57º SBPO
SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
Gramado • RS
DE 5 A 9 DE OUTUBRO DE 2025

A Hypervolume-Based Comparison of NSGA-II and NSGA-III for Three-Objective Extractive Text Summarization

Bruno Roth Vasconcellos
Pontifical Catholic University of Parana
R. Imac. Conceicao, 1155 - Prado Velho, Curitiba - PR
bruno.rothv@gmail.com

Roberta Lerin Lovisi
Pontifical Catholic University of Parana
R. Imac. Conceicao, 1155 - Prado Velho, Curitiba - PR
roberta.lovisi@gmail.com

Gilberto Reynoso Meza
Pontifical Catholic University of Parana
R. Imac. Conceicao, 1155 - Prado Velho, Curitiba - PR
g.reynosomeza@pucpr.br

ABSTRACT

Long documents pose challenges both for human readers, who may struggle to quickly extract the main ideas, and for retrieval-augmented generation (RAG) systems, which face difficulties in selecting and providing the most relevant context when processing entire texts. Extractive text summarization represents a solution for this, but its usefulness depends on selecting the best possible summary considering, among other objectives, the essential information with minimal length. This work addresses focuses on identifying high-quality summaries by treating sentence selection as multi-objective optimization problem that must balance competing objectives: ensuring semantic coverage, prioritizing relevant portions of the text, and maintaining the summary sufficiently short. It compares two widely used optimization evolutionary algorithms, NSGA-II and NSGA-III, on 462 CNN DailyMail news articles. The results indicate a superiority of NSGA-II when analyzing quality alone, but NSGA-III presents an advantage when considering performance and solution selectivity, which enriches the discussion on the benefits trade-offs.

KEYWORDS. Multi-objective optimization. Extractive Text summarization. Hypervolume.

Extractive Text Summarization; Evolutionary Multi-Objective Optimization; NSGA-II vs. NSGA-III Comparative Analysis; Hypervolum-Based Performance Evaluation

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

1. Introduction

Extractive text summarization has become a valuable technique for efficiently condensing large documents by selecting and reassembling the most relevant sentences. Unlike abstractive methods that attempt to generate new text, often relying on large language models (LLMs), extractive summarization preserves the original wording, ensuring factual consistency while being computationally lighter and faster [Abo-Bakr and Mohamed, 2023]. In the current landscape, in which LLMs are widely used for information synthesis, extractive summarization offers a complementary solution. LLMs can struggle with hallucinations, factual drift, or high computational costs, especially when dealing with long documents [Lewis et al., 2020]. Extractive methods mitigate these issues by pre-selecting informative, coherent sections of text and reducing the input size, or even serving as a standalone summarization method without requiring heavy model inference.

This approach can be particularly beneficial for retrieval-augmented generation (RAG) systems, which rely on efficiently retrieving and presenting relevant context to language models. High-quality extractive summaries improve retrieval precision, reduce noise, and enhance the factual foundation of downstream tasks such as question answering, knowledge base construction, and document comprehension. Recent studies, such as RAPTOR (Recursive Abstractive Processing for Tree-Organized Retrieval) [Sarthi et al., 2024], demonstrated that organizing summaries hierarchically within RAG pipelines can significantly enhance multi-step reasoning and long context understanding, achieving state-of-the-art results on benchmark datasets. For example, in a RAG pipeline, instead of feeding a 50-page document to an LLM, a system can first build a focused 3-page extractive summary that concentrates the most relevant passages, which lowers cost, reduces noise, and often improves answer accuracy downstream.

Modern extractive summarization strategies leverage neural sentence embeddings, such as those generated by sentence transformers, to capture semantic relationships without the need for aggressive pre-processing like stop-word removal or punctuation filtering. By working directly on dense vector representations, extractive summarization can identify semantically important sentences, enabling more robust and domain-adaptable summarization pipelines.

Given the multi-faceted nature of summarization, balancing content coverage while minimizing the length of the text, multi-objective optimization (MOO) techniques provide a structured framework to select the optimal combination of sentences, generating summaries that meet conflicting criteria simultaneously. Multi-objective optimization searches for a set of optimal trade-off solutions, called a Pareto front, where improving one objective necessarily worsens at least one other objective. This work aims to compare the performance of two prominent evolutionary algorithms, Non-dominated Sorting Genetic Algorithm II (NSGA-II) and III (NSGA-III), to achieve coherent but efficient text summaries.

