Advancing Robust and Aligned Measures of Semantic Similarity in Large Language Models

Samarth Goel

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-84
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-84.html

May 10, 2024

Advancing Robust and Aligned Measures of Semantic Similarity in Large Language Models

Samarth Goel

Spring 2024

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Kannan Ram

Professor Kannan Ramchandran
Research Advisor

March 8, 2024

(Date)

* * * * * * *

Professor Jiantao Jiao
Second Reader

05/09/2024

(Date)

Advancing Robust and Aligned Measures of Semantic Similarity in Large Language Models

by

Samarth Goel

A thesis submitted in partial satisfaction of the

requirements for the degree of

Masters of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Kannan Ramchandran, Advisor
Professor Jiantao Jiao

Spring 2024

Advancing Robust and Aligned Measures of Semantic Similarity in Large Language Models

Abstract

Advancing Robust and Aligned Measures of Semantic Similarity in Large Language Models

by

Samarth Goel

Masters of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Kannan Ramchandran, Advisor

With the increasing usage of text similarity measures in conjunction with Large Language Models (LLMs), greater scrutiny and evaluation methodologies are needed to ensure the correct metric choice for a given task. In this thesis, I will evaluate the ability of text similarity measures to be robust and aligned with a human understanding of semantic similarity and assess the effectiveness of popular LLMs in maintaining semantic understanding. My core contributions are as follows. I develop and introduce the Unified semantic Similarity Metric Benchmark (USMB), a novel leaderboard for text similarity metrics composed of 10+ datasets and original tasks measuring human preference alignment, robustness, sensitivity, and clustering performance. My next contribution is the development of an ensembled text similarity measurement that achieves top scores in all tasks composing the USMB, beating the previously measured best overall score by 48.2%. I also demonstrate the robustness of this ensembled text similarity measurement on popular information retrieval tasks. Lastly, I contribute a new LLM benchmarking task titled Semantic Elasticity, a generalization of summarization that measures a model's ability to compress and expand information and quantify the performance of 6 popular LLMs on this task. I hope that through this work, greater attention can be given to potential performance gains through proper metric treatment and selection and that the field's ability to measure semantic similarity advances as a result.

To my family

Contents

Acknowledgments

To my mentors: Professor Kannan Ramchandran, without whom I wouldn't have had the opportunity to explore academia, Professor Joshua Hug, for allowing me to make an impact in the Berkeley CS Community through education, and Professor Jiantao Jiao, for graciously agreeing to be the second reader for this thesis.

To my friends: Reagan J. Lee, for being a sounding board throughout the year and motivating many of the experiments conducted in this thesis, Chiraag Balu, for inspiring the original questions that developed into this thesis, and Sandeep Mukherjee, for providing feedback and insights on the initial ideas behind this thesis.

Lastly, to my family: My mom, my dad, and my sister. Thank you for everything you've done for me. I wouldn't be where I am without the unending support, belief, and love you've given me.

Chapter 1

Introduction

With the rapid advancement of large language models (LLMs) like OpenAI's ChatGPT, Anthropic's Claude, and Google's Gemini, the technology's potential impact on society is enormous. These tools can revolutionize information accessibility and education while aiding complex problem-solving across diverse industries. Realizing this potential hinges significantly on the models' reliability and efficacy, especially in complex and often unpredictable industry environments, far from the controlled academic scenarios where these technologies are typically tested. The stark contrast between idealized evaluation conditions and the multifaceted realities of deployment creates a critical gap in measuring model capabilities. Despite rapid advancements and frequent updates in AI technology, the practical application of LLMs remains a challenge due to the variable quality of the data they process.

In typical academic settings, text-based data is often clean and high-quality, contrasting the petabytes of unstructured, potentially compromised data that flood into production systems daily. These large-scale applications demand robust, flexible text similarity measures that can handle a broad spectrum of data types and quality, ensuring AI systems can operate across all levels of data integrity while remaining aligned with human preferences.

This thesis explores the need for advanced, resilient text similarity measures tailored to the complexities of real-world data. We begin by reviewing the motivation behind this thesis and the questions we hope to answer. We then move on to an explanation of the background required to understand the scope of the problem, covering text similarity metrics, text embedding models, and current benchmarks. Next, we propose the core contributions of this thesis, the introduction of ensembled text similarity metrics that better reflect semantic similarity and the Unified Similarity Metric Benchmark, a standardized suite of tasks and datasets meant to asses text similarity measures on their ability to reflect a human understanding of semantic meaning. Afterwards, we introduce a new benchmarking task for LLMs we call "semantic elasticity", and a corresponding dataset and set of evaluations for this task. Finally, we conclude with a discussion of future work and broader implications.

By advancing the robust and aligned measurement of semantic similarity, this thesis aims to enhance the dependability and practical utility of LLMs in diverse, dynamic environments, ensuring they not only function effectively but also responsibly.

Chapter 2

Motivation

Text similarity measurements, especially within the context of LLM applications, are a critical component of production-level machine learning systems. The use cases hinging on a proper understanding of text similarity range from consumer uses such as text summarization and content creation to industry use cases spanning legal documentation, patent similarity, and scientific writing. While modern research has typically focused on developing domain-specific models for these tasks, the text similarity metric used in these tasks is often overlooked and untested. For example, cosine similarity is the go-to in information retrieval tasks [71] despite other metrics being shown to perform better [53] depending on the context of the problem.

Historically, model-based methods that lend themselves to using cosine similarity have used a sentence transformer [47] or word embedding architecture [2] to gauge text similarity. There is a mismatch between the training pipelines behind these models and the document type and length they are commonly paired with. Text embedding models are typically trained on snippets ranging from short sentences to singular paragraphs [36], leading to their outputs falling short when comparing longer pieces of text such as essays or formal documents. This discrepancy between model capabilities and application requirements highlights a crucial gap in our available text similarity metrics. Use cases of transformer-based models in the context of longer documents have applications in information retrieval, model jailbreaking [62], the spread of misinformation [41], data poisoning [74], plagiarism/watermarking detection [64], retrieval-augmented generation [14], and semantic document parsing [56].

With all these use cases where measuring semantic similarity is paramount, there is a sore lack of exploration into methods that effectively and robustly encompass this broad human concept, and a lack of data to train models themselves towards this objective. This leads us to the core of our investigation, focused around the following central research questions:

1. Is cosine similarity the most appropriate metric for measuring similarity between multi-paragraph documents?

2. To what extent does cosine similarity align with human preferences and perceptions of semantic similarity?

3. How do different models perform in tasks that involve rewriting content while preserving its original meaning?

Our motivation is to bridge the divide between current text similarity metrics and the ever-evolving requirements of LLM applications, providing a foundation for future research and practical advancements in these important use cases. This thesis will answer the questions above and pave the way for further exploration into our understanding of text similarity and its desired qualities in reflecting a human perception of language.

Chapter 3

Background

3.1 Machine-Based Text Similarity Measurements

For humans, judging the similarity between two pieces of text can be framed as a grade-school-level task, a skill that comes naturally when learning to understand and analyze text. This type of overarching similarity, encompassing tone, writing style, meaning, and factual accuracy, is called *semantic similarity* [7] and is the core focus of this thesis.

While determining a high level of similarity is fairly straightforward for humans, quantifying this is fraught with variance. Determining an overall measure of text similarity can be incredibly subjective, and can even lead to contradictory results [8] when done by humans. In addition to these fundamental issues, human judgment isn't scalable compared to machine-generated scores due to its slow, scarce, and expensive nature. A human-powered solution isn't possible for corpora consisting of tens of thousands of documents, an extremely pressing issue in an age where the entire internet is being indexed for use with LLMs [60].

How do we frame text similarity as an algorithmic or machine-powered task? A sensible first approach would be to start with character-level or word-similarity metrics. One important character-level metric we will use is Levenshtein distance, measured as the edit (insertion, deletion, substitution) distance between two strings. We can frame this as a ratio between 0 and 1 with the following formula:

$$\text{Levenshtein Ratio}(str_1, str_2) = \frac{len(str_1) + len(str_2) - \text{Levenshtein Distance}(str_1, str_2)}{len(str_1) + len(str_2)}$$

Levenshtein distance and ratio are particularly effective at identifying misspellings and other surface-level differences between strings but fail to capture semantic similarity due to their granular nature. Similarly, we can measure similarity through the frequency and occurrence of words in the text. One notable metric that does so is the Jaccard similarity coefficient [19], calculated as follows:

$$words_1 = set(str_1)$$

$$words_2 = set(str_2)$$

$$\text{Jaccard Similarity}(str_1, str_2) = \frac{|words_1 \cap words_2|}{|words_1 \cup words_2|}$$

Here, we calculate $words_1$ and $words_2$ as the set of all words in str_1 and str_2, respectively. Next, to calculate the Jaccard Similarity, we divide the size of the intersection of these two sets by the size of their union. This roughly translates to comparing two documents by the relative overlap of their respective set of words. Jaccard similarity is most useful in contexts where the absence of certain words is more significant than their frequency or order, such as with keyword-based searches or document classification tasks.

