Introduction to Bioinformatics

BLG 348E

Term Project Report

KAAN KARATAŞ
150200081
ALP GİRAY KAÇIRA
150170091

Faculty of Computer and Informatics Engineering
Department of Computer Engineering
Date of submission: 20.12.2023

CONTENTS

1. Abstract

The project explores the comparative performance of various next-generation sequencing (NGS) data processing algorithms using the Comparative Sequencing Analysis Platform (CoSAP). CoSAP serves as an intuitive and comprehensive pipeline creation tool for NGS data, facilitating reproducibility and in-depth insights into the capabilities and limitations of current tools. The focus of the study revolves around variant calling pipelines, encompassing Read Trimming, Read Mapping, Duplicate Removal-Base Calibration, Variant Calling, and Variant Annotation steps.

The imported CoSAP library offers a range of tool options for each step, allowing users to configure parameters by adding or removing specific pipeline components. Within the project scope, pipelines are created and variant calling operations are applied to designated FASTQ files. The objective is to assess the performance of variant calling operations across different mappers and variant callers.

The project operatos on various combinations of mappers and callers, with an emphasis on the base recalibration parameter. Parameters like trimming and duplicate handling are fixed, ensuring a consistent experimental setup. The project involves two mapper algorithms, BWA and Bowtie, and three variant calling algorithms, namely Somatic-Sniper, Mutect, and Strelka. In total, 12 pipelines are generated, encompassing different configurations resulting from variations in base recalibration options, mapper choices, and variant callers.

The evaluation of pipeline outputs involves the comparison of results and the utilization of various metrics for performance assessment. The project is structured into four main parts: gaining an understanding of NGS algorithms, generating pipelines with different configurations using CoSAP library, comparing the pipelines through visualizations such as Principal Component Analysis and Heatmaps, and presenting the results. This section serves to show the essence of the analyses conducted, providing a concise summary of the project’s objectives, methods, and key findings.

2. Introduction

Advancements in DNA sequencing technologies have resulted in a new era of genomic exploration, enabling biological information to be interpreted with better precision. The focus of this study lies at the intersection of next-generation sequencing (NGS) and the Comparative Sequencing Analysis Platform (CoSAP). As DNA sequencing continues to evolve, so does the need for algorithms and data analysis tools to process the data generated from these experiments.

The project uses CoSAP library as a comprehensive and user-friendly pipeline creation tool for NGS data. CoSAP facilitates the comparison of different sequencing algorithms, offering a platform to assess the performance of various pipelines. The fundamental components of a typical variant calling pipeline, such as Read Trimming, Read Mapping, Duplicate Removal-Base Calibration, Variant Calling steps, are explored within the context of the project.

In the field of DNA sequencing, the algorithm selection plays a pivotal role in the accuracy and efficiency of variant calling processes. The project study delves into the intricacies of these algorithms, particularly focusing on read trimming, mapping, and variant calling tools. The versatile use cases of CoSAP allows users to configure pipeline parameters, introducing an element of flexibility and customization in the experimental design.

The datasets used in this project consist of specific FASTQ files chosen for their relevance to variant calling operations. The selected datasets are subjected to variant calling across different mappers and variant callers, creating a diverse set of pipelines for evaluation. The project specifically investigates the impact of base recalibration options, utilizing two mapper algorithms (BWA and Bowtie) and three variant calling algorithms (Somatic-Sniper, Mutect, and Strelka). The resulting 12 pipelines represent distinct configurations that contribute to the comparative analysis.

In the course of this study, the computational infrastructure plays a crucial role in executing and managing the complex algorithms and datasets. The properties of the computer used to execute pipelines is as follows:

- Microsoft Windows 11: Windows Subsystem for Linux
- 16 GB RAM
- 1 TB Storage Capacity
- Monster: Tulpar T7 V21.2

This introduction sets the stage for a comprehensive exploration of NGS algorithms, CoSAP, and the connection between computational resources and genomic data processing.

