempts to distinguish between real and generated examples. The generator and discriminator are trained adversarially, leading to the generation of high-quality examples. However, training GANs for text generation is challenging due to the discrete nature of text and the lack of a stable training objective (Caccia et al., 2020). LLMs, such as GPT (Radford et al., 2019) and BERT (Devlin et al., 2019), have shown remarkable performance in various NLP tasks, including text generation. These models are pre-trained on large-scale unlabeled text corpora and can be fine-tuned for specific tasks. By sampling from the learned probability distribution of the language model, new examples can be generated that capture the linguistic patterns and semantic relationships of the training data. In addition, language model-based augmentation has been successfully applied to tasks such as text classification (Kumar et al., 2019), machine translation (Sennrich et al., 2016), and question answering (Yang et al., 2020).

The experiments conducted in Chapter 5 investigated the performance of LLMs, particularly GPT-J, in the context of textual augmentation. The results have shown that LLMs excel at generating augmented examples that retain the semantic meaning of the original data, leading to improved performance on downstream NLP tasks.

7.1.2 Limitations of Existing Approaches

Despite the success of LLMs in textual augmentation, there are several limitations to their current usage.

- **Computational complexity**: The training and inference of LLMs can be computationally expensive, requiring significant computational resources and time. This computational complexity can hinder the scalability and practicality of using LLMs for augmentation in real-world applications, especially when dealing with large datasets or limited computational budgets (Patterson et al., 2021; Strubell et al., 2019).
- **Lack of control over the generation process**: LLMs often generate augmented examples in an uncontrolled manner, making it difficult to steer the generation towards specific desired properties or linguistic variations. The lack of fine-grained control over the generation process limits the ability to customize the augmented examples to suit specific task requirements or domain characteristics (Holtzman et al., 2020; Keskar et al., 2019).

7.2 GATA

To address the issues above, we propose GATA (Geometric Approach to Textual Augmentation), an algorithm that extends GAT's principles for text augmentation purposes. While GATFilter applies GAT for filtering augmented data, GATA leverages GAT's geometric analysis capabilities for the generation of new augmented examples.

GATA builds upon the insights gained from GAT's analysis of word embeddings through the lens of topology and geometry, aiming to leverage the geometric structure of the text embedding space for generating diverse and semantically consistent augmented examples. By extending GAT's dimensional reduction and geometric analysis techniques, GATA directly applies these principles to the generation of new data for text augmentation.

In comparison to GATFilter, which applies GAT's principles of post-augmentation for filtering, GATA incorporates them directly into the generative process. This approach allows for a more controlled and sound augmentation process that maintains the geometric properties established by GAT. The foundation of GATA lies in GAT's geometric understanding of how semantic information is represented in embedding spaces. As demonstrated in Chapter 5, GAT's analysis of the shape and boundaries of word embeddings revealed that operating within specific

geometric constraints leads to improved semantic consistency and performance. GATA builds upon this foundation by incorporating these geometric insights into the augmentation process.

7.2.1 Computational Geometry

GATA builds on GAT's geometric analysis by incorporating additional computational geometry techniques for data augmentation. While GAT uses convex hulls to analyze word embeddings, it does not leverage insights from Delaunay triangulation. GATA addresses this by applying Delaunay triangulation, which divides the space defined by GAT into simplices (e.g., triangles in 2D, tetrahedra in 3D), forming a network of interconnected points that preserves the geometric properties of the original word distribution. Delaunay triangulation is particularly advantageous as it maximizes the minimum angles within the triangles, minimizing sliver triangles and ensuring a more even distribution of points. This uniformity supports the generation of diverse yet semantically coherent augmented data.

7.2.2 Augmentation

Once Delaunay triangulation is complete, GATA employs a **nearest neighbor** approach to identify the closest data points within the triangulated space. These nearest neighbors are used as the basis for generating new data points. By selecting data points that are geometrically close to the original examples within the spaces defined by GAT, GATA ensures that the newly generated data remains semantically consistent with the original dataset.

GATA is designed with resource efficiency in mind, leveraging GAT's geometric analysis capabilities to generate high-quality augmented data without the computational overhead associated with more complex models like LLMs. The algorithm's key steps are illustrated below in 7.2.3.

7.2.3 GATA Algorithm

The GATA algorithm implements the geometric principles discovered in our previous analyses by strategically generating augmentations within the triangulated regions of the original data space. The key innovation lies in Step 9, where new words are selected based on their proximity to baseline data points within the Delaunay triangulation structure. This process ensures that augmented words maintain semantic coherence by remaining within the convex hull boundaries identified as optimal in Chapter 5, while the triangulation provides a principled method for

determining valid augmentation regions. Unlike traditional augmentation methods that may generate semantically inconsistent data points, GATA's geometric constraints prevent the creation of outlier words that could degrade classification performance. The algorithm processes each label separately to maintain class-specific geometric properties, and the "closest to baseline" selection criterion ensures that generated augmentations fall within the optimal distance ranges identified in our distance distribution analysis, thereby maximizing the likelihood of performance improvements while preserving the underlying data structure.

Algorithm 4 GATA (utilizing GAT)

Require: *data_path*, *model_name*, *num_augmentations*
Load model based on *model_name*
Load texts from *data_path*
words ← **GAT.process_text**(*texts*) ▷ Call GAT to process text
word_vectors ← **GAT.PCA**(*words*) ▷ Use GAT for dimension reduction
combined_df ← Initialize an empty DataFrame to store augmented data
for *target_label* in set(labels) **do**
    data_points ← Identify data points with *target_label*
    triangulation ← **DelaunayTriangulation**(*data_points*)
    new_words ← Augment data by selecting new (closest to baseline) words within *triangulation*
    new_data ← Create data entries from *new_words* and *target_label*
    combined_df ← Append *new_data* to *combined_df*
end for
    return *combined_df*

GATA is available as a Python package, leveraging libraries such as NumPy for efficient numerical computations, SciPy for spatial algorithms (like Delaunay triangulation), and Gensim for word embedding operations. The algorithm's modular design allows for easy integration with existing NLP pipelines and frameworks.

7.2.4 Experiment Setup

The effectiveness of GATA is evaluated through a series of experiments designed to assess its performance across multiple NLP tasks. The experiment setup mirrors the one used for GATFilter (Chapter 6), providing a consistent study for comparison. The key components of the experiment setup are as follows:

- **Datasets:** The evaluation is conducted on four widely-used NLP datasets: SST-2, TREC, SNIPS, and Question Topic. These datasets cover a range of text classification tasks,

including sentiment analysis, question classification, and intent detection. For each dataset, we generated 50 augmented data points per class using GATA.

- **Augmentation Methods:** In addition to GATA, three established augmentation methods – Backtranslation, ContextualBERT, and EDA are employed to augment the training data. Each method also generated 50 augmented examples per class, ensuring a fair comparison across methods.

- **Training Setup:** To evaluate the effectiveness of augmentation with varying amounts of original training data, we conducted experiments with three different base training sample sizes: 20, 50, and 80 original examples per dataset. For each base sample size, we then added the 50 augmented examples per class, resulting in final training set sizes of 70 (20+50), 100 (50+50), and 130 (80+50) examples per dataset. This approach allowed us to assess how augmentation impacts performance across different baseline data availability scenarios.

- **Classification Model Architecture:** A CNN model is trained on both the original and augmented datasets to assess the impact of each augmentation method on model performance. The architecture of this model is the same used for GATFilter (Chapter 6).

- **Evaluation Metrics:** The performance of the models is evaluated using standard metrics such as accuracy, precision, recall, and F1-score. These metrics provide a comprehensive view of how well each augmentation method supports the model in learning from the augmented data. Each dataset was tested with varying sample sizes (20, 50, and 80) to ensure a comprehensive analysis of the augmentation methods' performance.

- **Computational Efficiency:** In addition to evaluating model performance, we measure the computational resources required for each augmentation method, including execution time and memory usage. This aspect of the experiment setup addresses **RQ6**, focusing on the practicality of the algorithms in real-world applications.

  - **Execution Time:** The total time taken to perform the data augmentation measured in seconds.

  - **Memory Usage:** The memory consumption during the augmentation process, measured in megabytes (MB).

The results are discussed in the subsequent sections, providing insights into the strengths and limitations of GATA compared to traditional augmentation techniques.

7.3 Results, Analysis & Discussion

Table 7.1 shows that GATA consistently outperforms the baseline methods across all datasets and sample sizes. The computational efficiency results are shown in Table 7.2.