This document is organized as follows: Section 2 provides a brief overview of MOO theory and introduces the algorithms employed in this study. Section 3 defines the problem, the analysis approach, and presents the objective functions used in the optimization process. Section 4 discusses the experimental results and highlights the key findings. Finally, Section 5 summarizes the conclusions and offers suggestions for future research.

2. Background

Many problems and systems, particularly in computational intelligence, are fundamentally optimization problems. They are often non-trivial, involving complex and potentially conflicting relationships among objective functions, or criteria, to be minimized or maximized. Once it is not possible to achieve one unique solution that simultaneously meet all the criteria, multi-objective optimization approaches can be used to generate the optimal solutions for specific problems [Reynoso-Meza et al., 2016].

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

Among the optimization algorithms to solve these problems, population-based evolutionary algorithms offer several advantages over traditional optimization techniques and single-solution metaheuristics. Unlike gradient-based approximation methods, which are sensitive to initial conditions and often converge to local optima, evolutionary or genetic algorithms employ a population of candidate solutions, enabling parallel and wide exploration of the solution space. These methods reduce the probability to premature convergence and can also handle discrete and non-differentiable functions, which make them powerful to be applied in many different problems.

NSGA-II and NSGA-III are two well-known and explored genetic algorithms due to their simplicity and efficiency [Chaudhari et al., 2021]. Both maintain a population of candidate summaries and iteratively evolve them via selection, recombination, and mutation, but differ in how they preserve diversity along the Pareto front. In NSGA-II, elitism is achieved by merging parent and offspring populations and preserving the best non-dominated fronts along generations, while the crowding distance is used to manage diversity when truncation is required, promoting a well spread set of solutions. NSGA-III uses reference directions to preserve diversity in the solution set [Li et al., 2018], which is especially effective when the number of objectives is greater than three.

Once text summarization encompasses high computational cost and non-differentiable behavior, it is important to apply an algorithm that balances performance and solution quality. In order to compare and determine which of the methods better optimize a multi-objective optimization (MOO) extractive text summarization task, it is possible to calculate the hypervolumes related to each Pareto front, and compare their sizes based on a same reference point.

2.1. Non-dominated Sorting Genetic Algorithm II

NSGA-II (Non-dominated Sorting Genetic Algorithm II) is one of the widely adopted multi-objective evolutionary algorithms due to its effective balance of convergence and diversity without requiring user-defined parameters for diversity control. By combining a fast non-dominated sorting procedure with a crowding-distance metric, NSGA-II efficiently identifies Pareto-optimal solutions and maintains a well-distributed front across competing objectives [Deb et al., 2002]. Its relatively low computational overhead and straightforward implementation make it a natural baseline for multi-objective optimization in domains ranging from engineering design to text summarization. The algorithm proceeds in two parts as follows:

1. **Initialization**: generate an initial population P_0 of size N generated by sampling binary strings of length $|D|$.

2. **Non-dominated Sorting**: partition P_t into fronts $F_1, F_2, \ldots$ in which F_1 contains all non-dominated individuals, F_2 those dominated only by members of F_1, and so on.

3. **Crowding Distance**: for each front F_k, compute a crowding distance for each individual by summing normalized distances to its nearest neighbors in objective space.

4. **Parent Selection**: use binary tournament with the crowded comparison operator [Deb et al., 2002]. First select all individuals from front F_1, then front F_2 and then on until Given two candidates a and b, prefer the one with lower front rank. If the ranks are equal, prefer the one with higher crowding distance. If both criteria are equal, select one at random.

5. **Reproduction**: apply crossover and mutation to create an offspring population Q_t of size N.