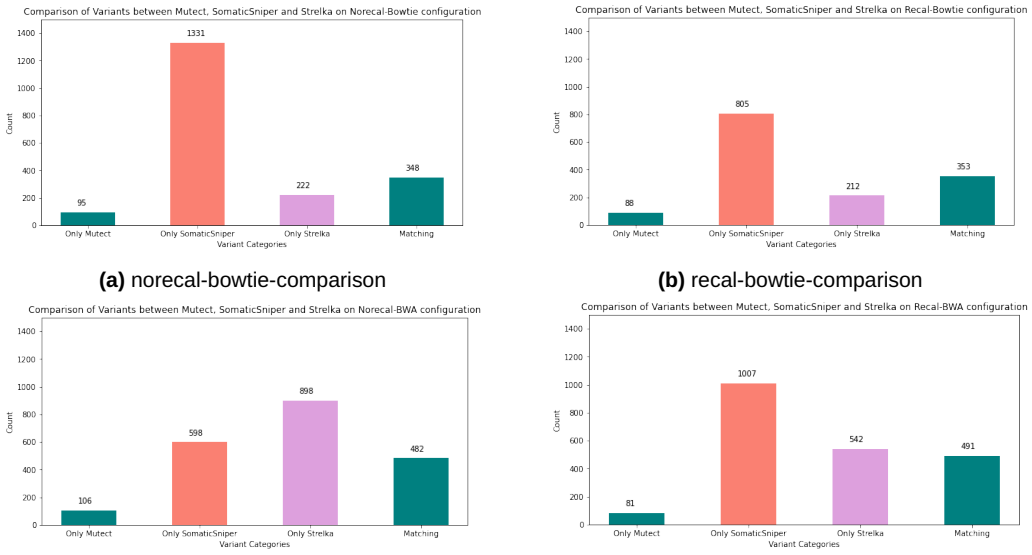
3. Results

3.1. Variant Caller Comparison

In understanding genetic variations through genomic variant calling, how we set up the analysis can greatly affect the results. In this section, we take a close look at the variant calls made by three variant calling algorithms Mutect, SomaticSniper, and Strelka across four different setups. Each setup is a unique combination of initial steps, specifically adjusting base recalibration and the choice of mapper algorithms (Bowtie and BWA).

We use histograms to show the comparison, with each histogram representing a specific setup Norecal-Bowtie, Recal-Bowtie, Norecal-BWA, and Recal-BWA. These visual graphs give us a picture of the types of variants found by each algorithm, highlighting both the ones that are unique to a algorithm and those that are found by three of them.

Through this comparison, we aim to uncover how the choices we make in preparing and analyzing genomic data impact the results obtained from Mutect, SomaticSniper, and Strelka.



(a) norecal-bowtie-comparison

(b) recal-bowtie-comparison

(c) norecal-bwa-comparison

(d) recal-bwa-comparison

Figure 3.1: Variant Caller Comparison Histograms

- Mutect:
  - Recalibration appears to refine Mutect's variant calling precision across both Bowtie and BWA configurations.
  - The reduction in unique variants in Recal-Bowtie and Recal-BWA suggests a positive impact on accuracy.

  - Mutect demonstrates a relatively stable performance with both Bowtie and BWA, showing a moderate sensitivity to the choice of mapper.

- SomaticSniper:
  - Recalibration significantly refines SomaticSniper's variant calling precision across both Bowtie and BWA configurations.
  - The reduction in unique variants in Recal-Bowtie suggests a positive impact on mitigating false positives.
  - SomaticSniper is highly sensitive to the choice of mapper, with substantial differences in unique variants between Bowtie and BWA configurations.
  - BWA tends to result in a lower variant count, potentially indicating a more conservative calling behavior.