Table 7.1 Augmentation Results

Aug. Method	Corpus	test_acc. (%)			test_precision (%)			test_recall (%)			test_f1 (%)		
	(Test Samples)	(20)	(50)	(80)	(20)	(50)	(80)	20	50	80	(20)	(50)	(80)
GATA	**QT**	**89.7**	**90.9**	**94.3**	**89.8**	**91.7**	**94.7**	**89.7**	**90.5**	**94.0**	**89.8**	**90.4**	**94.3**
	SNIPS	**78.9**	**81.0**	**84.1**	**76.1**	**82.2**	**85.6**	**88.9**	**86.0**	**80.1**	**85.0**	**85.7**	**86.4**
	SST2	**59.5**	**66.0**	**64.9**	**66.0**	**70.4**	**68.4**	**61.8**	**58.1**	**58.3**	**57.9**	**64.8**	**68.2**
	TREC	**39.4**	**43.0**	**55.4**	**49.0**	**53.1**	**57.3**	**50.8**	**54.0**	**63.9**	**40.6**	**45.3**	**54.0**
Backtranslation	QT	74.3	89.5	90.9	84.0	89.3	90.2	75.8	89.7	91.6	76.5	89.2	90.5
	SNIPS	63.1	80.6	83.1	72.6	81.4	84.3	63.1	80.6	80.1	61.7	80.6	83.3
	SST2	53.2	53.7	59.6	53.9	54.1	59.9	53.1	53.7	59.6	49.8	52.6	59.3
	TREC	30.8	38.6	48.8	48.9	36.2	53.2	43.7	51.6	59.7	32.2	36.5	51.3
ContextualBERT	QT	84.4	90.1	75.5	84.3	89.6	76.4	84.3	90.4	74.0	83.8	89.9	73.7
	SNIPS	68.4	75.4	80.1	75.0	79.4	81.7	68.4	70.4	72.1	68.6	68.4	80.6
	SST2	54.1	56.5	60.2	54.6	56.5	60.2	54.1	56.5	60.2	53.0	56.4	60.2
	TREC	27.0	36.2	45.8	35.7	45.8	51.3	41.1	48.1	56.1	26.1	35.6	46.4
EDA	QT	65.8	67.0	66.0	67.0	76.6	76.8	64.1	79.8	76.6	58.7	68.6	70.7
	SNIPS	65.6	74.7	83.1	75.4	77.5	83.9	68.6	64.7	72.1	69.2	74.1	75.1
	SST2	49.9	51.2	58.4	25.0	51.5	58.6	50.0	51.2	58.4	33.3	48.5	58.1
	TREC	27.4	34.0	48.0	39.3	52.4	52.4	38.2	44.5	58.8	21.5	33.4	47.9

Table 7.2 Model Time and Memory Usage

Aug. Method	Corpus	Execution Time (seconds)			Memory Usage (MB)		
	(Samples)	(20)	(50)	(80)	(20)	(50)	(80)
GATA	QT	54.25	116.53	167.58	4.22	20.68	19.07
	SNIPS	65.32	126.17	177.24	5.62	2.17	7.88
	SST2	38.06	48.32	64.03	1.85	5.09	6.77
	TREC	20.05	95.36	125.87	28.6	62.80	6.64
Backtranslation	QT	27.45	50.61	75.34	Initial Load: 3926.41	Avg Usage: 3.56	
	SNIPS	16.15	24.74	34.14			
	SST2	17.37	19.15	24.58			
	TREC	18.41	25.36	33.78			
ContextualBERT	QT	58.23	123.81	188.05	Initial Load: 3920.71	Avg Usage: 3.60	
	SNIPS	23.68	34.57	45.12			
	SST2	49.47	35.45	45.86			
	TREC	30.77	49.16	67.73			
EDA	QT	1.0	1.0	1.0	3.40	3.40	3.42
	SNIPS	1.0	1.0	1.50	3.40	3.40	3.42
	SST2	1.0	1.0	1.0	3.40	3.40	3.42
	TREC	1.0	1.0	1.50	3.40	3.40	3.42

The experimental results demonstrate GATA's effectiveness as a novel text augmentation method, offering significant improvements in performance with moderate computational costs.

Below, we discuss the implications of these findings, potential explanations for GATA's success, and insights into its computational efficiency.

Dataset	Text Sample	GATA Augment Terms
SNIPS (Label 3, Intent: PlayMusic)	Play some 1999 Symphony by Minami Takahashi	1880, 1895, 1898, 1960, 1965, 1977, 1984, music, musical...
QT (Label 2, Product Comparison)	Which has less carbohydrates per serving?	able, about, actually, advantages, later, package, same, something...
TREC (Label 4, Location)	What country is Mount Everest in?	field, from, geographical, out, mountain, at, located, sea...
SST2 (Label 0, Positive Sentiment)	a well-executed spy-thriller	amazing, bad, never, nice, notch, offering, tale...

Table 7.3 GATA Augment Sample Output

7.3.1 Performance Analysis Across Datasets

GATA's performance varies across datasets, reflecting the inherent complexity of each task:

Question Topic

GATA shows the highest overall performance (94.3% accuracy at 80 samples for QT), suggesting it's particularly effective for question-type classification tasks. This superior performance likely stems from GATA's ability to maintain semantic consistency within the augmented data, which is crucial for accurately classifying distinct question types.

SNIPS

The performance on SNIPS (84.1% accuracy at 80 samples) is not as high as QT. This might indicate that question classification tasks benefit more from GATA's augmentation, but perhaps less so than more semantically nuanced tasks like user intent.

SST2

The SST-2 dataset presented a unique challenge for GATA, deviating from the consistent upward performance trend observed in other datasets like QT, SNIPS, and TREC. While accuracy improved from 20 to 50 samples, a slight decrease in accuracy occurred when moving from 50 to 80 samples, dropping from 66.0% to 64.9%. This dip suggests that GATA may face specific challenges in sentiment analysis tasks that are heavily reliant on nuanced contextual

understanding, due to the nature of sentimental tasks which can provide conflicting or ambiguous examples, complicating the model's decision boundaries.

The augmentation results for SST-2 in Table 7.3 provide further insight. Although most of the augmented terms are generally positive, these terms can appear in negative contexts as well. While most of the augmented terms for SST2 are generally positive, some (like "bad" and "never") could be used in negative contexts, depending on phrasing. This contextual dependency complicates the sentiment preservation process, making it challenging for a word-level augmentation technique like GATA to maintain consistent sentiment, potentially leading to the performance dip observed with larger sample sizes.

TREC

The substantial improvement on TREC (55.4% accuracy at 80 samples) indicates GATA's strength in handling diverse question categories, though the lower overall accuracy suggests this is a more challenging dataset.

7.3.2 Computational Efficiency Insights

The computational efficiency of GATA was a critical aspect of our evaluation, focusing on both execution time and memory usage. These metrics are particularly relevant to our goal of developing resource-effective NLP models, as stated in Chapter 1.

Execution Time Patterns

The scaling behavior of GATA's execution time reveals intriguing efficiencies as dataset size increases. We observe that the increase in execution time from 20 to 80 samples is not linear across all datasets. For instance, in the QT dataset, the execution time grows from 54.25 seconds at 20 samples to 167.58 seconds at 80 samples. While this is an increase, it's not the fourfold increase one might expect with linear scaling.

This non-linear growth is even more pronounced in the SNIPS dataset, where execution time increases from 65.32 seconds at 20 samples to 177.24 seconds at 80 samples — only about 2.7 times longer despite a quadrupling of data. The SST2 dataset shows a similar trend, with execution times of 38.06 seconds at 20 samples and 64.03 seconds at 80 samples, less than doubling despite the four-fold increase in data size.

These observations suggest that GATA becomes more efficient in its processing as the dataset size grows. This could be attributed to potential optimizations in the algorithm that come into play with larger datasets, such as more efficient memory management or improved utilization of computational resources. Such behavior is particularly advantageous for real-world applications where scalability to larger datasets is crucial, indicating that GATA may maintain reasonable performance even as data volumes increase substantially.

Memory Usage Analysis

GATA's memory usage varied across datasets, ranging from 1.85 MB for SST2 at 20 samples to 62.80 MB for TREC at 50 samples. While GATA's memory usage was higher than some methods, it consistently avoided the large initial memory load required by Backtranslation and ContextualBERT. This variability indicates that GATA's memory efficiency may depend on the specific characteristics of each dataset, offering room for further optimization.