6. **Survivor Selection**: form $R_t = P_t \cup Q_t$, sort R_t into fronts, and fill the next generation P_{t+1} with size N by adding entire fronts until capacity, then selecting by descending crowding distance within the final partial front.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

2.2. Non-dominated Sorting Genetic Algorithm III

Non-dominated Sorting Genetic Algorithm III (NSGA-III) was introduced to address the challenge of maintaining diversity in problems with more than three objectives by replacing the crowding-distance mechanism with a reference-direction based diversity preservation strategy [Deb and Jain, 2014]. While originally designed for many-objective optimization, NSGA-III's structured diversity control via a predefined set of direction vectors can also benefit three objective tasks by ensuring an uniform spread of solutions along the Pareto front, thereby reducing gaps and improving the overall quality of the approximation set. The algorithm is computed as the following:

1. **Initialization & Non-dominated Sorting**: identical to NSGA-II.

2. **Normalization**: within each front, normalize objective values using extreme points and intercepts to account for scale differences.

3. **Reference Directions**: predefine a set of M evenly spaced unit vectors $\{\mathbf{d}_1, \ldots, \mathbf{d}_M\}$ in objective space.

4. **Association**: assign each normalized solution to its nearest reference direction by perpendicular distance.

5. **Diversity Preservation**: when filling P_{t+1}, prefer individuals in less-populated reference niches to ensure a uniform spread across all directions.

6. **Reproduction & Selection**: identical to NSGA-II except that niche counts replace crowding distances for tie-breaking in the final front.

2.3. Hypervolume Indicator

The hypervolume (HV) indicator measures the size of the objective space that is dominated by a Pareto approximation set and bounded by a user-defined reference point $\mathbf{r}$. For a set of solutions P in a three-objective minimization problem, the reference point $\mathbf{r} = (r_1, r_2, r_3)$ is usually chosen beyond the worst observed solution, which is also known as nadir point. The hypervolume is defined as in Equation 1.

$$\mathrm{HV}(P) \;=\; \mathrm{Vol}\Big(\bigcup_{\mathbf{f}\in P} [f_1,\, r_1] \times [f_2,\, r_2] \times [f_3,\, r_3]\Big) \tag{1}$$

There, $\mathbf{f} = (f_1, f_2, f_3)$ is the triplet of objective values for a candidate solution, and $\mathrm{Vol}(\cdot)$ denotes the Lebesgue volume in $\mathbb{R}^3$. A larger hypervolume indicates better convergence toward the ideal solution (closer to the origin of the space) and a more diverse spread of trade-off solutions, as represented in Figure 1.

Regarding text summarization, comparing NSGA-II and NSGA-III via hypervolume enables the capture of the combined effect of all objectives functions related in the problem - maximizing semantic coverage, favoring salient sentence positions, and enforcing concise length - within a single scalar metric. Once summarization inherently involves trade-offs (e.g., adding sentences to improve coverage may worsen brevity), the hypervolume reflects how effectively each algorithm navigates these competing demands. An algorithm that consistently produces Pareto sets with higher hypervolume is better suited to generate summaries that balance comprehensiveness, coherence, and conciseness across a diverse range of acceptable summary lengths and styles.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

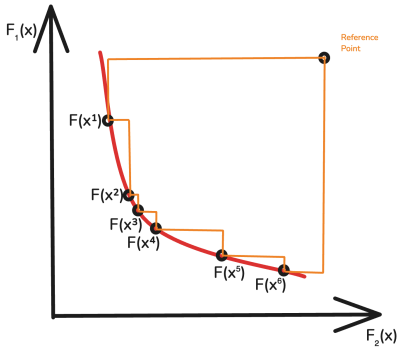


Figure 1: Representation of the hypervolume of a Pareto front

3. Methodology

To prepare the documents for multi-objective optimization, a natural language processing pipeline is applied to represent the text in a structured and semantically meaningful manner. Unlike other works that adopt classical preprocessing techniques such as punctuation removal, stop-word elimination, and stemming [Sanchez-Gomez et al., 2020a], this study preserves the original sentences in their entirety, maintaining the full richness and nuance of the text.

The document is first segmented into complete sentences using sentence tokenization techniques. Each sentence is then embedded into a dense vector representation using a pre-trained sentence transformer model (`all-MiniLM-L6-v2`), which captures the semantic meaning of each sentence within a shared embedding space. This process transforms the document into a matrix of semantic vectors, allowing comparisons between sentences based on their meaning rather than their surface form. Figure 2 illustrates this overall preparation process.