BM25 [50] is another advanced text similarity metric that builds on top of TF-IDF (Term Frequency-Inverse Document Frequency) [22] to rank documents based on a given query. The original BM25, Okapi BM25, modifies TF-IDF by incorporating document length and the frequency of terms in an external query [32]. BM25 is widely used for a wide range of ad-hoc retrieval tasks [49, 51], where the query and the collection of documents are not fixed in advance, providing a robust and flexible measure of text similarity in dynamic information environments. BM25 can be parametrized for a document d and a query q with t representing a term in the query, $|d|$ as the length of the document d, avgdl as the average document length in the corpus, and b and k_1 as hyperparameters. In this thesis, we will use BM25+, which adds a small constant δ to lower bound our scoring.

$$\text{TF}(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

$$\text{IDF}(t, D) = \log\left(\frac{\text{Total number of documents } D}{\text{Number of documents with term } t \text{ in it}}\right)$$

$$\text{BM25}(d, q, D) = \sum_{t \in q} \text{IDF}(t, D) \times \frac{\text{TF}(t, d) \times (k_1 + 1)}{\text{TF}(t, d) + k_1 \times \left(1 - b + b \times \frac{|d|}{\text{avgdl}}\right)} + \delta$$

Finally, we will introduce ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score [29], particularly useful for evaluating summarization tasks. ROUGE is primarily used to assess the quality of summaries by comparing them to one or more reference summaries. ROUGE measures the overlap of n-grams and word sequences between the target text and the reference to quantify how much of the core content and key phrases are captured in its summary. ROUGE focuses on recall, providing a distinct perspective that complements precision-focused metrics like TF-IDF and BM25.

$$\text{ROUGE}_n(str_1, str_2) = \frac{\sum_{\text{gram}_n \in str_1} \text{Count}_{str_2}(\text{gram}_n)}{\sum_{\text{gram}_n \in str_1} \text{Count}(\text{gram}_n)}$$

As we've seen, there are many algorithmic ways to measure text similarity, each with its strengths and weaknesses. There are many similarity metrics we haven't covered for brevity but are still important within their contexts, with some notable ones being BLUE [44], BERTscore [67], and MAUVE score [45]. The correct choice depends heavily on the context in which text similarity is assessed and used for downstream tasks, making this decision a nuanced yet highly impactful consideration.

3.2 Text Embedding Models

To move past word or character-level similarity measures, we need some way to "learn" the representation of text through a machine learning model. To do so, we use text embedding models, which take a string of arbitrary length and transform it into a fixed-length vector. This process is often referred to as "embedding" a piece of text. Early text embedding models such as Word2Vec [33] transform singular words into vectors, which we can then take the average of across the length dimension of our output to achieve a length D vector. Later models such as BERT [12] pad the input text to be a fixed length and use the vector representation of a special $[CLS]$ token to model the input text.

Before passing a piece of text to an embedding model, we need to "tokenize" it. Tokenization refers to the process of substituting a batch of characters in the input string with integers that correspond one-to-one with the substituted characters. For example, a character-level tokenization would convert the letter "a" to 1, "b" to 2, and so on until the input string is an array of integers. In this scheme, the word "dog" would become the array $[4, 15, 7]$ before being passed into our text embedding model. Today's LLMs use a tokenization scheme called Byte Pair Encoding (BPE) [55], which balances character-level and word-level tokenizations in its final mapping. Most models incorporate a BPE scheme with a vocabulary size of 50,000, meaning that their internal mapping incorporates 50,000 unique sets of characters capable of transforming an arbitrary string into an array of integers.

Contrastive loss is commonly used to train text embedding models [48], specifically Noise Contrastive Estimation (NCE) loss [34]. NCE loss simplifies the model training process into a binary classification task. This is achieved by distinguishing a target text from a set of randomly selected 'noise' texts. During each training step, a target text is paired with its true context words and several irrelevant randomly chosen texts from the corpus. The model then learns to increase the similarity between the target text and its true context text (positive examples) while decreasing the similarity with the irrelevant texts (negative examples).

The output of text embedding models is often interpreted as capturing the semantic meaning of the text passed into them [54] in a high-dimensional latent space. This space, geometrically crafted throughout the model training process, has been shown to arrange texts with similar meanings close together, allowing nuanced relationships between words to be discerned based on their proximity and orientation in this space. For instance, synonyms are typically embedded near each other, while antonyms might be positioned on opposite ends along a particular dimension. This spatial arrangement helps facilitate various Natural Language Processing (NLP) tasks such as synonym detection [1], sentiment analysis [5], and thematic grouping [52], as it provides a quantitative method for analyzing semantic similarities and differences within text data.

3.3 Cosine Similarity and Common Uses of Embedding Models

Now that we know **how** text is embedded, we'll transition to discussing **why** these embedding vectors are important. As we touched upon, most uses of text embeddings hinge on their ability to represent meaning in a high-level geometric space. Thus, we can utilize the implications of this assumption to use metrics that measure the distance between vectors to ”measure” the ”distance” between texts. This translates to assigning a quantitative similarity measure between any two documents in the contexts we will discuss. We can measure the distance between two vectors x and y through their dot product.

$$dot(x, y) = \sum_{i}^{n} x_i * y_i$$

As an alternative, we can use Euclidean distance.

$$dist(x, y) = \sqrt{\sum_{i}^{n} (x_i - y_i)^2}$$

The problem with these measures is that they break down in high dimensions [4], making it hard to use them as consistent measures across various embedding models with distinct output dimensionalities. Thus, we often use cosine similarity, which is bounded between 0 and 1 regardless of the vector length [70].

$$cosine(x, y) = \frac{dot(x, y)}{||x||_2||y||_2}$$

How do researchers and engineers use these measures in practice? One of the most common use cases is information retrieval, also known as ”Retrieval-Augmented Generation” (RAG) [27] in the context of LLMs. LLMs have been shown to benefit from ”in-context learning” [63], where providing complete question-answer pairs to the LLM before giving it a new question improves its performance. To accomplish this, we typically have an external database of potentially related documents, which have been embedded into vectors. To decide which documents to retrieve, we usually calculate the cosine similarity between the prompt and every document in our vector database, selecting the top-n documents with the highest similarity scores [25].

Classification is another application of applying text similarity measures to text embeddings. These tasks generally require categorizing documents into pre-specified buckets such as genre, author, academic field, or some other distinction. Some prevalent use cases of clustering are plagiarism detection [21] and LLM watermarking [18]. Classification is also useful for tasks that demand consistency, such as maintaining the writing style, tone, or meaning of a provided text during a rewrite, a common consumer use case of LLMs [13].

3.4 Usages of Text Similarity Measures in Benchmarking

With the popularity and numerous use cases for embedding models, research into improving these models and the vector representation of text in general has skyrocketed [69]. Along with this explosion in model development has come the problem of determining which model performs best for a specified use case, inspiring the creation of leaderboards to measure model performance in standardized manners. A popular example is the Massive Text Embedding Benchmark (MTEB) [35] from Hugging Face, which utilizes 56 datasets across 8 tasks to assign a final score to an embedding model. This leaderboard builds off of previous, more specialized benchmarks, such as the Benchmark for Zero-shot Evaluation of Information Retrieval Models (BEIR) [58] and Unsupervised Sentence Embedding Benchmark (USEB) [61] which encompass a smaller range of tasks. While these benchmarks are useful for model selection, they often use cosine similarity to determine the final score of each embedding model on a large subset of tasks, without analyzing whether or not this is the best metric to judge performance by.

While most benchmarks test models under conditions consisting of clean data and controlled selections, the robustness of a text embedding model and similarity measurement metric are crucial considerations in the real-world usage of these technologies. One weakness of popular benchmarks is not necessarily representing a balanced cross-section of real-world data [42], often skewing towards particular genres or domains, such as news articles over conversational text. These conditions suggest that today's benchmarks may not be the best suited for model selection in all contexts, particularly with many popular models confounding their results on these public benchmarks through overfitting [65] and data leakage [66]. These factors highlight the need for benchmarks that test a model's ability to handle a wide range of linguistic features and contexts, and also examine how well these models perform under varied and adversarial conditions.

Chapter 4

Evaluating Text Similarity Measurements on Robustness and Alignment

We will start by defining two desirable qualities of text similarity metrics: robustness and alignment.

Robustness refers to a metric's resilience against irrelevant or non-semantic modifications in text, such as random capitalization, deletions, or misspellings. A robust similarity metric consistently identifies texts that convey the same meaning, despite superficial changes, and clearly distinguishes texts that differ significantly in content, intent, or meaning.

Alignment, on the other hand, describes a metric's ability to reflect human judgments and preferences. A similarity metric is considered aligned if it reflects human evaluations regarding text similarity. This is especially important in applications like summarization and content moderation, where the end-user's perception and understanding significantly impact the effectiveness of the LLM.

By examining these qualities, we aim to shed light on the capabilities and limitations of existing text similarity metrics and propose three new text similarity metrics. These include two methods we labeled as linear similarity and token similarity metrics. Our last metric is an ensembled measurement metric, which leverages the strengths of multiple metrics to achieve superior performance.

All cosine similarity results are attained by averaging the subtask scores achieved by OpenAI's text-embedding-3-small model [37] and BAAI's bge-large-en-v1.5 model [9] to represent both open and closed-source embedding models in our evaluations. We noticed no significant differences when using these two models and thus chose to combine their performances for brevity.

4.1 Alignment with Human Preferences

To gauge the alignment of text similarity metrics with human preferences, we focus on datasets consisting of human-based scoring systems. Specifically, we utilize two datasets from OpenAI [57] consisting of machine-generated summaries for Reddit posts and news articles. The first dataset contains machine-generated summaries evaluated by human crowd workers based on overall effectiveness, cohesion, factual accuracy, and tone. The second dataset involved choosing between two machine-generated summaries for each post, with crowd workers selecting the one they preferred based on overall quality.