These observations demonstrate that GATA achieves our goal of resource efficiency while maintaining high performance, offering a viable alternative to more computationally intensive methods in NLP augmentation tasks.

7.3.3 Factors Contributing to Performance Improvements

Several key factors contribute to GATA's effectiveness in improving model performance across various NLP tasks:

- **Semantic Preservation through Geometric Constraints:** The use of the convex hull as a boundary mechanism stands as a fundamental element of GATA's design. Unlike methods that randomly sample from a latent space, such as VAEs or GANs, GATA explicitly constrains new examples within the semantic "space" defined by the convex hull. This approach leads to more controlled, task-relevant augmentations. For instance, in sentiment analysis, augmented examples are more likely to align with the overall sentiment, reducing the risk of sentiment drift.

- **Generation of Balanced Augmentations:** The consistent improvements observed across precision, recall, and F1 scores demonstrate GATA's ability to generate balanced augmentations. This balanced approach effectively addresses both false positives and false negatives, a crucial factor in developing robust classifiers across various NLP tasks. The

balance achieved through GATA's geometric approach helps ensure that models trained on the augmented data perform consistently across different evaluation metrics.

- **Effective Scalability:** GATA demonstrates strong scalability with increasing sample sizes, indicating its ability to leverage additional data effectively. This scalability characteristic proves particularly valuable in scenarios where data collection is ongoing or expandable. As more data becomes available, GATA's geometric framework can adapt and utilize the expanded semantic space to generate increasingly diverse yet relevant augmentations.

7.3.4 Practical Implications

Computational Efficiency

GATA offers significant practical advantages that extend beyond its performance metrics. A key benefit is its remarkable computational efficiency, demonstrated by dramatic reductions in both execution time and memory usage compared to methods like Backtranslation and ContextualBERT. As shown in Table 7.2, GATA's execution time with the SST2 corpus (20 samples) is only 38.06 seconds, compared to 49.47 seconds for ContextualBERT a 23% reduction.

Even more impressive, GATA's TREC implementation runs in just 20.05 seconds for 20 samples, which is 35% faster than ContextualBERT's 30.77 seconds for the same corpus and sample size. When scaled to larger datasets (80 samples), GATA's efficiency advantage remains consistent, with its SST2 implementation completing in 64.03 seconds versus ContextualBERT's 45.86 seconds. In terms of memory usage, GATA demonstrates significant efficiency for the QT corpus at 80 samples, using only 19.07 MB compared to ContextualBERT's equivalent task which requires an initial load of 3920.71 MB—a reduction of over 99% in memory footprint. This efficiency makes GATA particularly suitable for real-world applications where computational resources are limited or where rapid iteration is necessary. Small to medium-sized enterprises or research groups without access to extensive computing infrastructure can effectively implement GATA for their NLP tasks. Moreover, this computational efficiency enables more rapid experimentation and model refinement, potentially accelerating the development cycle for NLP applications.

Interpretability Enhancement for NLP

GATA improves NLP model interpretability through transparent geometric decision-making. Unlike black-box methods, GATA's augmentation choices can be visually explained and quantita-

tively measured. For example, Figure 5.7 demonstrates this interpretability through a Delaunay triangulation visualization, where users can directly observe the geometric relationships between original and augmented terms. The convex hull boundaries, shown in Figures 5.5 and 5.6 provide visual evidence of semantic constraints, making GATA's decision process transparent and verifiable. This level of transparency addresses a critical gap in current augmentation methods, delivering both performance improvements and explainable processes.

Less Computational Power

The reduced computational requirements of GATA also address growing concerns about the environmental impact of AI and ML models. By achieving superior results with significantly less computational power, GATA represents a more sustainable approach to NLP augmentation, aligning with the increasing focus on environmentally conscious AI development practices.

7.4 Limitations and Future Work

While GATA has demonstrated significant improvements over existing augmentation methods, it's important to acknowledge its current limitations and areas for future research.

A key limitation is GATA's task-specific performance variability. As observed in our results, GATA's performance varies across different NLP tasks. For instance, while it excels in question classification tasks (QT dataset), its performance on SST2 is less pronounced. This suggests that GATA's effectiveness may be task-dependent, and further research is needed to understand and potentially mitigate these variations. For instance, creating a sentiment-aware version of GATA that better preserves sentiment polarity during augmentation for tasks like SST2.

GATA's dependence on word embeddings presents another significant constraint. GATA's performance is inherently tied to the quality and coverage of the underlying word embeddings. For rare words or domain-specific terminology not well-represented in the embedding space, GATA's augmentation capabilities may be limited. Future work could explore methods to incorporate domain-specific embeddings or handle out-of-vocabulary words more effectively. For example, exploring the use of contextual embeddings (like those from BERT or GPT) instead of static word embeddings. This could potentially capture more nuanced semantic relationships and improve performance on context-dependent tasks.

This analysis underscores the importance of understanding GATA's interaction with different dataset sizes and task complexities. While GATA generally benefits from larger datasets, tasks

like sentiment analysis may require more careful management of augmentation strategies to avoid introducing conflicting data that could reduce model effectiveness. Additional future research could be directed towards several areas, including:

- **Multi-lingual Augmentation:** Extending GATA to support multi-lingual datasets, investigating how geometric properties of word embeddings translate across languages, and developing language-agnostic augmentation strategies.
- **Adaptive Augmentation:** Developing methods for GATA to adaptively adjust its augmentation strategy based on the characteristics of the input data and the specific requirements of the downstream task.
- **Domain Adaptation:** Investigating how GATA can be adapted for domain-specific tasks where the vocabulary and semantic relationships might differ significantly from general language use.
- **Integration with Active Learning:** Exploring how GATA could be integrated into active learning frameworks to guide the selection of the most informative examples for human annotation.
- **Scalability Studies:** Conducting more extensive scalability studies to understand how GATA's performance and efficiency evolve with very large datasets, and developing optimizations for big data scenarios.

7.5 Summary

GATA is a novel algorithm for NLP data augmentation, expanding on the geometric principles established by GAT and complementing the filtering capabilities of GATFilter.

This evolution from GAT's foundational framework to practical applications in both filtering (GATFilter) and data generation (GATA) directly addresses the thesis's two main objectives: establishing geometric and topological frameworks for analyzing model behavior and developing resource-efficient algorithms based on these insights.

Our experiments demonstrate GATA's effectiveness across various NLP tasks, showcasing the successful application of GAT's geometric principles to data generation. Key performance highlights include:

- **QT Dataset:** 94.3% accuracy with 80 samples, showcasing GATA's strength in question-type classification.
- **SNIPS Dataset:** 84.1% accuracy at 80 samples, demonstrating strong performance in intent recognition.
- **SST2 Dataset:** Improvement from 20 to 50 samples, with a slight decrease from 50 to 80 samples (66.0% to 64.9% accuracy), highlighting challenges in sentiment analysis.
- **TREC Dataset:** 55.4% accuracy at 80 samples, showing capability in handling diverse question categories.

Across all these datasets, GATA consistently outperformed established methods such as Backtranslation, ContextualBERT, and EDA, particularly in scenarios with limited training data. Building on GAT's efficient geometric analysis, GATA achieves significant computational efficiency:

- **Execution Time:** GATA showed non-linear growth in execution time as dataset size increased. For example, in the QT dataset, execution time grew from 54.25 seconds at 20 samples to 167.58 seconds at 80 samples, demonstrating improved efficiency with larger datasets.
- **Memory Usage:** GATA's memory usage varied across datasets but consistently avoided the large initial memory load required by methods like Backtranslation and ContextualBERT. This variability suggests potential for further optimization but already represents a significant efficiency gain.

The development of GATA, alongside GATFilter, demonstrates the versatility of GAT's geometric principles in addressing different aspects of text augmentation. While GATFilter applies these principles to evaluate and filter augmented data, GATA extends them to generate new, semantically consistent examples. This comprehensive approach, grounded in GAT's framework, bridges the gap between understanding and practical application in NLP.

While challenges remain, particularly in highly context-dependent tasks like sentiment analysis, the success of both GATA and GATFilter validates GAT's geometric approach to text analysis. The observed performance variability across different tasks suggests that further refinement of our framework could yield even greater improvements, both in performance and in our understanding of language processing by machines. Looking ahead, the principles underlying

GAT and its applications may have broader implications beyond text augmentation. These insights could potentially influence approaches to data synthesis, semantic analysis, and even model architecture design in NLP.