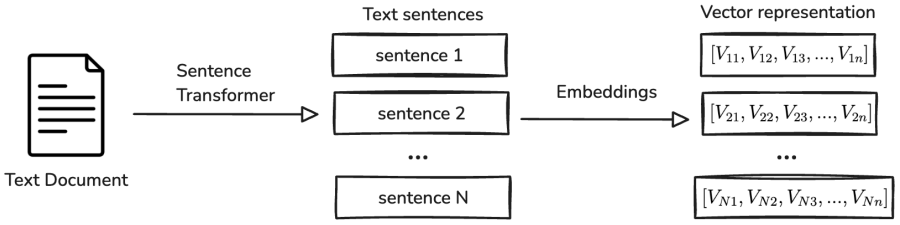


Figure 2: NLP pipeline

Following the embedding step, a similarity matrix is computed using the cosine similarity between each pair of sentence embeddings. This matrix reflects the semantic similarity between sentences, enabling the system to identify redundancy and ensure broad coverage of the document's main topics.

Subsequently, extractive summarization is achieved by selecting a subset of sentences that maximizes important summarization criteria. A MOO process is applied to find the best subset of sentences that balances semantic coverage, redundancy reduction, and information density. The optimization seeks to generate a summary that is compact while maintaining the essential information and semantic integrity of the original document.

3.1. Objective Functions

The extractive summarization task is formulated as a three-objective optimization problem, each of which quantifies a distinct trade-off: coverage, position, and length penalty. The

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

following sections present their formal definitions and mathematical formulations. Table 1 presents a concise overview of the three objectives and the optimization direction for each.

Table 1: Objective Functions

Objective	Direction	Purpose
Coverage	Maximize	Ensure selected sentences collectively cover the full content of the document by maximizing the average of the maximum cosine similarities between each document sentence and the chosen summary sentences.
Position	Minimize	Encourage selection of sentences near the beginning or end of the document, via a parabolic weighting function, considering that sentences in these positions cover the most part of the information.
Length Penalty	Minimize	Penalize summaries whose length deviates from a target range, using an asymmetric penalty that penalizes overly long summaries while tolerating shorter ones.

3.1.1. Coverage Objective

The first objective function defined in the optimization process is semantic coverage. Its purpose is to ensure that the selected summary sentences adequately represent the overall content of the original document. This objective is common in extractive summarization and rewards summaries whose sentences collectively cover the source content [Saleh et al., 2015; Sanchez-Gomez et al., 2018, 2020b].

Let the document be the sequence of sentences $D = \{s_1, \ldots, s_n\}$. A candidate extractive summary is a subset $S \subseteq D$ with $|S| = k$. To calculate coverage, a full similarity matrix is constructed by computing the cosine similarity between all pairs of sentences in the document, based on their dense vector embeddings. Using the precomputed similarity matrix $\text{sim}(s_i, s_j)$, a submatrix can be extracted on demand containing the similarities between each selected sentence and all other sentences in the original document. This submatrix has dimensions $[|S| \times |D|]$, where $|S|$ is the number of selected sentences and $|D|$ is the total number of document sentences.

For each document sentence $s_i \in D$, the goal is to find the maximum similarity score with any selected summary sentence $s_j \in S$. Then, the overall coverage score for a candidate summary S is computed as the average of the maximum similarities across all document sentences and is represented in Equation 2. There, k is the number of sentences in the summary evaluated and n is the total length of the original document.

$$\text{Coverage}(S) = \frac{1}{k} \sum_{i=1}^{n} \max_{s_j \in S} \text{sim}(s_i, s_j) \tag{2}$$

Since the optimization problem is formulated as a minimization task, the coverage, originally a maximization objective, was inverted, as shown in Equation 3 to ensure that lower values correspond to higher coverage, in which 0 is best and 1 is worst.

$$\min_{S \subseteq D} 1.0 - \text{Coverage}(S) \tag{3}$$

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

This formulation encourages the summaries remain both comprehensive and aligned with natural writing structures.

3.1.2. Position Objective

The second objective function is sentence position importance. Its purpose is to encourage the selection of sentences that appear near the beginning or the end of the document, under the assumption that these positions often carry the most relevant information for the understanding of the original text.

Considering that each selected sentence $s_j \in S$ has an original index i_j related to its position in the document of total length n, to calculate the position score for a candidate summary $S \subseteq D$, one can follow the steps below.