We first focus on alignment with human selection. To determine the summary preferred by a metric, we calculate the score between each post and its respective summaries, choosing the one with the higher similarity score as favored by the metric. We then analyze how accurately these choices reflect the choices made by the crowd workers.

Next, we use the correlation between the overall rating and the text similarity score for each summary, article pair to measure a metric's alignment with human scoring. While there is work showing that models tend to favor content written by themselves [43], we believe this isn't a concern due to the disconnect between generative and embedding models in their training methodology and optimization process, even when from the same research group.

Table 4.1: Alignment with Human Choice

Metric	**Accuracy**	**Precision**	**Recall**	**F1**
Cosine Similarity	**0.66**	**0.66**	**0.66**	**0.66**
Levenshtein Ratio	0.60	0.60	0.62	0.61
ROUGE Score	0.57	0.57	0.59	0.58
Jaccard Sinilarity	0.59	0.58	0.60	0.59
BM25 Score	0.57	0.56	0.58	0.57

Table 4.2: Correlation with Human Scoring

Metric	**Overall**	**Accuracy**	**Coverage**	**Coherence**
Cosine Similarity	**0.57**	**0.48**	**0.55**	**0.51**
Levenshtein Ratio	0.43	0.25	0.433	0.15
ROUGE Score	0.45	0.32	0.44	0.18
Jaccard Similarity	0.46	0.34	0.46	0.19
BM25 Score	0.44	0.35	0.43	0.20

Before we review our results, let us define precision, recall, and the F1 score.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision calculates the number of true positive predictions (TP) and false positives (FP) made by our classifier, measuring the exactness of the classification.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall, or sensitivity, assesses the ability to identify all relevant instances by calculating the proportion of true positives (TP) to false negatives (FN).

$$F1\,\text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 Score is the harmonic mean of precision and recall, providing a balanced measure of both.

Among the pre-existing metrics we evaluate in tables 4.1 and 4.2, cosine similarity consistently outperforms all others in both tasks, with a preference alignment score of 0.66 and a correlation score of 0.57. Jaccard similarity, Levenshtein ratio, ROUGE score, and BM25 lag by approximately 10% in the accuracy task and 20% in the correlation task, demonstrating lower but still significant reflections of human preference. These metrics perform similarly, suggesting a fundamental limitation between n-gram-based methods and embedding-based methods [17].

In our research on ensembled methods for measuring text similarity, we develop two novel metrics, labeled “Linear Similarity” and “Token Similarity”.

Linear Similarity derives from an effort to reflect the sentence-level similarity between two longer pieces of text. This concept builds on interpretability research, demonstrating that the geometric interpretation of an embedding vector holds semantic significance [24]. For example, embeddings for “Ferrari” and “Lamborghini” are likely to be positioned near each other, along with embedding vectors for other vehicle brands. Another research line delves into interpreting the embedding of multi-word or multi-concept pieces of text, arguing that combining embeddings can represent new concepts [26]. In this context, the embedding for ”brown dog” would be the sum of the embedding vectors for “brown” and “dog”. We hypothesize that this summation pattern should hold when comparing the sentence embeddings of a document to the embedding of the document itself and that using these sentence embeddings may yield more precise insights due to their granularity.

To test this hypothesis, we run a simple linear regression model to achieve a weighted sum of a document’s sentence embeddings against the document embedding. We find that our resulting prediction had an $R^2 > 0.8$ in the vast majority of cases, suggesting that over 80% of the variance in the document’s embedding vector can be accounted for by combining its sentence embeddings. Although our reconstruction is lossy, this result provides a basis to use a document’s sentence representation to try and measure text similarity.

For our final functional form of Linear Similarity, we draw from insights in the ColBERT model [23] and sum over the maximum similarity of each sentence in the reference text with every sentence in the generated text to formulate our similarity measure. We parameterize the *linear_similarity* function with R and T, where $R \in R^{L_1 \times D}$ and $T \in R^{L_2 \times D}$, representing the number of sentences in our reference and generated (target) texts, respectively, and D represents the dimension of our text embedding model.

$$\text{linear\_similarity}(R, T) = E[\max(R(T^T), \text{dim} = 1)]$$

We will discuss our token similarity metric construction methodology later in this section.

While these metrics perform well on their own, they do not quite match the performance of cosine similarity on our two tasks. However, we can see that many of our metrics are fairly uncorrelated, suggesting that they can be combined to enhance performance [39]. The correlations between each similarity metric's scores on all measurements we made for this task are shown in table 4.3.

Table 4.3: Metric Cross-Correlations

	Cosine	**Levenshtein**	**ROUGE**	**Jaccard**	**BM25**	**Linear**	**Token**
Cosine	1	-	-	-	-	-	-
Levenshtein	0.52	1	-	-	-	-	-
ROUGE	0.52	0.97	1	-	-	-	-
Jaccard	0.54	0.95	0.97	1	-	-	-
BM25	0.45	0.60	0.67	0.69	1	-	-
Linear	0.14	0.22	0.21	0.23	0.05	1	-
Token	0.72	0.59	0.63	0.66	0.69	0.13	1

Borrowing from insights gleaned from ensembling in classical machine learning contexts, we focus on developing an ensembled metric for each specific task. The popularity of ensembled methods such as Random Forest Classifiers or XGBoost relies on the fact that when individual metrics exhibit strong performance but low correlation with each other, they likely capture different aspects or features of the data. By aggregating these diverse, independent metrics into a singular measure, we're able to use the strengths of each to achieve more robust and accurate predictions than any single metric could achieve on its own. This integration can reduce the variance and bias of our ensembled estimator's final outputs.

To leverage this, we implement two statistical models: a Random Forest Classifier for selecting the preferred summary in the binary comparison task, and a Linear Regression model for scoring summary quality, both trained on human scores and preferences. We verify the effectiveness of these methods through standard statistical t-tests, achieving significance levels ¡¡ 0.001 for the Linear Regression Model in 4.5 and 0.01 for our Random Forest Classifier in 4.4 when testing whether these results improved upon the results from using just cosine similarity as a feature.

We can see that while our ensembled method increases performance by $\sim 4.6\%$ overall and generally across the board in mimicking human choice, it is simply competitive with cosine similarity in the human scoring correlation task, yielding a 7.1% improvement to

Table 4.4: New Metric Alignment with Human Choice

Metric	**Accuracy**	**Precision**	**Recall**	**F1**
Cosine Similarity	0.66	0.66	0.66	0.66
Linear Similarity	0.63	0.62	0.66	0.64
Token Similarity	0.59	0.58	0.60	0.59
Ensembled Similarity	**0.70**	**0.70**	**0.70**	**0.70**

Table 4.5: New Metric Correlation with Human Scoring

Metric	**Overall**	**Accuracy**	**Coverage**	**Coherence**
Cosine Similarity	0.57	0.48	0.55	**0.51**
Linear Similarity	0.07	0.11	0.09	0.08
Token Similarity	0.46	0.46	0.45	0.42
Ensembled Similarity	**0.61**	**0.51**	**0.60**	0.46

correlation on overall performance and coverage, likely following from their high pre-existing correlation.

4.2 Robustness to Semantic and Superficial Transformations

Next, we aim to assess text similarity metrics on alterations in a document that either preserve or alter the underlying meaning. Our goal is to determine whether a metric can accurately distinguish between texts that are semantically similar despite superficial changes, and texts that differ fundamentally in meaning despite superficial similarities.

Superficial Transformations:
We categorize superficial transformations into three types:

- Random Capitalization: Randomly capitalizing letters in the text with a given probability.
- Deletion: Removing every n^{th} letter from the text, excluding spaces.
- Numerization: Substituting specific letters with numerals (e.g., replacing 'o' with '0').

Despite these changes, the semantic content of the text remains intact. For example, numerization might transform the phrase "The brown fox jumps over the lazy dog" into "Th3 br0wn f0x jumps 0v3r th3 l4zy d0g". Although these may appear different at first glance, humans would recognize them as the same sentences. A robust text similarity metric should rate such altered texts as highly similar to the original, aligning with human perception of semantic similarity. These transformations test the metric's ability to disregard non-semantic variations, focusing on meaning rather than form.

Semantically Altering Transformations:
Conversely, we explore transformations that significantly alter the meaning of the text:

- Negation: Changing affirmative statements to negative (e.g., changing "is" to "is not").
- Shuffle Sentences: Randomly rearranging the order of sentences.
- Shuffle Words: Scrambling the order of words within the text.

These changes disrupt the original semantic structure. For example, negation reverses the intended meaning, while shuffling sentences or words can render the text nonsensical or disjointed. An example of word shuffling would be transforming the phrase "The brown fox jumps over the lazy dog" into "brown jumps dog The over the fox lazy". Although these strings contain the same words, the latter holds no coherent meaning, and this effect worsens in longer documents. A reliable similarity metric should reflect these substantial changes by assigning low similarity scores to texts with these alterations when compared to the original.

To empirically test the effect of these changes on our similarity measures, we utilize a dataset of PubMed publications [11] separated into their abstracts and main texts, which we will refer to as articles. We compare the articles and abstracts against each other, and also against both superficially and semantically altered versions of themselves. We hypothesize that:

- Texts with superficial changes should exhibit higher similarity scores compared to their original counterparts than texts with semantically altering changes.
- Semantically altered texts should receive lower similarity scores compared to the original text than their counterparts (article to abstract) in the same paper.