Overall, the progression from GAT's foundation to its practical applications in GATFilter and GATA represents a significant achievement in meeting both objectives set forth in this thesis. It demonstrates that by developing a stronger foundation in understanding of NLP through geometric analysis, we can create more effective and efficient techniques for both generating and filtering augmented data. As we continue to refine and extend this approach, we anticipate its impact will contribute to more efficient, effective, and semantically grounded NLP systems, ultimately advancing our ability to develop AI systems that can better understand and generate human language while remaining computationally viable for real-world applications.

Chapter 8

Conclusion

This thesis has explored the intersection of TDA, computational geometry and NLP, with a focus on improving text classification through data augmentation techniques.

We began by structuring it as two different objectives, which formed our research questions and guided our exploration from principled approaches to practical applications. These objectives have guided our exploration from theoretical foundations to practical applications, leading us through a series of investigations and developments:

- Examining the phenomenon of neural collapse in NLP models, providing new insights into the behavior of these models during training.
- Applied manifold theory to textual data, developing a deeper understanding of the geometric properties of word embeddings and their implications for data augmentation.
- Utilized techniques such as TDA, convex hull and Delaunay triangulation to analyze the inner workings of NLP models, revealing how these mathematical frameworks can enhance model interpretability.
- Building on these insights, we introduced two novel algorithms: GATFilter, a geometric approach to filtering augmented data, and GATA, a geometry-based text augmentation method.

In the following sections, we will synthesize the key findings from this research, discuss their broader implications for the field of NLP, and outline future research directions that emerge from our work.

8.1 Summary of Key Findings and Contributions

8.1.1 Principled Frameworks using Computational Geometry

Objective 1: Establish Geometric and Topological Frameworks for Analyzing NLP Model Behaviour

Insights from the use of computational geometry and TDA have not only deepened our understanding of NLP models but have also laid the groundwork for practical applications. By translating these concepts into actionable tools and methodologies, we have demonstrated how such tools can directly inform and enhance algorithm development in NLP. Below are the specific research questions that we have answered.

RQ1: What insights do topological approaches provide into the inner workings and physical structure of deep neural networks processing textual data?

Our investigation into this question, primarily through the study of neural collapse in NLP models in Chapter 3, revealed several key insights:

1. **Occurrence of Neural Collapse in NLP Models:** We demonstrated, for the first time, that the phenomenon of neural collapse occurs in textual data, specifically in text classification tasks using CNNs. This finding extends the understanding of neural collapse beyond computer vision to the domain of NLP. Potential applications from this finding applies to the field of semi-supervised learning.

2. **Relationship Between Model Architecture and Topological Features:** Our studies suggested a correlation between the architecture of NLP models and the topological features of their learned representations. This insight opens avenues for architecture optimization based on desired topological properties.

3. **Implications for Model Interpretability:** The simplified structure resulting from neural collapse suggests potential avenues for improving model interpretability. By focusing on these simplified representations, we may develop more transparent explanations of model decisions.

RQ2: What do topological and geometric analyses reveal about the mechanisms by which NLP models interpret meaning and learn from training data?

RQ3: What are the key differences in topological and geometric properties between high-performing LLMs and poor-performing models in data augmentation tasks?

Our investigations into these questions was based on Chapters 4 and 5, which revealed the following insights:

1. **Topological Structure Preservation:** Topological data analysis revealed that high-performing LLMs better preserved the topological features (e.g., clusters, loops) of the original data in the augmented set. This preservation was quantified through smaller bottleneck distances between the original and augmented data distributions.

2. **Semantic Boundaries in Embedding Space:** Convex hull computations revealed that effective NLP models create distinct semantic boundaries in the embedding space. High-performing LLMs like GPT-J generated augmented data points that consistently fall within these boundaries, while poorer-performing models often produced augmentations that extended beyond them.

3. **Impact of Augmentation on Geometric Structure:** Different augmentation methods affected the geometric structure of the data in varying ways. Methods that preserved the original geometric structure (like those generated by LLMs) tended to produce more effective augmentations, suggesting that maintaining geometric relationships is crucial for preserving meaning.

4. **Role of Data Point Connectivity:** Delaunay triangulation analysis showed that increased connectivity between data points within semantic boundaries correlates with improved model performance. Augmentations from LLMs resulted in more uniform and dense triangulations, indicating a more coherent and connected augmented space.

5. **Density and Distribution of Augmented Points:** LLMs tended to generate a higher density of augmented points within the original data's convex hull. This increased density correlated with improved classification accuracy, suggesting that effective augmentation enriches the existing semantic space rather than expanding it.

These findings highlight that the superior performance of LLMs in data augmentation tasks is closely tied to their ability to preserve and enrich the existing geometric and topological structure of the data. This preservation maintains semantic consistency while introducing beneficial variations, leading to more effective augmentation for downstream NLP tasks.

Overall, these insights gained from topological and geometric analyses have significantly enhanced our understanding of how NLP models process and learn from textual data, providing a new perspective on the mechanisms underlying effective NLP models and data augmentation techniques.

8.1.2 Practical Contributions

Objective 2: Develop NLP algorithms that are both computational and resource efficient for data augmentation.

This integration of the findings and practice is crucial. Our geometric framework findings from objective 1 leads us to this section of practical implementations that embody these insights. Below are the specific questions we have addressed.

RQ4: To what extent do the data augmentation algorithms developed based on geometric principles improve model performance while maintaining computational efficiency?

RQ5: What quantifiable performance differences exist between the developed algorithms and existing methods across standard NLP benchmarks?

RQ6: In what ways do the computational resource requirements of the developed algorithms differ from those of resource-intensive LLMs, particularly in terms of time, memory, and energy consumption?

We ultimately developed and evaluated two novel algorithms. GATFilter (Chapter 6) and GATA (Chapter 7).

1. **GATFilter:** A geometry-based method for filtering augmented data, which improved the quality of augmented datasets and showed performance improvements (ranged from 2-8%)

across various NLP datasets (SST2, SNIPS, TREC, Question Topic) and augmentation strategies (EDA, Backtranslation, Contextual Augmentation using BERT). GATFilter [1][2] is available online and accessible as Python packages.

2. **GATA:** A geometry-based text augmentation method that leveraged the geometric properties of word embeddings to achieve high accuracy on several benchmark datasets. GATA demonstrated significant memory efficiency, using considerably less memory compared to other augmentation methods like TinyBERT. These findings provide a strong case for the value of incorporating geometric insights into NLP augmentation techniques, potentially leading to more effective and resource-efficient NLP systems.

3. **Scalability and Resource Efficiency** Our work demonstrates that geometrically-informed approaches can lead to algorithms that are both scalable and resource-efficient. This finding, particularly evident in the performance of GATA, addresses growing concerns about the computational demands of state-of-the-art NLP models.

8.2 Limitations and Future Perspectives

8.2.1 Current Limitations

While our research has yielded significant theoretical insights and practical advancements, it is important to acknowledge the limitations of our work:

Dataset and Language Constraints: Our study primarily focused on English language datasets and a limited number of NLP tasks. The generalizability of our findings to other languages and a broader range of NLP applications remains to be fully explored.

Architectural Specificity: The research largely centered on specific model architectures, particularly CNNs for text classification. The applicability of our methods to other architectures requires further investigation.

Word-Level Analyses: Our perspective is primarily based on word-level analyses, focusing on the geometric properties of words within a convex hull. This word-level focus might miss the broader contextual and syntactic structures that span across phrases or sentences, which could lead to a reduction in the richness of linguistic relationships captured by the model.

[1]https://github.com/SherryFeng123/gatfilter
[2]https://pypi.org/project/gatfilter/

8.2.2 Future Research Directions

New Perspectives through Topology and Geometry Frameworks

Our research introduces a paradigm shift in how synthetic linguistic data is generated and utilized. The geometric methods developed for data augmentation focus on preserving and enhancing the inherent geometric structure of the data, leading to more effective and contextually relevant augmentations. This approach challenges traditional methods by emphasizing the importance of maintaining the integrity of the data's geometric relationships, which is particularly beneficial in low-resource settings where data scarcity is a significant challenge. Additionally, these methods contribute to improved model efficiency by ensuring that the augmented data is both high-quality and computationally manageable. The success of this approach suggests that geometric principles could play a central role in the future of data augmentation strategies, optimizing both the quality of the data and the efficiency of the models.