- The indices are normalized in relation to the original text length:

$$p_j = \frac{i_j}{n-1}, \quad p_j \in [0,1]$$

- A parabolic weighting $w(p_j)$ function assigns lower weights to sentences near the start ($p_j \approx 0$) and end ($p_j \approx 1$), and higher weights to those near the middle ($p_j \approx 0.5$).

$$w(p_j) \;=\; 1 \;-\; 4\left(p_j - 0.5\right)^2$$

  Here, 0.5 is subtracted from p_j, centering the values around zero: sentences in the middle tend to 0.0, while those at the edges reach -0.5 and 0.5. The result is then squared, forming a positive parabola that equalizes the sign and emphasizes distance from the center, and multiplied by 4 to scale the maximum to 1.0. Finally, once optimization algorithms usually minimize objective functions, this value is subtracted from 1.0 to invert the parabola. As a result, the weight function $w(p_j)$ equals 0.0 for edge sentences ($p_j = 0$ or $p_j = 1$) and 1.0 for central sentences ($p_j = 0.5$).

- The overall position score of the candidate summary is the average of its sentence weights:

$$\text{Position}(S) \;=\; \frac{1}{|S|} \sum_{s_j \in S} w(p_j)$$

- Finally, the position objective is designed as a minimization problem, as in Equation 4.

$$\min_{S \subseteq D} \text{Position}(S) \tag{4}$$

3.1.3. Length Penalty Objective

The third optimization objective penalizes summaries whose length deviates from a desired range, prioritizing brevity while allowing some flexibility. Considering that the document contains n sentences, it is first defined the limit allowable sizes for the summary:

$$L_{\min} = \max\big(3, \lfloor 0.15\, n \rfloor\big), \quad L_{\max} = \min\big(\lfloor 0.5\, n \rfloor, \max(L_{\min} + 5,\, 15)\big)$$

These values were heuristically defined since the length of the original document can vary significantly. The goal is to keep the summary length proportional to the size of the original document, adapting dynamically.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

- $L_{\min}$ ensures a minimum meaningful coverage, with a minimum of 3 sentences, or at least 15% of the document, whichever is larger.

- $L_{\max}$ prevents overly long summaries by limiting them to at most 50% of the document length, but with an upper cap that allows for some flexibility, especially in shorter documents.

These boundaries are not fixed by any theoretical requirement, but designed to balance conciseness and informativeness across texts of varying sizes, avoiding excessively short or verbose summaries regardless of the input length. A target length L_t is set closer to the minimum to encourage brevity:

$$L_t = L_{\min} \; + \; 0.3\,(L_{\max} - L_{\min})$$

A length penalty is then computed, in which k is the length of the candidate solution:

$$\text{penalty}(k) = \begin{cases} \dfrac{k - L_t}{L_{\max} - L_t}, & k > L_t, \\ 0.5 \times \dfrac{L_t - k}{L_t - L_{\min}}, & k \leq L_t \end{cases}$$

In the end, the limits $[0, 1]$ are set to grant that it does not extrapolate boundaries:

$$\text{LengthPenalty}(S) = \min\big(1, \max\big(0, \text{penalty}(|S|)\big)\big)$$

Since the goal is to minimize the length of the summary (in which 0 is best), the penalty score for the objective function is in Equation 5.

$$\min_{S \subseteq D} \; \text{LengthPenalty}(S) \tag{5}$$

3.2. Experimental Setup

All experiments are conducted on the CNN/DailyMail dataset. To obtain a representative sample with a 95% confidence level of a 5% margin of error, it was randomly selected 462 articles out of 287,113 from the train database. Each article is preprocessed by sentence-splitting and embedding with the transformer model presented in Section 3. All experiments were run on a MacBook Pro M4 Pro with 24 GB of RAM.

Both NSGA-II and NSGA-III are implemented using the `pymoo` framework and share identical hyperparameters (settings that are standard for binary multi-objective evolutionary search were adopted): a population size of 300, 150 generations, BinaryRandomSampling initialization (to ensure a uniformly diverse starting population), TwoPointCrossover with probability 0.9 (to recombine contiguous blocks of sentence selection bits while retaining high exploration), BitflipMutation with probability $1/|D|$ (so that on average one sentence index is toggled per solution, enabling fine grained local search), and duplicate elimination[1]. The three objectives are evaluated as defined in Section 3.1.