To quantify how well a text similarity measure performs on these metrics, we introduce a scoring function, assigning values between 0 and 1 to measure the effectiveness of the measures under our defined ablations.

$$S_1 = E[\mathrm{f}(\mathrm{article}, \mathrm{abstract}) > \mathrm{f}(\mathrm{article}, \mathrm{shuffle\_word}(\mathrm{article}))]$$

$$S_2 = E[\mathrm{f}(\mathrm{article}, \mathrm{deletion}(\mathrm{article})) > \mathrm{f}(\mathrm{article}, \mathrm{abstract})]$$

$$S_3 = E[\mathrm{f}(\mathrm{article}, \mathrm{numerization}(\mathrm{article})) > \mathrm{f}(\mathrm{article}, \mathrm{shuffle\_word}(\mathrm{article}))]$$

$$score = \frac{S_1 + S_2 + S_3}{3}$$

Here, we aim to evaluate whether our text similarity metric, denoted by f, is robust under each condition we have established. We average the scores over each condition to determine our final score.

Our findings, presented in Table 4.6, are quite surprising, particularly with respect to cosine similarity. We discovered that cosine similarity often rates texts with their words completely shuffled—which typically renders the content nonsensical—as more similar to the original than texts with only 25% of letters randomly capitalized. This counterintuitive outcome highlights a significant challenge: traditional similarity metrics may prioritize structural similarity over semantic coherence, failing to adequately penalize the loss of logical narrative that results from word shuffling. This phenomenon could be attributed to the inherent design of these metrics, which do not inherently distinguish between syntactic rearrangement and semantic alteration [16]. Additionally, we find that numerization is the most detrimental transformation to cosine similarity, possibly due to differences in tokenization between numbers and alphabetic characters. This suggests a limitation in cosine similarity's ability to capture the true semantic essence of a text. Conversely, metrics that focus on word or character-level similarities demonstrate greater resilience to superficial changes but struggle to accurately reflect the semantic similarity between full texts and their abstracts.

Table 4.6: Performance on Robustness Tests

Metric	S_1	S_2	S_3	**overall**
Cosine	**0.55**	0.61	0	0.38
Levenshtein	0.02	**1.00**	0.88	0.63
ROUGE	0.01	0.98	0	0.33
Jaccard	0	0.96	0	0.32
BM25	0.47	0.67	0.93	0.69
Token	0	0.79	0	0.26
Ensembled	0.35	0.97	**0.94**	**0.75**

When using a simple linear weighting of similarity metrics to develop an ensembled scoring measurement, we achieve our highest overall score on this task by a large margin. Our ensembled metric beats all others when scoring semantically similar changes over semantically altering changes, largely due to its dependence on Levenshtein similarity and BM25 score. While it doesn't score higher than BM25 and cosine similarity on the task of scoring a semantically similar piece of text over a semantically altering transformation and lags slightly behind Levenshtein and ROUGE score on the last task of ranking superficial changes over semantically altering changes, its dependence on multiple metrics helps it achieve a 20% improvement over the next metric, Levenshtein, in terms of overall score.

4.3 Sensitivity of Measurements to Unrelated Text

Continuing, we examine how the insertion of irrelevant text affects the effectiveness of text similarity metrics. This test is intended to assess the robustness of these methods against unrelated content that does not contribute meaningfully to the document. To facilitate this analysis, we develop a function to insert a 'needle' into a text at a specified location.

$$\text{insertion}(\text{needle}, \text{location}, \text{text}) = \text{text}[: \text{location}] + \text{needle} + \text{text}[\text{location} :]$$

We control the effect of various needles by using Lorem Ipsum text and variants/extensions of the popular needle "The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day." [31]. We hypothesize that an ideal similarity score should decrease linearly as the size of the needle being used increases and that this change should be mostly invariant to the location of the needle.

Surprisingly, our findings, plotted below in figures 4.1 and 4.2, reveal a strong dependency on the location of the needle when using cosine similarity as our text similarity measurement. Specifically, inserting the needle at the start of the text decreased the similarity between the resulting and original text the most, with a slightly lower but still pronounced effect when inserting the needle into the back of the text. This effect is fairly negligible when the needle is inserted into the middle of the text, a placement location with the least variance in similarity scores. These effects persist when changing the needle size, and become even more pronounced as the needle size increases.

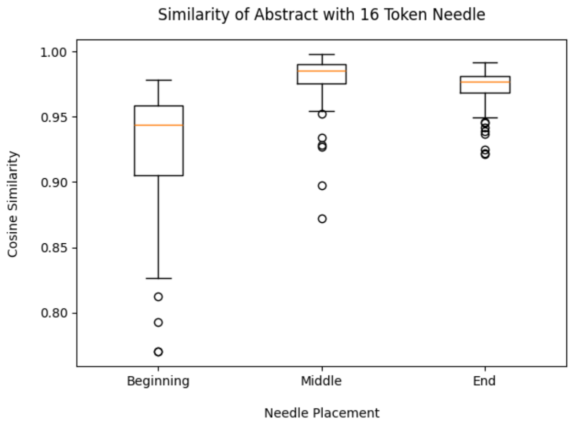

Figure 4.1: Effect of 16 Token Needle

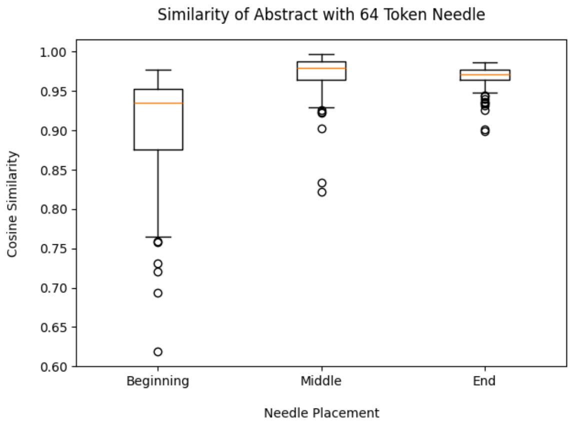

Figure 4.2: Effect of 64 Token Needle

This result suggests that the first and last sentences carry more weight than sentences in the middle when determining the embedding vector for a document. There are several potential explanations for this, such as this being the way humans naturally write, dataset bias, or an architectural effect of embedding models. Isolating the cause and downstream effects of this finding is an interesting direction for future work.

Diving deeper into the effects of the needle on text similarity, we next explore how the cosine similarity changes as a function of the size of the needle as a fraction of the overall text. We find that while the cosine similarity decreases as the size of the needle increases, as expected, it drops at a much lower rate than the proportion of the needle to the original text, with a high dependence on position. In figures 4.3 and 4.4, we can see that even when the needle is well over the length of the original text and accounts for over 70% of the entirety of the new text, the cosine similarity is still between 0.5 and 0.85, indicating a surprisingly high degree of similarity. The dropoff in cosine similarity high depends on positioning, with the figure in 4.3 decreasing much more evenly than in 4.4, where even with 50% of the text as a needle the average similarity score is above 0.9.

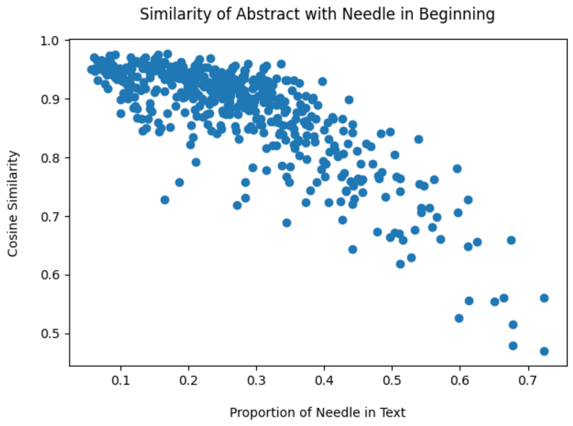

Figure 4.3: Needle insertion at Beginning

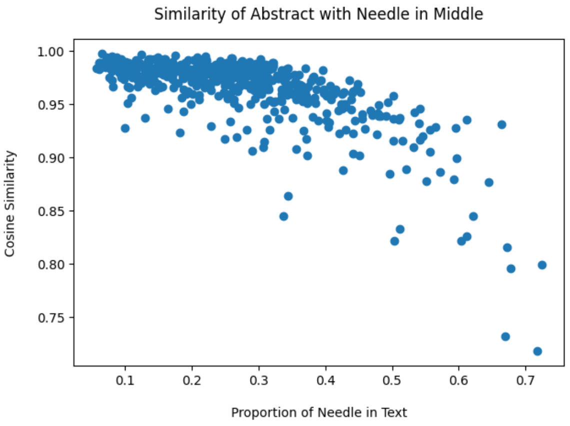

Figure 4.4: Needle insertion at Middle

When a user is trying to retrieve documents where relevant information is surrounded by noise, this characteristic of cosine similarity can be highly desirable. On the flip side, when we're trying to achieve a pure understanding of the semantic similarity between two documents, this result can be worrying. This result does not dissuade the use of cosine similarity in document retrieval or text understanding but instead characterizes that the context of its usage has an outsized effect on its efficacy as a metric.