Future Directions: Moving forward, research could explore how these geometric principles can be further refined and applied to a broader range of NLP tasks and datasets. This could include developing advanced algorithms that account for the contextual and syntactic structures beyond the word level, potentially enhancing the richness of the data generated. Additionally, exploring the scalability of these methods for large-scale models and datasets will be crucial for their broader adoption.

Model Interpretability

The geometric visualization tools introduced in this research offer novel ways to interpret and explain the decisions made by NLP models. These tools provide a clearer understanding of the semantic spaces in which models operate, offering insights into the decision-making processes of complex models. Improving interpretability is crucial for building trust in AI systems, particularly in high-stakes applications where transparency is essential. By enabling a deeper understanding of how models arrive at their decisions, these tools contribute to the development of more explainable and user-friendly AI systems, enhancing the relationship between human users and AI technologies.

Future Directions: Looking ahead, further research could focus on developing more sophisticated visualization techniques that capture the complexity of high-dimensional semantic spaces

while remaining accessible to non-expert users. Additionally, extending these interpretability tools to handle more complex NLP tasks and models, such as those involving multi-lingual data or conversational AI, would be a valuable direction. Investigating the potential for these tools to also assess and mitigate biases in AI decision-making processes is another critical area for future work.

8.3 Concluding Remarks

Through exploring neural collapse, manifold theory, and developing geometry-aware algorithms like GATFilter and GATA, we have demonstrated the impact of a principled framework in advancing NLP.

Our research shows that applying computational geometry to language processing not only deepens our understanding of text processing but also enables the creation of more efficient, accurate, and interpretable tools across NLP tasks. This approach has improved classification accuracy while reducing computational demands, addressing critical challenges in modern NLP, including model interpretability and resource efficiency.

Furthermore, this work opens new avenues for interdisciplinary collaboration, inviting mathematicians, linguists, and computer scientists to jointly investigate the intricate structures underlying language. The principles and methods introduced here may serve as a foundation for developing more intuitive, explainable, and powerful NLP systems.

While this thesis provides significant insights and practical advancements, it also highlights the vast potential of computational geometry analyses in NLP. As NLP continues to play an increasingly critical role in our digital world, approaches grounded in mathematical concepts, like those presented here, hold promise for shaping the future of human-machine communication.

This work is not an endpoint but a stepping stone towards a deeper understanding of language and computation, and we hope these ideas inspire further exploration.

Thank you.

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. (2018). Sanity checks for saliency maps. *Advances in neural information processing systems*, 31.

Anaby-Tavor, A., Carmeli, B., Goldbraich, E., Kantor, A., Kour, G., Shlomov, S., Tepper, N., and Zwerdling, N. (2020). Do Not Have Enough Data? Deep Learning to the Rescue! In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7383–7390. AAAI Press.

Andreas, J. (2020). Good-enough compositional data augmentation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Armstrong, M. (2013). *Basic Topology*. Undergraduate Texts in Mathematics. Springer New York.

Arora, R., Basu, A., Mianjy, P., and Mukherjee, A. (2018). Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*.

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. (2016). A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Arras, L., Montavon, G., Müller, K.-R., and Samek, W. (2017). Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 159–168.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996a). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483.

Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996b). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483.

Bayer, M., Kaufhold, M.-A., and Reuter, C. (2022). A Survey on Data Augmentation for Text Classification. *ACM Computing Surveys*, 55(7):146:1–146:39.

Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I. W., Ng, L. G., Ginhoux, F., and Newell, E. W. (2019). Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, 37(1):38–44.

Belinkov, Y., Poliak, A., Shieber, S., Van Durme, B., and Rush, A. (2019). Don't take the premise for granted: Mitigating artifacts in natural language inference. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 877–891, Florence, Italy. Association for Computational Linguistics.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Bengio, Y., Ducharme, R., and Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.

Bengio, Y. and Senécal, J.-S. (2003). Quick training of probabilistic neural nets by importance sampling. In *International Workshop on Artificial Intelligence and Statistics*, pages 17–24. PMLR.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Berger, A., Della Pietra, S. A., and Della Pietra, V. J. (1996). A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., and Kalai, A. T. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer.

Bouraoui, Z., Gutiérrez-Basulto, V., and Schockaert, S. (2022). Integrating ontologies and vector space embeddings using conceptual spaces. In *International Research School in Artificial Intelligence in Bergen (AIB 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Bowman, S., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Bowyer, A. (1981). Computing dirichlet tessellations. *The computer journal*, 24(2):162–166.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

Brown, P. F., Della Pietra, V. J., Desouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Caccia, M., Caccia, L., Fedus, W., Larochelle, H., Pineau, J., and Charlin, L. (2020). Language gans falling short. In *International Conference on Learning Representations (ICLR)*.

Camacho-Collados, J. and Pilehvar, M. T. (2018). From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788.

Camastra, F. and Staiano, A. (2016). Intrinsic dimension estimation: Advances and open problems. *Information Sciences*, 328:26–41.

Cao, Y. and Wang, L. (2017). Automatic selection of t-sne perplexity. In *ICML 2017 AutoML Workshop*, volume 1 of *JMLR Workshop and Conference Proceedings*, pages 1–7.

Carlsson, G. (2009). Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308.

Carlsson, G. (2014). Topological pattern recognition for point cloud data. *Acta Numerica*, 23:289–368.

Casadio, M., Komendantskaya, E., Rieser, V., Daggitt, M. L., Kienitz, D., Arnaboldi, L., and Kokke, W. (2022). Why Robust Natural Language Understanding is a Challenge. *arXiv preprint arXiv:2206.14575*. Accessed on 2024-06-23.

Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate behavioral research*, 1(2):245–276.

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10:377–409.

Che, Z., Purushotham, S., Khemani, R., and Liu, Y. (2016). Interpretable deep models for icu outcome prediction. In *AMIA annual symposium proceedings*, volume 2016, page 371. American Medical Informatics Association.

Chen, S., Dobriban, E., and Lee, J. H. (2020). A group-theoretic framework for data augmentation. *Journal of Machine Learning Research*, 21(245):1–71.

Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In Wu, D., Carpuat, M., Carreras, X., and Vecchi, E. M., editors, *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Chomsky, N. (1953). Systems of syntactic analysis. *The Journal of Symbolic Logic*, 18(3):242–256.

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. (2019). What does BERT look at? an analysis of BERT's attention. In Linzen, T., Chrupała, G., Belinkov, Y., and Hupkes, D., editors, *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2005). Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 263–271.

Corneanu, C. A., Escalera, S., and Martinez, A. M. (2020). Computing the testing error without a testing set. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2677–2685.

Costa, C. J., Aparicio, M., Aparicio, S., and Aparicio, J. T. (2024). The democratization of artificial intelligence: Theoretical framework. *Applied Sciences*, 14(18):8236.

Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., Lavril, T., Primet, M., and Dureau, J. (2018). Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. In *Proceedings of the 1st Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU)*, pages 72–79.

Coulombe, C. (2018). Text data augmentation made simple by leveraging nlp cloud apis.

D. A. Eisa, Ahmed I. Taloba, S. S. I. I. (2018). A comparative study on using principle component analysis with different text classifiers. *International Journal of Computer Applications*, 180(31):1–6.

Dai, H., Liu, Z., Liao, W., Huang, X., Cao, Y., Wu, Z., Zhao, L., Xu, S., Liu, W., Liu, N., Li, S., Zhu, D., Cai, H., Sun, L., Li, Q., Shen, D., Liu, T., and Li, X. (2023). Auggpt: Leveraging chatgpt for text data augmentation.

Dao, T., Gu, A., Ratner, A., Smith, V., De Sa, C., and Ré, C. (2019). A kernel theory of modern data augmentation. In *International conference on machine learning*, pages 1528–1537. PMLR.

de Berg, M., Cheong, O., van Kreveld, M., and Overmars, M. (2008). Convex Hulls. In *Computational Geometry: Algorithms and Applications*, pages 243–258. Springer, Berlin, Heidelberg.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Demner-Fushman, D., Chapman, W. W., and McDonald, C. J. (2009). What can natural language processing do for clinical decision support? *Journal of biomedical informatics*, 42(5):760–772.

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

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Doshi, P. (2018). *Using topological data analysis for text classification.* PhD thesis, The University of North Carolina at Charlotte.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. Accessed on 2024-05-01.