[1]Each individual is a binary vector of length $|D|$, where 1 marks a selected sentence and 0 marks a non-selected one. BinaryRandomSampling uniformly creates binary vectors for each individual p for the initial population, TwoPointCrossover performs the recombination of two parents to create children by selecting two crossover points and swapping only the middle part (between the two points), BitflipMutation mutates each bit of the individual p vector with probability $1/|D|$, and duplicate elimination removes identical individuals to preserve diversity.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

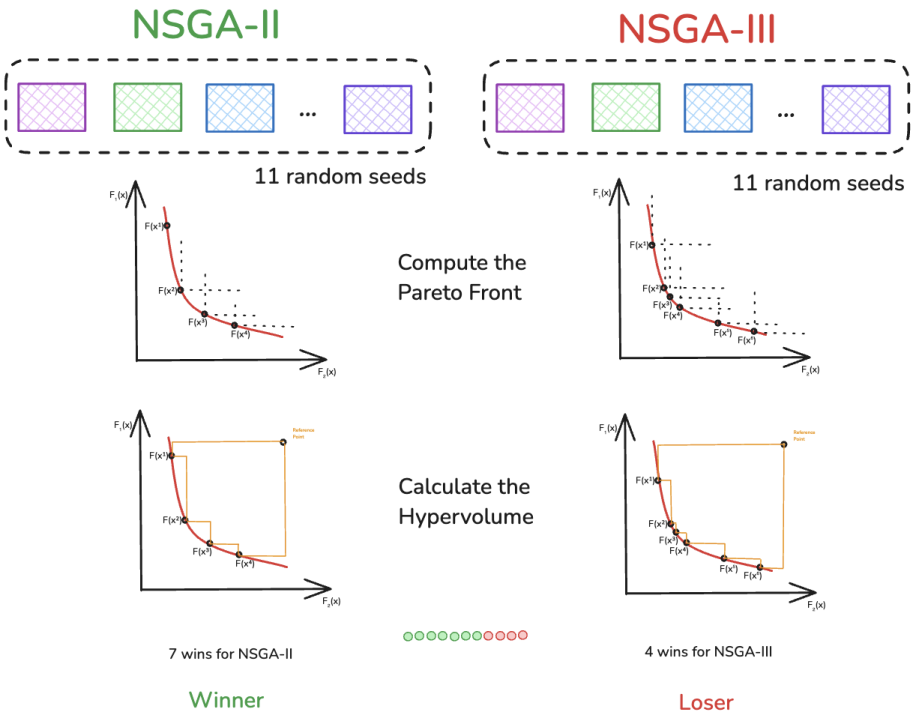


Figure 3: Comparison process

For each of the 462 articles, it was performed 11 independent runs of both NSGA-II and NSGA-III, each initialized with a distinct random seed[2]. At the end of each run, the final Pareto front is extracted and its hypervolume is computed against a common reference point, which bounds the worst-case values for the three objectives, i.e., the nadir point of the set of solutions. The algorithm with the highest median hypervolume over 11 runs for the majority of text articles is considered the best for the problem, as represented in Figure 3.

4. Results

After executing the algorithms and computing the median hypervolume across 11 runs for each article, it was observed that NSGA-II consistently outperforms NSGA-III. Specifically, in 97% of the evaluated texts, NSGA-II achieved a higher hypervolume than NSGA-III. This is illustrated in Figure 4, which shows the difference in hypervolumes between the two algorithms (with positive values indicating an advantage for NSGA-II). Given that the hypervolume values and their respective differences are of a small magnitude (on the order of 10^{-2}), they were scaled by a factor of 10^2 to enhance interpretability in the Figure 4.

However, NSGA-III consistently returned fewer Pareto-optimal solutions (Figure 5) than NSGA-II but fairly equidistanced when compared to NSGA-II, which is a direct result of its reference direction mechanism. In this case specifically, it was used a fixed set of reference directions (23 in each run). Nevertheless, the fact that NSGA-III achieves Pareto fronts with hypervolume values differing from those of NSGA-II by only an order of 10^{-2}, despite returning fewer solutions, shows the algorithm's effectiveness and precision in its trade-offs.