Lastly, it's interesting to note the relation between our observations and the "Lost in the Middle" effect [30] observed in needle-in-the-haystack tests conducted on decoder models. Here, information at the beginning and end of a model's context has been shown to receive more weighting and is more heavily attended to than content in the middle of the provided context [28]. Further research into this effect could yield a parallel approach to popular "re-ranking" models, often used to re-order pieces of context after retrieval [15] to place the most important context at the beginning of the prompt.

4.4 Investigating Token Frequency's Relationship with Cosine Similarity

While investigating the effect of superficial and semantically altering changes on the measurement of cosine similarity between texts, we noticed an interesting trend. Shuffling the sentences or words of the original text consistently yielded surprisingly high similarity scores, despite the significant loss of semantic meaning. These transformations consistently resulted in higher similarity scores compared to those obtained when comparing an article to its abstract or vice versa. This was also true when applying superficial transformations such as randomly capitalizing portions of the text or substituting some letters with numbers. The impact of each transformation on cosine similarity score is detailed in Tables 4.7 and 4.8.

Table 4.7: Similarity with Original Article

	Similarity
Article	1
Cleaned Article	0.97
Sentences Shuffled	0.93
Random Deletion (10%)	0.85
Words Shuffled	0.83
All Caps	0.83
Abstract	0.82
Random Capitalization	0.73
Random Deletion (20%)	0.67
Numerization	0.34

Table 4.8: Similarity with Original Abstract

	Similarity
Abstract	1
Cleaned Abstract	0.97
Sentences Shuffled	0.94
All Caps	0.89
Random Deletion (10%)	0.85
Words Shuffled	0.85
Article	0.82
Random Capitalization	0.82
Random Deletion (20%)	0.66
Numerization	0.35

These results led us to hypothesize that cosine similarity was correlated with some functional representation of token frequency, as the shuffling sentences and shuffling words wouldn't heavily alter the input tokens distribution, but numerization and capitalization would. To test this hypothesis, we constructed a *token_similarity* metric that captured the frequency of each term in the reference and target texts.

$$terms = \cup(set(tokens(text_1)), set(tokens(text_2)))$$

$$freq_i = \frac{token\_freq(text_i)}{1 + log(1 + token\_freq(text_i))}$$

$$\text{token\_similarity}(freq_1, freq_2) = dot(\frac{freq_1}{||freq_1||_2}, \frac{freq_2}{||freq_2||_2})$$

Here, *token_freq* buckets the frequency of each token in $text_i$ into an index of a vector with length equivalent to the number of unique tokens in all texts. Each index of this vector represents a specific token. After constructing these vectors for both texts, we take their normalized dot product to achieve our final score.

We took two approaches to determine how effectively our new metric represented cosine similarity. One was observing how well the trends in using this metric reflected the trends shown by cosine similarity, and the other was directly calculating its correlation with cosine similarity under various transformations, as shown in table 4.9.

Table 4.9: Correlation of Token Similarity with Cosine Similarity

	Article	**Cleaned**	**Shuffled**	**Abstract (Abs)**	**Cleaned$_{\mathbf{Abs}}$**	**Shuffled$_{\mathbf{Abs}}$**
Article	1	-	-	-	-	-
Cleaned	0.66	1	-	-	-	-
Shuffled	0.23	0.25	1	-	-	-
Abstract	079	0.80	0.76	1	-	-
Cleaned$_{\mathbf{Abs}}$	0.82	0.82	0.80	0.48	1	-
Shuffled$_{\mathbf{Abs}}$	0.81	0.80	0.82	0.47	0.56	1

In most scenarios, our token similarity metric correlates highly with cosine similarity. On the upper end, we see correlations of 0.8 between the similarity measures comparing our abstract and article, suggesting that we can represent a large fraction of cosine similarity in an untransformed setting. While this correlation goes lower for certain transforms, only being 0.21 when comparing scores for an article under a sentence shuffling transformation, the high correlation for untransformed settings suggests a potentially inexpensive way of realizing the benefits of cosine similarity with having to use a larger model. It's clear, however, that cosine similarity captures some semantic meaning in its output that can't be represented by a permutation invariant function of the text's tokens.

4.5 Introducing the Unified semantic Similarity Metric Benchmark

Now that we've thoroughly investigated the strengths and weaknesses of multiple text similarity measures, with a focus on cosine similarity, it is time to measure performance on each of our subtasks to determine an overall leaderboard quantitatively combining robustness and alignment. We propose the Unified semantic Similarity Metric Benchmark (USMB), a suite of tests, tasks, and datasets to measure the ability of text similarity metrics to adhere to a human understanding of semantic similarity and meaning. The USMB leaderboard is composed of four tasks: preference alignment, semantic transformation robustness, information sensitivity, and clustering/classification. We assign a numerical score bounded between 0 and 1 for each metric's performance in each category and compute an overall score as the mean of the category scores.

For the human preference task, we take the average of a metric's performance on the scoring and preference tasks. For the robustness task, we use the scoring system detailed earlier in its respective section. For the sensitivity task, we assume an ideal behavior as an inverse 1:1 relationship with the size of the needle as a percentage of the overall text, invariant to needle position. Thus, we calculate the overall error as the Mean Absolute Error (MAE) between a metric's score and the ideal behavior, bounded between 0 and 1 due to all predictors being within this range. To transform this into a measure where higher is better, we subtract this bounded error from 1.

Lastly, we average accuracy across all constituent datasets for the clustering tasks, sourcing our datasets and scoring methodology from the MTEB benchmark [35].

Table 4.10: Unified semantic Similarity Metric Benchmark

Metric	**Human Pref.**	**Robustness**	**Sensitivity**	**Clustering**	**Overall**
Cosine	0.612	0.385	0.200	0.420	0.404
Levenshtein	0.515	0.632	0.134	0.177	0.365
Jaccard	0.525	0.320	0.064	0.187	0.274
ROUGE	0.510	0.330	0.124	0.182	0.287
BM25	0.501	0.691	0.314	0.187	0.423
Ensembled	**0.657**	**0.753**	**0.511**	**0.585**	**0.627**

According to the constructed USMB in table 4.10, our task-specific ensembled methods perform better than a stand-alone metric across the board, with an overall score of 0.627 being 48.2% higher than the second-best metric, BM25. Of the individual metrics, cosine similarity and BM25 similarity score the highest, corroborating their stance as the most popular measures for text similarity and retrieval tasks but highlighting the potential for ensembled methods to increase performance in their respective use cases. While Levenshtein ratio, Jaccard Similarity, and ROUGE score lag on overall scoring, they contribute their strengths and weaknesses to form a much stronger ensembled metric, hinting that these measures should not be overlooked in today's text similarity research and measurement.

We see the biggest gain from Ensembling in the robustness and sensitivity subtasks, where measuring semantic similarity is both the most subjective and the most difficult to discern for automated text similarity metrics. Clustering also sees a 39.2% gain from ensembling over cosine similarity, which already scores over double that of the next best metric's score. Our findings in the clustering subtask underscore the effectiveness of cosine similarity in text similarity compared to other traditionally used methods but continue the theme of simple ensembling performing significantly better.

4.6 Ensembling Performance as a Measure of Training Data

Given that our ensembled methods reflected human preference more closely than a standalone method after fitting a task-specific statistical model, we wanted to measure the effect of the number of data points used to fit these models with their resulting performance. To test this, we varied the size of our training set from 10% to 90% of our dataset and measured performance on the resulting test set, with results shown in figures 4.5 and 4.6. To ensure fairness, we ran the same test with the same models but used only cosine similarity scores as features. We then fit each model 1000 times at each level of train-test split, each time with a different random seed, and averaged the results. We then used a standard t-test to verify the statistical significance of the results, which follows from the theoretically and empirically normal distribution of scores in each category. Here, our null hypothesis was the mean of cosine similarity-based scores.

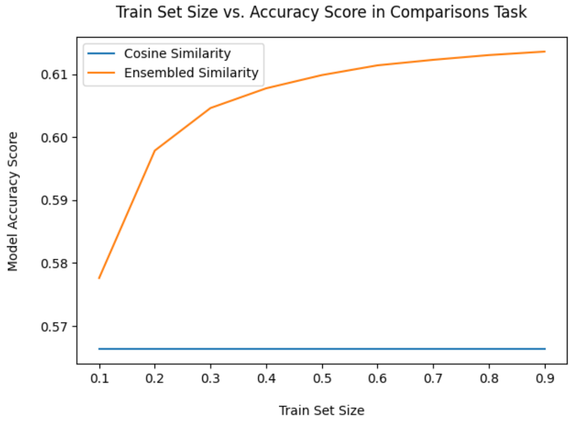

Figure 4.5: Scoring Task Performance

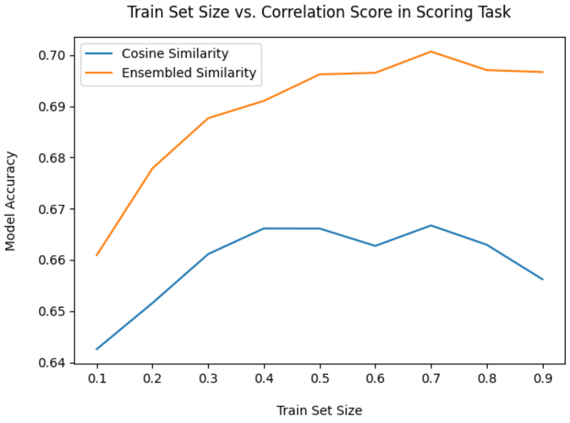

Figure 4.6: Preference Task Performance

As expected, our model performance increases with the size of the training set. There's no clear saturation point in the preference task, and a saturation point of 0.7 in the scoring task, making it difficult to pinpoint an ideal split size. On the other hand, cosine similarity stays constant in the preference task and saturates at 0.3 in the scoring task, suggesting that the data's cosine representation is fairly homogeneous [46].