Draganov, O. and Skiena, S. (2024). The shape of word embeddings: Recognizing language phylogenies through topological data analysis. *arXiv preprint arXiv:2404.00500.* Accessed: October 30, 2024.

Edelsbrunner, H., Harer, J., et al. (2008). Persistent homology-a survey. *Contemporary mathematics*, 453(26):257–282.

Edelsbrunner, H. and Harer, J. L. (2022). *Computational topology: an introduction*. American Mathematical Society.

Edwards, A., Ushio, A., Camacho-collados, J., Ribaupierre, H., and Preece, A. (2022). Guiding generative language models for data augmentation in few-shot text classification. In Dragut, E., Li, Y., Popa, L., Vucetic, S., and Srivastava, S., editors, *Proceedings of the Fourth Workshop on Data Science with Human-in-the-Loop (Language Advances)*, pages 51–63, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Elshakhs, Y. S., Deliparaschos, K. M., Charalambous, T., Oliva, G., and Zolotas, A. (2024). A comprehensive survey on delaunay triangulation: Applications, algorithms, and implementations over cpus, gpus, and fpgas. *IEEE Access*, 12:12562–12585.

Ethayarajh, K. (2019). How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65. Association for Computational Linguistics.

Fadaee, M., Bisazza, A., and Monz, C. (2017). Data augmentation for low-resource neural machine translation. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Fefferman, C., Mitter, S., and Narayanan, H. (2016). Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049.

Feng, J. H., Lai, E. M.-K., and Li, W. (2023). A study of neural collapse for text classification. In *International Conference on Deep Learning Theory and Applications*, pages 126–142. Springer.

Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., and Hovy, E. (2021). A survey of data augmentation approaches for NLP. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Fu, Z., Tan, X., Peng, N., Zhao, D., and Yan, R. (2018). Style transfer in text: Exploration and evaluation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Galanti, T., György, A., and Hutter, M. (2022). On the role of neural collapse in transfer learning. Presented as a poster at the International Conference on Learning Representations (ICLR) 2022. Available at: https://iclr.cc/virtual/2022/poster/6574.

Gao, W., Peng, M., Wang, H., Zhang, Y., Xie, Q., and Tian, G. (2019). Incorporating word embeddings into topic modeling of short text. *Knowledge and Information Systems*, 61:1123–1145.

Gao, X., Xiao, B., Tao, D., and Li, X. (2010). A survey of graph edit distance. *Pattern Analysis and applications*, 13:113–129.

Gholizadeh, S., Seyeditabari, A., and Zadrozny, W. (2020). A novel method of extracting topological features from word embeddings.

Ghosh, S., Evuru, C. K. R., Kumar, S., Ramaneswaran, S., Sakshi, S., Tyagi, U., and Manocha, D. (2023). DALE: Generative data augmentation for low-resource legal NLP. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8511–8565, Singapore. Association for Computational Linguistics.

Ghrist, R. W. (2014). *Elementary applied topology*, volume 1. Createspace Seattle.

Goldberg, Y. (2022). *Neural network methods for natural language processing*. Springer Nature.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Graham, R. L. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Info. Proc. Lett.*, 1:132–133.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42.

Hady, M. F. A. and Schwenker, F. (2013). Semi-supervised learning. *Handbook on Neural Information Processing*, pages 215–239.

Hameed, A., Sleeman, D., and Preece, A. (2002). Detecting Mismatches among Experts' Ontologies acquired through Knowledge Elicitation. In Bramer, M., Coenen, F., and Preece, A., editors, *Research and Development in Intelligent Systems XVIII*, pages 9–22, London. Springer.

Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016). Diachronic word embeddings reveal statistical laws of semantic change. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Han, X., Papyan, V., and Donoho, D. L. (2022). Neural collapse under MSE loss: Proximity to and dynamics on the central path. In *International Conference on Learning Representations*.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hashimoto, T. B., Alvarez-Melis, D., and Jaakkola, T. S. (2016). Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics*, 4:273–286.

Hatcher, A. (2002). *Algebraic Topology*. Cambridge University Press.

Hayes, P. J. (1981). The frame problem and related problems in artificial intelligence. In *Readings in artificial intelligence*, pages 223–230. Elsevier.

He, F. and Tao, D. (2021). Recent advances in deep learning theory. Accessed on 2024-05-01.

Heimerl, F. and Gleicher, M. (2018). Interactive analysis of word vector embeddings. In *Computer Graphics Forum*, volume 37, pages 253–265. Wiley Online Library.

Hensel, F., Moor, M., and Rieck, B. (2021). A survey of topological machine learning methods. *Frontiers in Artificial Intelligence*, 4:681108.

Hirschberg, J. and Manning, C. D. (2015). Advances in natural language processing. *Science*, 349(6245):261–266.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2020). The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Hou, Y., Liu, Y., Che, W., and Liu, T. (2018). Sequence-to-Sequence Data Augmentation for Dialogue Language Understanding. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Hovy, E., Gerber, L., Hermjakob, U., Lin, C.-Y., and Ravichandran, D. (2001). Toward Semantics-Based Answer Pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Hoy, M. B. (2018). Alexa, siri, cortana, and more: an introduction to voice assistants. *Medical reference services quarterly*, 37(1):81–88.

Hui, L., Belkin, M., and Nakkiran, P. (2022). Limitations of neural collapse for understanding generalization in deep learning.

Iacobacci, I. J., Pilehvar, M. T., Navigli, R., et al. (2016). Embeddings for word sense disambiguation: An evaluation study. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016-Long Papers*, volume 2, pages 897–907. Association for Computational Linguistics (ACL).

Jabbar, H. and Khan, R. Z. (2015). Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices*, 70(10.3850):978–981.

Jain, S. and Wallace, B. C. (2019). Attention is not Explanation. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Jakubowski, A., Gasic, M., and Zibrowius, M. (2020). Topology of word embeddings: Singularities reflect polysemy. In Gurevych, I., Apidianaki, M., and Faruqui, M., editors, *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 103–113, Barcelona, Spain (Online). Association for Computational Linguistics.

Ji, W., Lu, Y., Zhang, Y., Deng, Z., and Su, W. J. (2022). An unconstrained layer-peeled perspective on neural collapse. In *International Conference on Learning Representations*.

Jia, R. and Liang, P. (2017). Adversarial examples for evaluating reading comprehension systems. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. (2020). TinyBERT: Distilling BERT for Natural Language Understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Johnson, R. and Zhang, T. (2015). Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112.

Jolliffe, I. T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Jurafsky, D. (2000). *Speech & language processing*. Pearson Education India.

Kaczynski, T., Mischaikow, K. M., and Mrozek, M. (2004). *Computational homology*, volume 157. Springer.

Kallay, M. (1984). The complexity of incremental convex hull algorithms in rd. *Information Processing Letters*, 19(4):197.

Keskar, N. S., McCann, B., Varshney, L., Xiong, C., and Socher, R. (2019). CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*. Accessed on 2024-08-23.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kingma, D. P. and Welling, M. (2022). Auto-encoding variational bayes.

Kobak, D. and Berens, P. (2019). The art of using t-sne for single-cell transcriptomics. *Nature communications*, 10(1):5416.

Kobak, D. and Linderman, G. C. (2019). Umap does not preserve global structure any better than t-sne when using the same initialization. *BioRxiv*, pages 2019–12.

Kobayashi, S. (2018). Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457. Association for Computational Linguistics.

Kothapalli, V. (2023). Neural collapse: A review on modelling principles and generalization. *Transactions on Machine Learning Research*.

Kozlowski, A. C., Taddy, M., and Evans, J. A. (2019). The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, 84(5):905–949.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kumar, V., Choudhary, A., and Cho, E. (2020). Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26.

Kumar, V., Glaude, H., de Lichy, C., and Campbell, W. (2019). A closer look at feature space data augmentation for few-shot intent classification. In Cherry, C., Durrett, G., Foster, G., Haffari, R., Khadivi, S., Peng, N., Ren, X., and Swayamdipta, S., editors, *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Kutuzov, A., Øvrelid, L., Szymanski, T., and Velldal, E. (2018). Diachronic word embeddings and semantic shifts: a survey. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kuzborskij, I. and Lampert, C. (2018). Data-dependent stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 2815–2824. PMLR.

Labani, M., Moradi, P., Ahmadizar, F., and Jalili, M. (2018). A novel multivariate filter method for feature selection in text classification problems. *Engineering Applications of Artificial Intelligence*, 70:25–37.