[2]It was used an odd quantity of runs so the median is the single middle run, and eleven is a practical balance of reliability and time.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

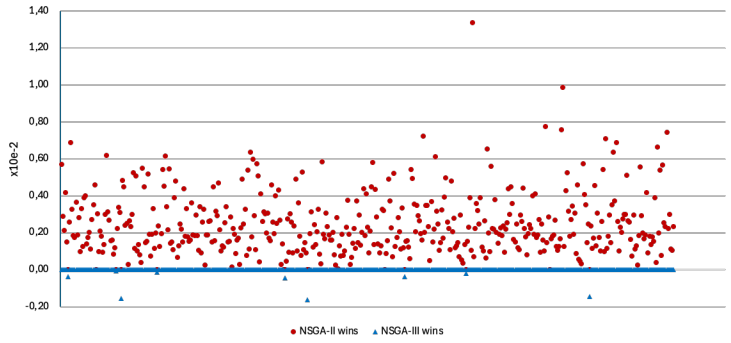


Figure 4: Difference between NSGA-II and NSGA-III hypervolumes

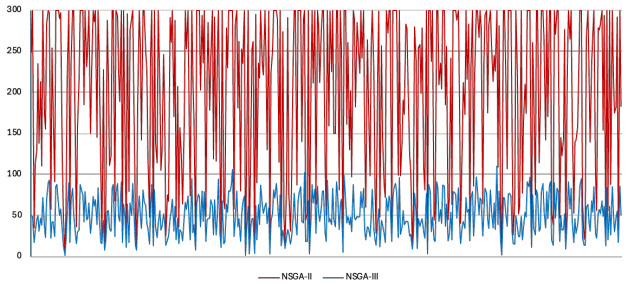
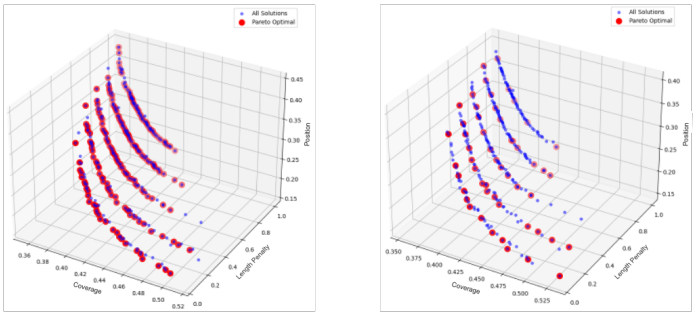


Figure 5: Number of solutions in the Pareto front, used to calculate the hypervolumes selected, from each algorithm

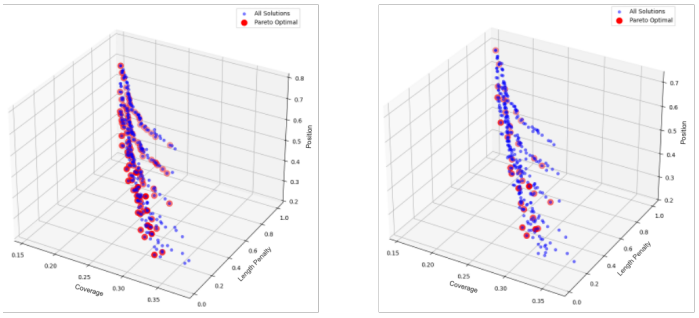
Figures 6 and 7 present the Pareto fronts obtained by NSGA-II and NSGA-III for two different news articles. In both cases, the general shape and distribution of solutions in the objective space are similar, indicating that both approaches converge toward comparable regions of the Pareto-optimal space. As noted by [Chaudhari et al., 2021], both algorithms behave similarly in the convergence of the solutions, however, NSGA-II's optimal solutions cover the entire Pareto front but are not equidistant, whereas NSGA-III presents a widespread front with almost equidistant solutions.



a) NSGA-II Pareto front b) NSGA-III Pareto front

Figure 6: Pareto front of a news article by NSGA-II and NSGA-III

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br



a) NSGA-II Pareto front b) NSGA-III Pareto front

Figure 7: Pareto front of a news article by NSGA-II and NSGA-III

5. Conclusion and future work

The findings of this work highlight the importance of evaluating multi-objective optimization algorithms as a whole, rather than simply declaring a winner based on a single metric. While hypervolume analysis alone often favors NSGA-II, incorporating additional diversity metrics, such as spacing or spread, and performance reveals that NSGA-III tends to converge to a smaller set of selective, equidistant, and diverse solutions.