Balancing the dual considerations of performance and the difficulty of labeling data, we recommend using 40% of the data to fit an ensembled text similarity metric. This improves over existing model-based approaches [10], where a typical training set is at least 60% of the original dataset and is much more expensive and time-consuming to fit compared to our regression and tree-based models.

Chapter 5

Ensembled Metric Performance on Information Retrieval Tasks

Before moving on, it's worth exploring the practical ramifications of our findings from the USMB, namely in testing the performance of our proposed text similarity metrics in a real-world scenario. The most applicable use case is in Information Retrieval (IR). IR benchmarks measure the ability of a model or metric to retrieve the most relevant documents to a given query, with applications in search, synthesis, and Retrieval-Augmented Generation (RAG) [73].

To measure the performance of our text metrics on IR tasks, we borrow our datasets and scoring methodology from BIER [58], an IR benchmark with 18 diverse datasets representing varied tasks and domains. Again, we use both OpenAI's text-embedding-3-small model [37] and BAAI's bge-large-en-v1.5 model [9] and average the final results for the two models due to their similar performance.
We mimicked a typical IR pipeline, which consists of the following steps:

1. Embed all documents into vectors and add them to a vector database.

2. Embed the query (search term) into a vector.

3. Return the top-n documents with the highest similarity scores with the query.

After retrieving our top-n documents, we determined how many matched our provided ”ideal” documents to determine the metric's score as a percentage of the highest possible score as shown in figure 5.1. While our Ensembled metric performed well, pure cosine similarity achieved a score that was 2.6% higher. Levenshtein, which performed well on our robustness tasks, fails significantly here due to the distribution mismatch between the query and the documents. In general, Levenshtein, ROUGE, and Jaccard are meant to compare text to text and not query to documents [68], indicating a fundamental divide in their ability to perform well in IR tasks.

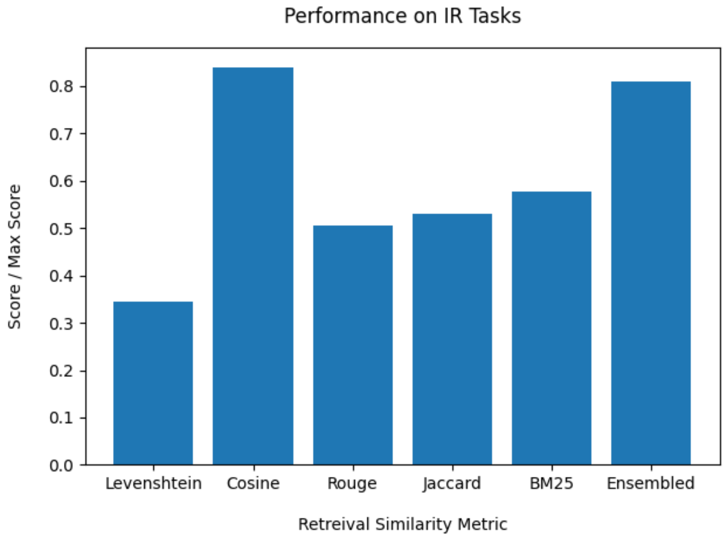

Figure 5.1: Metric Performance on IR Tasks

While the IR tasks in BIER are meant to mimic real-world data, the datasets are constructed from high-quality documents, such as Wikipedia articles, news articles, and PubMed papers. Thus, there is a potential distribution mismatch between our evaluative measures and machine learning systems that handle large-scale data that can often be nonsensical, riddled with typos, or poisoned [6]. We want to use this information to construct new datasets to measure our metrics abilities to stay aligned and robust with human preferences in unclean and noisy environments mimicking real-world machine learning systems.

To accomplish this, we create 6 copies of each original dataset in the BIER benchmark and apply the following transformations: shuffling the sentences, shuffling the words, random capitalization, random deletion, adversarial needle insertion, and original (no transformation). We then concatenate these 6 transformations to create a new dataset, 6 times the length of the original, and repeat for all 18 datasets in the original benchmark. Since the original documents are still in this "augmented" dataset, our text similarity metrics can score similarly to their previous performance on the original dataset. Since random capitalization and random deletion are superficial changes, we de-duplicate them from our scoring if our metric retrieves the same document multiple times under these transformations. To quantify the ability of our metrics to remain robust to these changes, we measure their performance retention as a percentage of their original scores pre-augmentation as shown in 5.2.

Now we see a similar trend where cosine similarity and our ensembled metrics perform best but with the flipped result of ensembling performing 4.4% better. Levenshtein, Rouge, Jaccard, and BM25 still score significantly lower, demonstrating their inability to achieve high initial retrieval scores, and their lack of robustness to data augmentation in IR tasks.

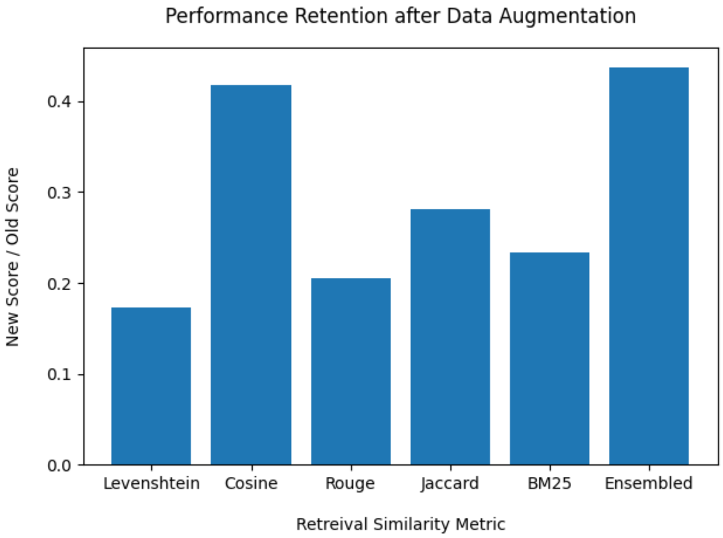

Figure 5.2: Performance Retention after Data Augmentation

These metrics are still valuable, however, as they contribute to the improved performance of our ensembled metric in overall robustness.

While our methodology aims to reflect the realities of unclean environments, our pipeline can still be significantly improved. For example, we would like to mimic the distribution of typos, poor structure, and poisoned data in real-world systems [72]. Without datasets to draw these numbers from, however, an accurate reflection of real-world data may be impossible to attain. Another potential improvement would be the exploration of different IR pipelines, such as document chunking or sampling as a replacement for top-n selection.

Chapter 6

Benchmarking Model Capabilities with Semantic Elasticity

To conclude the contributions of this thesis, we introduce a benchmark and dataset to compare LLMs in their ability to maintain the semantic meaning of a reference text under a rewriting operation. We evaluate our criteria on popular closed and open-source models, including LLaMa 3 [59], Claude 3 Haiku [3], GPT-4 [38], GPT 3.5 Turbo [40], and Mixtral [20].

Specifically, our benchmark supersets summarization tasks to include information compression and expansion. Each model was tasked with rewriting a reference text to be 50% shorter, 25% shorter, 25% longer, 50% longer, 100% longer while maintaining the original tone, style, and meaning. Our dataset contains over 300 long-form essays/data points from thought leaders Paul Graham and Sam Altman, chosen for their rich language and complex ideas, which we hypothesize would pose significant challenges for models in maintaining semantic integrity.

To assess the ability of each model to adhere to our specified rewriting criteria, we employ a multifaceted scoring system that combines cosine similarity, Levenshtein ratio, ROUGE score, Jaccard similarity, BM25 score, and a new metric we call word count score. The word count score measures how well the model achieved the target percentage change in the reference text's word count.

$$expected = \text{word\_count}(\text{reference})$$

$$actual = \text{word\_count}(\text{target})$$

$$\text{word\_count\_score}(expected, actual) = 1 - \frac{|expected - actual|}{expected + actual}$$

This formula measures the relative error between the expected and actual word counts, normalized to a scale of 0 to 1, where a 1 represents perfect adherence to the target word count and a 0 represents maximum deviation. This metric is particularly insightful as it quantifies the precision with which models can control output length given the task requirement, a critical aspect of semantic elasticity.

With all of our constituent scores being bounded between 0 and 1, we calculate the overall score for each model as the average of its performance over each metric considered.