Lawson, C. L. (1977). Software for c1 surface interpolation. In *Mathematical software*, pages 161–194. Elsevier.

Lebret, R. and Collobert, R. (2014). Word embeddings through hellinger PCA. In Wintner, S., Goldwater, S., and Riezler, S., editors, *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490, Gothenburg, Sweden. Association for Computational Linguistics.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Lee, D.-T. and Schachter, B. J. (1980). Two algorithms for constructing a delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242.

Leykam, D. and Angelakis, D. G. (2023). Topological data analysis and machine learning. *Advances in Physics: X*, 8(1):2202331.

Li, G. and Yu, Y. (2016). Visual saliency detection based on multiscale deep cnn features. *IEEE transactions on image processing*, 25(11):5012–5024.

Li, X., Liu, S., Zhou, J., Lu, X., Fernandez-Granda, C., Zhu, Z., and Qu, Q. (2024). Understanding and improving transfer learning of deep models via neural collapse. *Transactions on Machine Learning Research*.

Li, X. and Roth, D. (2002). Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics*, volume 1, pages 1–7, Taipei, Taiwan. Association for Computational Linguistics.

Li, Y., Pan, Q., Wang, S., Yang, T., and Cambria, E. (2018). A generative model for category text generation. *Information Sciences*, 450:301–315.

Linzen, T. (2019). What can linguistics and deep learning contribute to each other? response to pater. *Language*, 95(1):e99–e108.

Linzen, T. (2020). How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217.

Liu, B. and Zhang, L. (2012). A survey of opinion mining and sentiment analysis. In *Mining text data*, pages 415–463. Springer.

Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., and Tang, J. (2021). Self-supervised learning: Generative or contrastive. *IEEE transactions on knowledge and data engineering*, 35(1):857–876.

Lopes, R. G., Smullin, S. J., Cubuk, E. D., and Dyer, E. (2020). Affinity and diversity: Quantifying mechanisms of data augmentation. *CoRR*.

Lopes, R. H., Hobson, P. R., and Reid, I. D. (2008). Computationally efficient algorithms for the two-dimensional kolmogorov–smirnov test. In *Journal of Physics: Conference Series*, volume 119, page 042019. IOP Publishing.

Lu, J. and Steinerberger, S. (2022). Neural collapse under cross-entropy loss. *Applied and Computational Harmonic Analysis*, 59:224–241.

Lum, P. Y., Singh, G., Lehman, A., Ishkanov, T., Vejdemo-Johansson, M., Alagappan, M., Carlsson, J., and Carlsson, G. (2013). Extracting insights from the shape of complex data using topology. *Scientific reports*, 3(1):1236.

Lyle, C., van der Wilk, M., Kwiatkowska, M., Gal, Y., and Bloem-Reddy, B. (2020). On the benefits of invariance in neural networks.

Manning, C. (1999). *Foundations of statistical natural language processing*. The MIT Press.

Manning, C. D., Raghavan, P., and Sch"utze, H. (2008). *Introduction to information retrieval*. Cambridge university press.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Maria, C. (2016). Persistent cohomology user manual — gudhi documentation.

Marivate, V. and Sefara, T. (2020). Improving short text classification through global augmentation methods. In *Machine Learning and Knowledge Extraction: 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25–28, 2020, Proceedings 4*, pages 385–399. Springer.

Massey Jr, F. J. (1951). The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. (2020). On faithfulness and factuality in abstractive summarization. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

McCoy, T., Pavlick, E., and Linzen, T. (2019). Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018). Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

McKeown, K. R. (1986). Language generation: Applications, issues, and approaches. *Proceedings of the IEEE*, 74(7):961–968.

McNamara, D. S., Crossley, S. A., and Roscoe, R. (2013). Natural language processing in an intelligent writing strategy tutoring system. *Behavior research methods*, 45:499–515.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Team, G. B., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., et al. (2011). Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. In *Workshop Proceedings of the 2013 International Conference on Learning Representations*.

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

Mikolov, T., Yih, W.-t., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Milnor, J. W. (1963). *Morse theory*. Princeton university press.

Mimno, D. and Thompson, L. (2017). The strange geometry of skip-gram with negative sampling. In *Empirical Methods in Natural Language Processing*.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., and Gao, J. (2021). Deep learning–based text classification: a comprehensive review. *ACM computing surveys (CSUR)*, 54(3):1–40.

Mixon, D. G., Parshall, H., and Pi, J. (2022). Neural collapse with unconstrained features. *Sampling Theory, Signal Processing, and Data Analysis*, 20(2):11.

Mu, J. and Viswanath, P. (2018). All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

Munch, E. (2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, 4(2):47–61.

Munkres, J. R. (2018). *Elements of algebraic topology*. CRC press.

Møller, A. G., Dalsgaard, J. A., Pera, A., and Aiello, L. M. (2024). The parrot dilemma: Human-labeled vs. llm-augmented data in classification tasks. Accessed on 2024-10-01.

Naitzat, G., Zhitnikov, A., and Lim, L.-H. (2020). Topology of deep neural networks. *Journal of Machine Learning Research*, 21(184):1–40.

Nash, C. and Sen, S. (1988). *Topology and geometry for physicists*. Elsevier.

Nickel, M. and Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30.

Ning-min, S. and Jing, L. (2015). A Literature Survey on High-Dimensional Sparse Principal Component Analysis. *International Journal of Database Theory and Application*, 8(6):57–74.

Niyogi, P., Smale, S., and Weinberger, S. (2011). A topological view of unsupervised learning from noisy data. *SIAM Journal on Computing*, 40(3):646–663.

Papyan, V., Han, X., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663.

Paradis, F. and Nie, J.-Y. (2007). Contextual feature selection for text classification. *Information Processing & Management*, 43(2):344–352.

Parikh, A., Täckström, O., Das, D., and Uszkoreit, J. (2016). A Decomposable Attention Model for Natural Language Inference. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., and Dean, J. (2021). Carbon emissions and large neural network training. Accessed on 2023-01-23.

Pedrosa, L., Iyer, R., Zaostrovnykh, A., Fietz, J., and Argyraki, K. (2018). Automated synthesis of adversarial workloads for network functions. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 372–385.

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

Perez, I. and Reinauer, R. (2022). The topological bert: Transforming attention into topology for natural language processing. Accessed: October 30, 2022.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018a). Deep contextualized word representations. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peters, M. E., Neumann, M., Zettlemoyer, L., and Yih, W.-t. (2018b). Dissecting contextual word embeddings: Architecture and representation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Queiroz Abonizio, H. and Barbon Junior, S. (2020). Pre-trained Data Augmentation for Text Classification. In Cerri, R. and Prati, R. C., editors, *Intelligent Systems*, Lecture Notes in Computer Science, pages 551–565. Springer International Publishing.

Quirk, C., Brockett, C., and Dolan, B. (2004). Monolingual machine translation for paraphrase generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 142–149. Association for Computational Linguistics.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. Technical report, Technical Report, Open AI. Publisher: OpenAI.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raganato, A., Pasini, T., Camacho-Collados, J., and Pilehvar, M. T. (2020). Xl-wic: A multilingual benchmark for evaluating semantic contextualization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7193–7206.

Rangamani, A. and Banburski-Fahey, A. (2022). Neural collapse in deep homogeneous classifiers and the role of weight decay. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4243–4247. IEEE.

Rathore, A., Zhou, Y., Srikumar, V., and Wang, B. (2023). Topobert: Exploring the topology of fine-tuned word representations. *Information Visualization*, 22(3):186–208.

Raunak, V., Gupta, V., and Metze, F. (2019). Effective Dimensionality Reduction for Word Embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Rawson, M., Dooley, S., Bharadwaj, M., and Choudhary, R. (2022). Topological data analysis for word sense disambiguation. Accessed: October 30, 2022.

Reif, E., Yuan, A., Wattenberg, M., Viegas, F. B., Coenen, A., Pearce, A., and Kim, B. (2019). Visualizing and measuring the geometry of bert. *Advances in neural information processing systems*, 32.

Roth, M. and Lapata, M. (2016). Neural semantic role labeling with dependency path embeddings. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Ruder, S. (2019). *Neural transfer learning for natural language processing*. PhD thesis, NUI Galway.