In modern applications, particularly high-dimensional and computationally-intensive tasks, being directionally correct and computationally efficient is often more valuable than achieving complete, fine-grained trade-off exploration. So, although NSGA-II may offer advantages in scenarios that require finer granularity in trade-off exploration, NSGA-III emerges as a more efficient and lightweight alternative for high-dimensional and computationally-heavy applications, like text summarization for feeding RAG systems. The relatively small difference in hypervolumes between the algorithms suggests that NSGA-III offers a favorable cost-quality trade-off, while the reduced number of solutions simplifies decision-making by the problem designer.

For future works, several possibilities can be considered. First, considering the NSGA-III performance and trade-off as previously discussed, and that it is a many-objective algorithm, it may be interesting to incorporate additional objectives, such as readability scores (semantic coherence) or redundancy. This could reveal how well the algorithm scales beyond three objectives and whether it becomes clearly more advantageous than NSGA-II, e.g., in truly many-objective settings.

Second, extending the evaluation to larger and more diverse texts (e.g., scientific articles, legal texts, multi-lingual datasets, long interviews or even patient reports in health industry), would test the robustness of the application under varying document structures and lengths.

Third, alongside the hypervolume and diversity analyses, it is interesting to evaluate the generated summaries using standard summarization metrics, like ROUGE for n-gram overlap and BERTScore for semantic similarity. This will help correlate Pareto-front quality with human-oriented measures of informativeness and fluency, ensuring that gains in multi-objective optimization translate into real improvements in summary quality.

References

Abo-Bakr, H. and Mohamed, S. A. (2023). Automatic multi-documents text summarization by a large-scale sparse multi-objective optimization algorithm. *Complex Intelligent Systems*.

Chaudhari, P., Thakur, A. K., Kumar, R., Banerjee, N., and Kumar, A. (2021). Comparison of nsga-iii with nsga-ii for multi objective optimization of adiabatic styrene reactor. *Materials Today: Proceedings*, 57:1509–1514.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Deb, K. and Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601.

Lewis, P., Perez, E., Piktusa, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *In Advances in Neural Information Processing Systems, volume 33, pages 9459–9474. Curran Associates, Inc.*

Li, Y., Kou, Y., and Li, Z. (2018). An improved nondominated sorting genetic algorithm iii method for solving multiobjective weapon-target assignment part i: The value of fighter combat. *International Journal of Aerospace Engineering*, 2018:1–23.

Reynoso-Meza, G., Blasco, X., Sanchis, J., and Martinez, M. (2016). Preference driven multiobjective optimization design procedure for industrial controller tuning. *Information Sciences*.

Saleh, H., Kadhim, N., and Attea, B. (2015). A genetic based optimization model for extractive multi-document text summarization. 56:1489–1498.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., and Pérez, C. J. (2018). Extractive multidocument text summarization using a multi-objective artificial bee colony optimization approach. *Knowledge-Based Systems*, 159:1–8. ISSN 0950-7051. URL https://www.sciencedirect.com/science/article/pii/S0950705117305609.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., and Pérez, C. J. (2020a). A decomposition-based multi-objective optimization approach for extractive multi-document text summarization. *Applied Soft Computing*, 91:106231. ISSN 1568-4946. URL https://www.sciencedirect.com/science/article/pii/S156849462030171X.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., and Pérez, C. J. (2020b). Experimental analysis of multiple criteria for extractive multi-document text summarization. *Expert Systems with Applications*, 140:112904. ISSN 0957-4174. URL https://www.sciencedirect.com/science/article/pii/S0957417419306220.

Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., and Manning, C. D. (2024). Raptor: Recursive abstractive processing for tree-organized retrieval. URL https://arxiv.org/abs/2401.18059.

DOI: 10.59254/sbpo-2025-212185
https://proceedings.science/p/212185?lang=pt-br