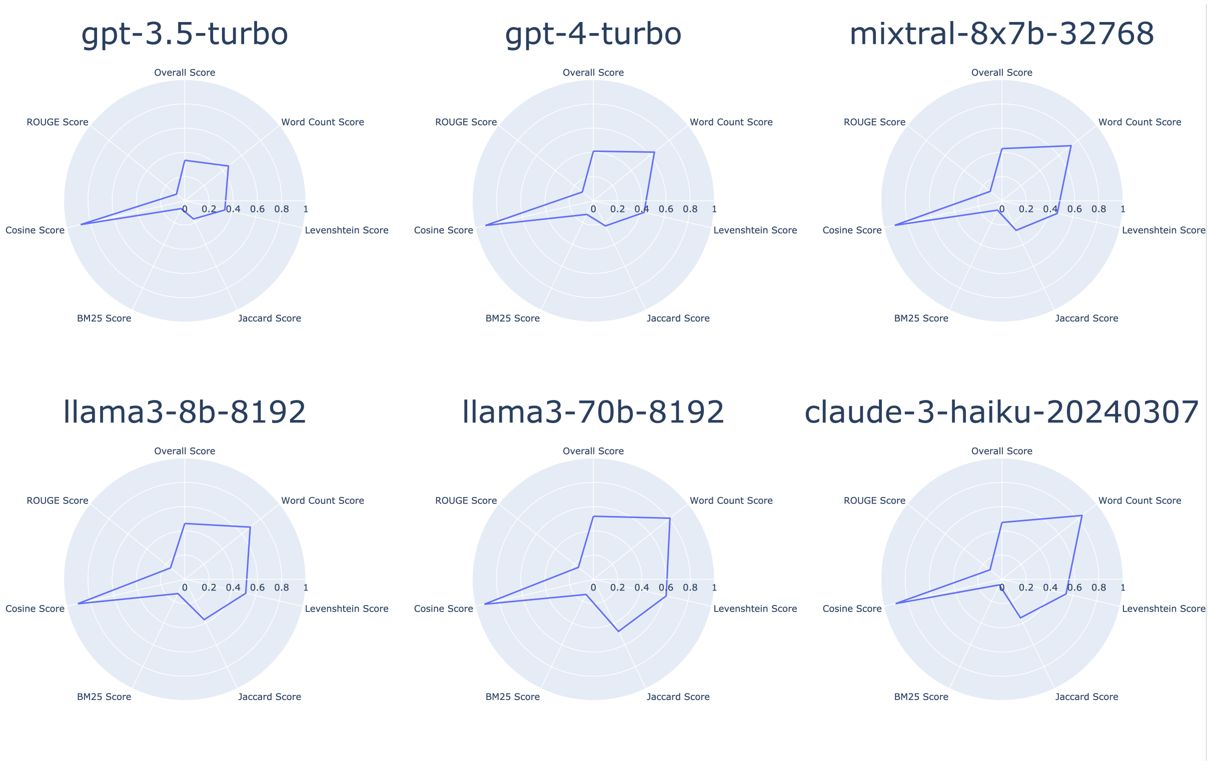

Figure 6.1: Subtask and Overall Scores on Semantic Elasticity Task

Our findings, shown in figure 6.1, reveal meaningful differences in model capabilities based on our scoring methodology. From a high-level overall score perspective, LLaMa 3 70B and Claude 3 Haiku perform the best, closely followed by LLaMa 3 8B and Mixtral. Surprisingly, GPT-4 and GPT 3.5 Turbo have the worst scores, with another clear gap between the two. All the models perform well in cosine similarity and are equally weak when measuring the BM25 and ROUGE similarity. The Jaccard and Levenshtein scores exhibit similar patterns and variance as the overall score. The greatest factor to overall variance comes from the word count score, with some models such as Claude 3 performing close to perfect and the GPT family notably struggling on this metric.

The specific struggles of the GPT family highlight potential limitations in their training regimes or inherent architectural biases that may not prioritize precise length control as effectively as models like Claude 3. This likely stems from innovations since their release and the year of work done between the release of all other models and GPT-4.

Chapter 7

Future Steps

Follow-up work to this thesis can explore many distinct and equally important directions. Augmenting my findings can take the form of developing more sophisticated and effective text similarity metrics, expanding the USMB framework with a greater number of datasets and challenging tasks, and exploring new architectural modifications to LLMs to enhance their performance in semantic elasticity. Equally impactful would be assessing a broader range of models from classical machine learning on their ability to best combine existing text similarity metrics into even stronger ensembled measures. One last major follow-up would entail training a novel embedding model to perform well on the tasks we laid out, further increasing potential performance gains. The innovation here would mainly lie in the training methodology, where we could include our superficial and semantically altering transformations from both the robustness and sensitivity tasks in the model's training data. By comparing and contrasting these transformations with the original text, there is high potential to imbue a stronger sense of semantic similarity into the model's outputs.

As an additional branch, I hope to delve deeper into the interpretation and impact of text position within a document's final embedding vector. While this thesis explored the effect of placing a variable-sized needle into various document positions, a more robust set of tests must be created to understand our findings. This can take the form of further analysis of the effects of sentence removal, shuffling, and addition on the document's embedding vector. Lastly, we can use the ideas behind our linear similarity metric to quantify each document component's effect on the overall vector representation.

Finally, the practical ramifications of this thesis warrant further investigation. Our tests and findings have implications in information retrieval, data poisoning, model jailbreaking, and beyond. Future steps should involve utilizing well-known benchmarks in these fields to evaluate the effects of unclean or poisoned data on information retrieval tasks, the most significant application of our findings to industrial-scale databases and machine learning systems. Additionally, our sensitivity results can be applied to assess an attacker's ability to corrupt an information retrieval database with misleading and undesirable inputs. Here, we can explore the ensuing consequences for downstream system usage.

Chapter 8

Conclusion

In this thesis, we explored and decomposed semantic similarity, a uniquely human concept, and the ability of existing text similarity metrics to measure this concept. We discovered that every metric exhibits differing strengths and weaknesses depending on which aspect of semantic similarity it's measured on. Using these findings, we developed task-specific ensembled methods that perform better than a stand-alone metric in every task we set out, demonstrating the potential to combine classical statistical modeling methods with modern neural embedding models. We proposed the Unified semantic Similarity Measure Benchmark (USMB) to rank text similarity metrics on all our proposed measures. Lastly, we introduced semantic elasticity, a new task encompassing information compression and expansion, quantifying the ability of existing models on this challenging task.

I hope this thesis catalyzes a greater examination of the text similarity methods used to perform various tasks involving LLMs. Research has focused heavily on model development, overlooking performance gains from developing novel semantic similarity measurements. In addition, I hope that the variety of failures induced by the use of cosine similarity in non-standard environments puts greater scrutiny on the ubiquitous popularity of this measure in critical tasks encompassing the vast majority of LLMs in industry settings.

Bibliography

[1] Muhammad Asif Ali et al. *Antonym-Synonym Classification Based on New Sub-space Embeddings*. 2019. arXiv: 1906.05612 [cs.CL].

[2] Felipe Almeida and Geraldo Xexéo. *Word Embeddings: A Survey*. 2023. arXiv: 1901.09069 [cs.CL].

[3] Anthropic. *The Claude 3 Model Family: Opus, Sonnet, Haiku*. 2024. URL: https://api.semanticscholar.org/CorpusID:268232499 (visited on 05/08/2024).

[4] Vitaly Bulgakov and Alec Segal. *Dimensionality Reduction in Sentence Transformer Vector Databases with Fast Fourier Transform*. 2024. arXiv: 2404.06278 [cs.DB].

[5] Erion Çano and Maurizio Morisio. "Quality of Word Embeddings on Sentiment Analysis Tasks". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2017, pp. 332–338. ISBN: 9783319595696. DOI: 10.1007/978-3-319-59569-6_42. URL: http://dx.doi.org/10.1007/978-3-319-59569-6_42.

[6] Nicholas Carlini et al. *Poisoning Web-Scale Training Datasets is Practical*. 2024. arXiv: 2302.10149 [cs.CR].

[7] Dhivya Chandrasekaran and Vijay Mago. "Evolution of Semantic Similarity—A Survey". In: *ACM Computing Surveys* 54.2 (Feb. 2021), pp. 1–37. ISSN: 1557-7341. DOI: 10.1145/3440755. URL: http://dx.doi.org/10.1145/3440755.

[8] Guiming Hardy Chen et al. *Humans or LLMs as the Judge? A Study on Judgement Biases*. 2024. arXiv: 2402.10669 [cs.CL].

[9] Jianlv Chen et al. *BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation*. 2023. arXiv: 2309.07597 [cs.CL].

[10] Liang-Chieh Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *ECCV* (2018). URL: https://link.springer.com/chapter/10.1007/978-3-030-01234-2_49.

[11] Arman Cohan et al. *A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents*. 2018. arXiv: 1804.05685 [cs.CL].

[12] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[13] Ge Gao et al. *Aligning LLM Agents by Learning Latent Preference from User Edits*. 2024. arXiv: 2404.15269 [cs.CL].

[14] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2024. arXiv: 2312.10997 [cs.CL].

[15] Michael Glass et al. *Re2G: Retrieve, Rerank, Generate*. 2022. arXiv: 2207.06300 [cs.CL].

[16] Chaitanya Ekanadham Harald Steck and Nathan Kallus. "Is Cosine-Similarity of Embeddings Really About Similarity?" In: (2024). DOI: 10.48550/arXiv.2403.05440. URL: https://doi.org/10.48550/arXiv.2403.05440.

[17] Felix Hill et al. *Embedding Word Similarity with Neural Machine Translation*. 2015. arXiv: 1412.6448 [cs.CL].

[18] Mingjia Huo et al. *Token-Specific Watermarking with Enhanced Detectability and Semantic Coherence for Large Language Models*. 2024. arXiv: 2402.18059 [cs.LG].