Sahu, G., Rodriguez, P., Laradji, I., Atighehchian, P., Vazquez, D., and Bahdanau, D. (2022). Data augmentation for intent classification with off-the-shelf large language models. In Liu, B., Papangelis, A., Ultes, S., Rastogi, A., Chen, Y.-N., Spithourakis, G., Nouri, E., and Shi, W., editors, *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 47–57, Dublin, Ireland. Association for Computational Linguistics.

Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. (2011). Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops*, pages 166–171. IEEE.

Saunshi, N., Plevrakis, O., Arora, S., Khodak, M., and Khandeparkar, H. (2019). A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pages 5628–5637. PMLR.

Schmidhuber, J. (2015). Deep learning. *Scholarpedia*, 10(11):32832.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2020). Green ai. *Communications of the ACM*, 63(12):54–63.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Shen, D., Zheng, M., Shen, Y., Qu, Y., and Chen, W. (2020). A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*. Accessed on 2024-05-01.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48.

Sizemore, A. E., Phillips-Cremins, J. E., Ghrist, R., and Bassett, D. S. (2019). The importance of the whole: topological data analysis for the network neuroscientist. *Network Neuroscience*, 3(3):656–673.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S., editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Stephens, M. A. (1974). Edf statistics for goodness of fit and some comparisons. *Journal of the American statistical Association*, 69(347):730–737.

Stolarek, I., Samelak-Czajka, A., Figlerowicz, M., and Jackowiak, P. (2022). Dimensionality reduction by umap for visualizing and aiding in classification of imaging flow cytometry data. *Iscience*, 25(10).

Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Taha, A. and Hadi, A. S. (2019). Anomaly detection methods for categorical data: A review. *ACM Comput. Surv.*, 52(2).

Thorndike, R. L. (1956). Chapter ii: Development and applications of tests of special aptitudes. *Review of Educational Research*, 26(1):14–25.

Thrampoulidis, C., Kini, G. R., Vakilian, V., and Behnia, T. (2022). Imbalance trouble: Revisiting neural-collapse geometry. *Advances in Neural Information Processing Systems*, 35:27225–27238.

Tishby, N. and Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. IEEE.

Toma, P. (1977). Systran as a multilingual machine translation system. In *Proceedings of the Third European Congress on Information Systems and Networks, overcoming the language barrier*, pages 569–581.

Toussaint, G. T. (1989). Computational geometry: Recent developments. In *New Advances in Computer Graphics: Proceedings of CG International'89*, pages 23–51. Springer.

Umer, M., Imtiaz, Z., Ahmad, M., Nappi, M., Medaglia, C., Choi, G. S., and Mehmood, A. (2023a). Impact of convolutional neural network and fasttext embedding on text classification. *Multimedia Tools and Applications*, 82(4):5569–5585.

Umer, M., Imtiaz, Z., Ahmad, M., Nappi, M., Medaglia, C., Choi, G. S., and Mehmood, A. (2023b). Impact of convolutional neural network and FastText embedding on text classification. *Multimedia Tools and Applications*, 82(4):5569–5585.

Van Der Maaten, L. (2009). Learning a parametric embedding by preserving local structure. In *Artificial intelligence and statistics*, pages 384–391. PMLR.

Van Der Maaten, L. (2014). Accelerating t-sne using tree-based algorithms. *The journal of machine learning research*, 15:3221–3245.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

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

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., and Bengio, Y. (2019). Manifold mixup: Better representations by interpolating hidden states. In *International conference on machine learning*, pages 6438–6447. PMLR.

Vig, J. and Belinkov, Y. (2019). Analyzing the structure of attention in a transformer language model. In Linzen, T., Chrupała, G., Belinkov, Y., and Hupkes, D., editors, *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy. Association for Computational Linguistics.

Vrehuuvrek, R., Sojka, P., et al. (2011). Gensim-statistical semantics in python. *Retrieved from Genism*.

Wan, Z., Wan, X., and Wang, W. (2020). Improving Grammatical Error Correction with Data Augmentation by Editing Latent Representation. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019). Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wang, B. (2021). Mesh-transformer-JAX: Model-parallel implementation of transformer language model with JAX. https://github.com/kingoflolz/mesh-transformer-jax. Apache-2.0 License.

Wang, B. and Komatsuzaki, A. (2022). Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021. *URL https://github. com/kingoflolz/mesh-transformer-jax*.

Wang, P., Xu, J., Xu, B., Liu, C., Zhang, H., Wang, F., and Hao, H. (2015). Semantic clustering and convolutional neural network for short text categorization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 352–357.

Wang, Y., Huang, H., Rudin, C., and Shaposhnik, Y. (2021). Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73.

Wasserman, L. (2018). Topological data analysis. *Annual Review of Statistics and Its Application*, 5:501–532.

Watson, D. F. (1981). Computing the n-dimensional delaunay tessellation with application to voronoi polytopes. *The computer journal*, 24(2):167–172.

Wattenberg, M., Viégas, F., and Johnson, I. (2016). How to use t-sne effectively. *Distill*, 1(10):e2.

Weaver, W. (1952). Translation. In *Proceedings of the Conference on Mechanical Translation*.

Wei, J. and Zou, K. (2019). EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388. Association for Computational Linguistics.

Weizenbaum, J. (1966). Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Woods, W. (1972). The lunar sciences natural language information system. *BBN report*.

Woolf, M. (2017). textgenrnn. *Github Repository*.

Wu, L., Xia, Y., Tian, F., Zhao, L., Qin, T., Lai, J., and Liu, T.-Y. (2018). Adversarial neural machine translation. In *Asian Conference on Machine Learning*, pages 534–549. PMLR.

Wu, X., Gao, C., Lin, M., Zang, L., and Hu, S. (2022). Text smoothing: Enhance various data augmentation methods on text classification tasks. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 871–875, Dublin, Ireland. Association for Computational Linguistics.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Xie, Q., Dai, Z., Hovy, E., Luong, T., and Le, Q. (2020). Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268.

Xie, Z., Genthial, G., Xie, S., Ng, A., and Jurafsky, D. (2018). Noising and Denoising Natural Language: Diverse Backtranslation for Grammar Correction. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana. Association for Computational Linguistics.

Xu, P., Cheung, J. C. K., and Cao, Y. (2020). On variational learning of controllable representations for text without supervision. In *International Conference on Machine Learning*, pages 10534–10543. PMLR.

Yang, Y., Malaviya, C., Fernandez, J., Swayamdipta, S., Le Bras, R., Wang, J.-P., Bhagavatula, C., Choi, Y., and Downey, D. (2020). Generative Data Augmentation for Commonsense Reasoning. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Yaras, C., Wang, P., Zhu, Z., Balzano, L., and Qu, Q. (2022). Neural collapse with normalized features: A geometric analysis over the riemannian manifold. *Advances in neural information processing systems*, 35:11547–11560.

Yousefzadeh, R. (2020). Deep learning generalization and the convex hull of training sets. In *NeurIPS 2020 Workshop: Deep Learning through Information Geometry*.

Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. (2018). QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *6th International Conference on Learning Representations*.

Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Zhang, S., Jafari, O., and Nagarkar, P. (2021). A survey on machine learning techniques for auto labeling of video, audio, and text data. Accessed: October 30, 2021.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Zhang, Y. and Wallace, B. (2017). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In Kondrak, G. and Watanabe, T., editors, *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 253–263, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., and Zhou, X. (2020). Semantics-aware bert for language understanding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9628–9635.

Zhou, J., Li, X., Ding, T., You, C., Qu, Q., and Zhu, Z. (2022). On the optimization landscape of neural collapse under mse loss: Global optimality with unconstrained features. In *International Conference on Machine Learning*, pages 27179–27202. PMLR.

Zhu, C., Cheng, Y., Gan, Z., Sun, S., Goldstein, T., and Liu, J. (2020). Freelb: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.

Zhu, X. (2013). Persistent homology: An introduction and a new text representation for natural language processing. In *IJCAI*, pages 1953–1959.

Zhu, X., Lafferty, J., and Ghahramani, Z. (2003). *Semi-supervised learning: From Gaussian fields to Gaussian processes*. School of Computer Science, Carnegie Mellon University.

Zhu, Z., Ding, T., Zhou, J., Li, X., You, C., Sulam, J., and Qu, Q. (2021). A geometric analysis of neural collapse with unconstrained features. *Advances in Neural Information Processing Systems*, 34:29820–29834.

Zomorodian, A. (2010). Fast construction of the vietoris-rips complex. *Computers & Graphics*, 34(3):263–271.

Zomorodian, A. J. (2005). *Topology for computing*, volume 16. Cambridge university press.