[19] Paul Jaccard. "The Distribution of the Flora in the Alpine Zone". In: *New Phytologist* 11.2 (1912), pp. 37–50. DOI: 10.1111/j.1469-8137.1912.tb05611.x.

[20] Albert Q. Jiang et al. *Mixtral of Experts*. 2024. arXiv: 2401.04088.

[21] MAC Jiffriya, MAC Akmal Jahan, and Roshan G. Ragel. *Plagiarism Detection on Electronic Text based Assignments using Vector Space Model (ICIAfS14)*. 2014. arXiv: 1412.7782 [cs.IR].

[22] Karen Spärck Jones. "A Statistical Interpretation of Term Specificity and Its Application in Retrieval". In: *Journal of Documentation* 28.1 (1972), pp. 11–21. DOI: 10.1108/eb026526.

[23] Omar Khattab and Matei Zaharia. *ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT*. 2020. arXiv: 2004.12832 [cs.IR].

[24] Austin C. Kozlowski, Matt Taddy, and James A. Evans. "The Geometry of Culture: Analyzing the Meanings of Class through Word Embeddings". In: *American Sociological Review* 84.5 (Sept. 2019), pp. 905–949. ISSN: 1939-8271. DOI: 10.1177/0003122419877135. URL: http://dx.doi.org/10.1177/0003122419877135.

[25] Robert Lakatos et al. *Investigating the performance of Retrieval-Augmented Generation and fine-tuning for the development of AI-driven knowledge-based systems*. 2024. arXiv: 2403.09727 [cs.CL].

[26] Ngoc Luyen Le, Marie-Hélène Abel, and Philippe Gouspillou. *Combining Embedding-Based and Semantic-Based Models for Post-hoc Explanations in Recommender Systems*. 2024. arXiv: 2401.04474 [cs.IR].

[27] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. 2021. arXiv: 2005.11401 [cs.CL].

[28] Tianle Li et al. *Long-context LLMs Struggle with Long In-context Learning*. 2024. arXiv: 2404.02060 [cs.CL].

[29] Chin-Yew Lin. "ROUGE: a Package for Automatic Evaluation of Summaries". In: *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*. Barcelona, Spain, July 2004, pp. 74–81.

[30] Nelson F. Liu et al. *Lost in the Middle: How Language Models Use Long Contexts*. 2023. arXiv: 2307.03172 [cs.CL].

[31] Daniel Machlab and Rick Battle. *LLM In-Context Recall is Prompt Dependent*. 2024. arXiv: 2404.08865 [cs.CL].

[32] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2009. ISBN: 978-0521865715.

[33] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[34] Andriy Mnih and Yee Whye Teh. *A Fast and Simple Algorithm for Training Neural Probabilistic Language Models*. 2012. arXiv: 1206.6426 [cs.CL].

[35] Niklas Muennighoff et al. *MTEB: Massive Text Embedding Benchmark*. 2023. arXiv: 2210.07316 [cs.CL].

[36] Arvind Neelakantan et al. *Text and Code Embeddings by Contrastive Pre-Training*. 2022. arXiv: 2201.10005 [cs.CL].

[37] OpenAI. *New embedding models and API updates*. 2024. URL: https://openai.com/index/new-embedding-models-and-api-updates/ (visited on 05/10/2024).

[38] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL].

[39] D. Opitz and R. Maclin. "Popular Ensemble Methods: An Empirical Study". In: *Journal of Artificial Intelligence Research* 11 (Aug. 1999), pp. 169–198. ISSN: 1076-9757. DOI: 10.1613/jair.614. URL: http://dx.doi.org/10.1613/jair.614.

[40] Long Ouyang et al. *Training language models to follow instructions with human feedback*. 2022. arXiv: 2203.02155 [cs.CL].

[41] Yikang Pan et al. *On the Risk of Misinformation Pollution with Large Language Models*. 2023. arXiv: 2305.13661 [cs.CL].

[42] Richard Yuanzhe Pang et al. *QuALITY: Question Answering with Long Input Texts, Yes!* 2022. arXiv: 2112.08608 [cs.CL].

[43] Arjun Panickssery, Samuel R. Bowman, and Shi Feng. *LLM Evaluators Recognize and Favor Their Own Generations*. 2024. arXiv: 2404.13076 [cs.CL].

[44] Kishore Papineni et al. “BLEU: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*. 2002, pp. 311–318. citeseerx: 10.1.1.19.9416. URL: http://www.aclweb.org/anthology/P02-1040.pdf.

[45] Krishna Pillutla et al. *MAUVE: Measuring the Gap Between Neural Text and Human Text using Divergence Frontiers*. 2021. arXiv: 2102.01454 [cs.CL].

[46] Christopher A. Ramezan et al. “Effects of Training Set Size on Supervised Machine-Learning Land-Cover Classification of Large-Area High-Resolution Remotely Sensed Data”. In: *Remote Sensing* 13.3 (2021), p. 368. DOI: 10.3390/rs13030368. URL: https://www.mdpi.com/2072-4292/13/3/368.

[47] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL].

[48] Yunwei Ren and Yuanzhi Li. *On the Importance of Contrastive Loss in Multimodal Learning*. 2023. arXiv: 2304.03717 [cs.LG].

[49] Stephen Robertson and Hugo Zaragoza. “The Probabilistic Relevance Framework: BM25 and Beyond”. In: *Foundations and Trends in Information Retrieval* 3.4 (2009), pp. 333–389. DOI: 10.1561/1500000019.

[50] Stephen E. Robertson et al. “Okapi at TREC-3”. In: *Proceedings of the Third Text REtrieval Conference (TREC 1994)*. Gaithersburg, USA: National Institute of Standards and Technology (NIST), Nov. 1994, pp. 109–126.

[51] Guilherme Moraes Rosa et al. *Yes, BM25 is a Strong Baseline for Legal Case Retrieval*. 2021. arXiv: 2105.05686 [cs.IR].

[52] Maja Rudolph et al. *Structured Embedding Models for Grouped Data*. 2017. arXiv: 1709.10367 [cs.CL].

[53] Keshav Santhanam et al. *ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction*. 2022. arXiv: 2112.01488 [cs.IR].

[54] Lutfi Kerem Senel et al. “Semantic Structure and Interpretability of Word Embeddings”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.10 (2018). ISSN: 2329-9304. DOI: 10.1109/taslp.2018.2837384. URL: http://dx.doi.org/10.1109/TASLP.2018.2837384.

[55] Rico Sennrich, Barry Haddow, and Alexandra Birch. *Neural Machine Translation of Rare Words with Subword Units*. 2016. arXiv: 1508.07909 [cs.CL].

[56] Richard Shin and Benjamin Van Durme. *Few-Shot Semantic Parsing with Language Models Trained On Code*. 2022. arXiv: 2112.08696 [cs.CL].

[57] Nisan Stiennon et al. *Learning to summarize from human feedback*. 2022. arXiv: 2009.01325 [cs.CL].

[58] Nandan Thakur et al. *BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models*. 2021. arXiv: 2104.08663 [cs.IR].

[59] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: `2302.13971 [cs.CL]`.

[60] Pablo Villalobos et al. *Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning*. 2022. arXiv: `2211.04325 [cs.LG]`.

[61] Kexin Wang, Nils Reimers, and Iryna Gurevych. *TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning*. 2021. arXiv: `2104.06979 [cs.CL]`.

[62] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. *Jailbroken: How Does LLM Safety Training Fail?* 2023. arXiv: `2307.02483 [cs.LG]`.

[63] Jason Wei et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: `2206.07682 [cs.CL]`.

[64] Junchao Wu et al. *A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions*. 2024. arXiv: `2310.14724 [cs.CL]`.

[65] Shuo Yang et al. *Rethinking Benchmark and Contamination for Language Models with Rephrased Samples*. 2023. arXiv: `2311.04850 [cs.CL]`.

[66] Hugh Zhang et al. *A Careful Examination of Large Language Model Performance on Grade School Arithmetic*. 2024. arXiv: `2405.00332 [cs.CL]`.

[67] Tianyi Zhang et al. *BERTScore: Evaluating Text Generation with BERT*. 2020. arXiv: `1904.09675 [cs.CL]`.

[68] Penghao Zhao et al. *Retrieval-Augmented Generation for AI-Generated Content: A Survey*. 2024. arXiv: `2402.19473 [cs.CV]`.

[69] Wayne Xin Zhao et al. *A Survey of Large Language Models*. 2023. arXiv: `2303.18223 [cs.CL]`.

[70] Vitalii Zhelezniak et al. *Correlation Coefficients and Semantic Textual Similarity*. 2019. arXiv: `1905.07790 [cs.CL]`.

[71] Kaitlyn Zhou et al. *Problems with Cosine as a Measure of Embedding Similarity for High Frequency Words*. 2022. arXiv: `2205.05092 [cs.CL]`.

[72] Kaijie Zhu et al. *PromptBench: Towards Evaluating the Robustness of Large Language Models on Adversarial Prompts*. 2023. arXiv: `2306.04528 [cs.CL]`.

[73] Yutao Zhu et al. *Large Language Models for Information Retrieval: A Survey*. 2024. arXiv: `2308.07107 [cs.CL]`.

[74] Wei Zou et al. *PoisonedRAG: Knowledge Poisoning Attacks to Retrieval-Augmented Generation of Large Language Models*. 2024. arXiv: `2402.07867 [cs.CR]`.