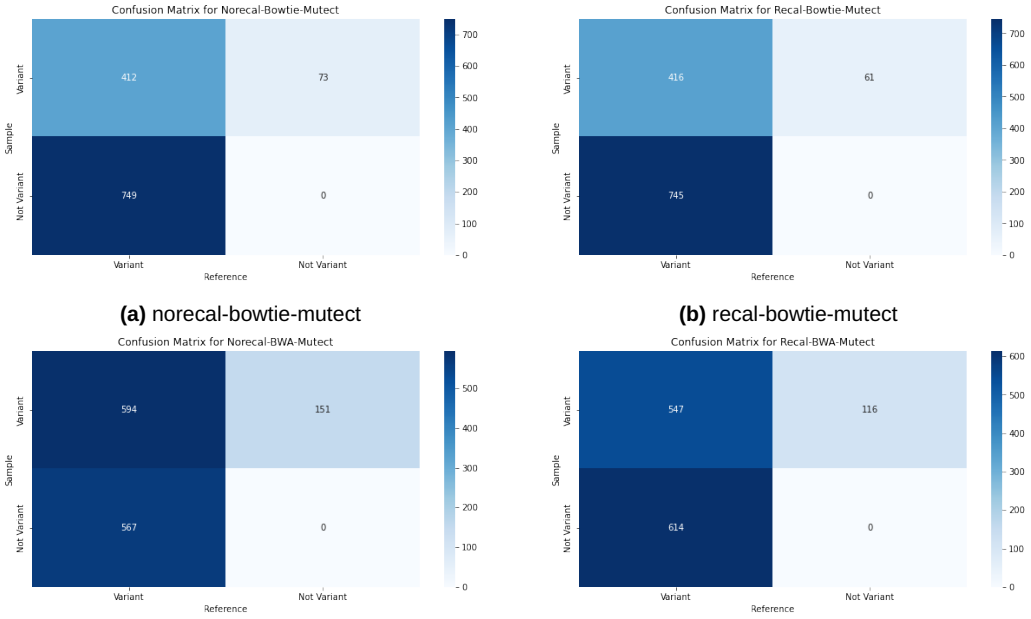
- Strelka:
  - Recalibration appears to refine Strelka's variant calling precision across both Bowtie and BWA configurations.
  - The reduction in unique variants in Recal-Bowtie and Recal-BWA suggests a positive impact on mitigating false positives.
  - Strelka is sensitive to the choice of mapper, with substantial differences in unique variants between Bowtie and BWA configurations.
  - BWA tends to result in a higher variant count, indicating a potential influence on sensitivity.

3.2. Confusion Matrices

Before analysing the confusion matrices of the pipelines, we have to clarify some metrics. Precision, Recall, F1 Score, and Accuracy are key metrics used to evaluate the performance of classification algorithms, including variant calling pipelines in next-generation sequencing (NGS) data analysis.

Firstly, Precision, also known as positive predictive value, is the ratio of true positive predictions to the total number of positive predictions made by the model. Precision answers the question, "Of all the instances predicted as positive, how many were actually positive?" High precision indicates that the model is making accurate positive predictions, minimizing false positives. The formula to calculate precision can be observed below.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$



(a) norecal-bowtie-mutect

(b) recal-bowtie-mutect

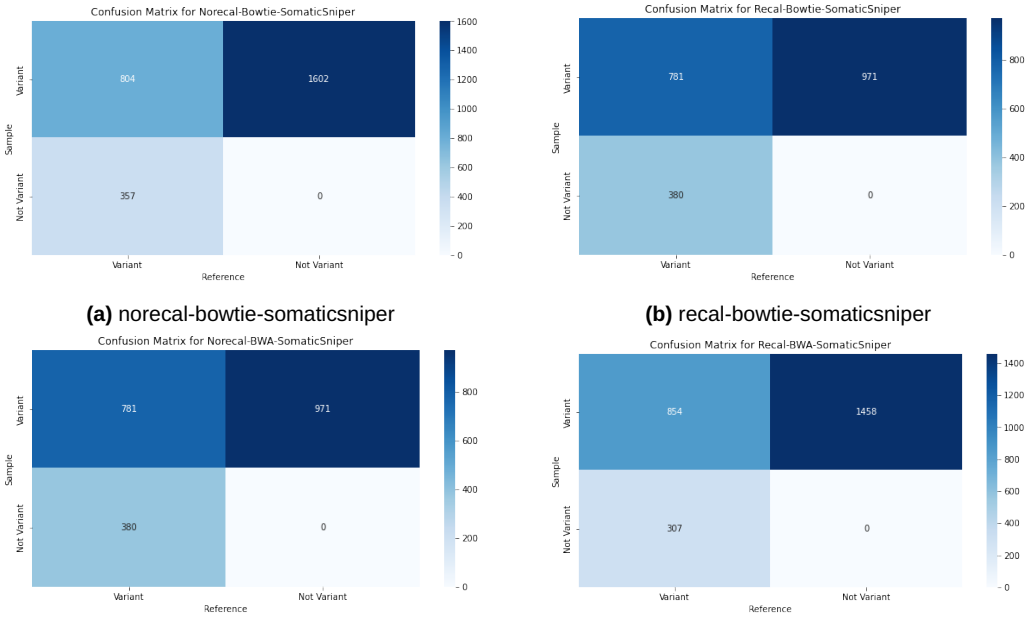
(c) norecal-bwa-mutect

(d) recal-bwa-mutect

Figure 3.2: Mutect Confusion Matrices

Recall on the other hand, also known as sensitivity or true positive rate, is the ratio of true positive predictions to the total number of actual positive instances in the dataset. It is the answer when we ask the question "Of all the actual positive instances, how many did the model correctly predict?" High recall indicates that the model is effective at capturing the positive instances, minimizing false negatives. Recall's formula goes like this:

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$



(a) norecal-bowtie-somaticsniper

(b) recal-bowtie-somaticsniper

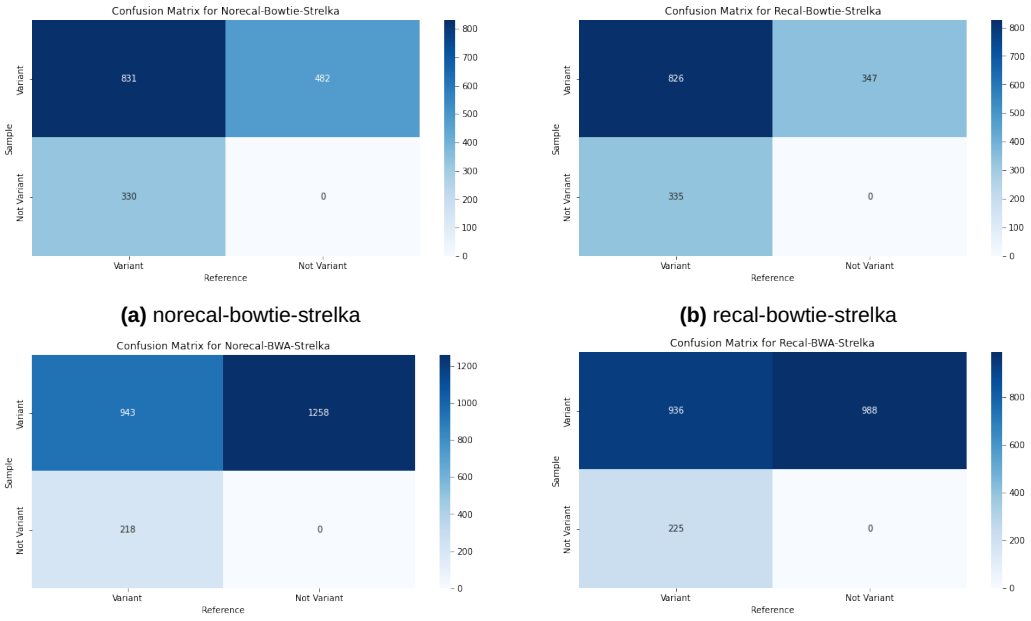
(c) norecal-bwa-somaticsniper

(d) recal-bwa-somaticsniper

Figure 3.3: SomaticSniper Confusion Matrices

The F1 score is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially in situations where there is an imbalance between the number of positive and negative instances and it is particularly useful when both precision and recall values are important, and there is a need to balance false positives and false negatives.

$$F1 = 2 * \frac{Precision * Recal}{Precision + Recal}$$



(a) norecal-bowtie-strelka

(b) recal-bowtie-strelka

(c) norecal-bwa-strelka

(d) recal-bwa-strelka

Figure 3.4: Strelka Confusion Matrices

Accuracy is the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances in the dataset. Accuracy provides an overall measure of the model's correctness. However, it may not be suitable when classes are imbalanced, as a high accuracy could be achieved by correctly predicting the majority class while ignoring the minority class.

$$F1 = \frac{TruePositives * TrueNegatives}{TotalInstances}$$

When we observe Figures 3.2, 3.3 and 3.4, it is seen that several patterns emerge. For the Mutect pipeline, the recalibrated versions consistently demonstrate improved performance, as indicated by higher true positives and lower false positives compared to their non-recalibrated counterparts. This trend suggests that recalibration significantly enhances the precision of variant calling in the Mutect context. Moving to the SomaticSniper pipeline, recalibration also demonstrates its positive impact, particularly in reducing false positives. Notably, the Bowtie mapping consistently yields higher false positives than BWA across all scenarios. Lastly, for the Strelka pipeline,

recalibration continues to play an important role, with the recalibrated pipelines outperforming their non-recalibrated counterparts. Additionally, the choice of mapping algorithm influences the performance, with BWA showing a favorable balance between true positives and false positives.

3.3. Performance Analysis of Pipelines

The performance metrics provide valuable insights into the impact of different pipeline configurations on variant calling operations. Here, we analyze the trends and effects observed across the specified pipelines:

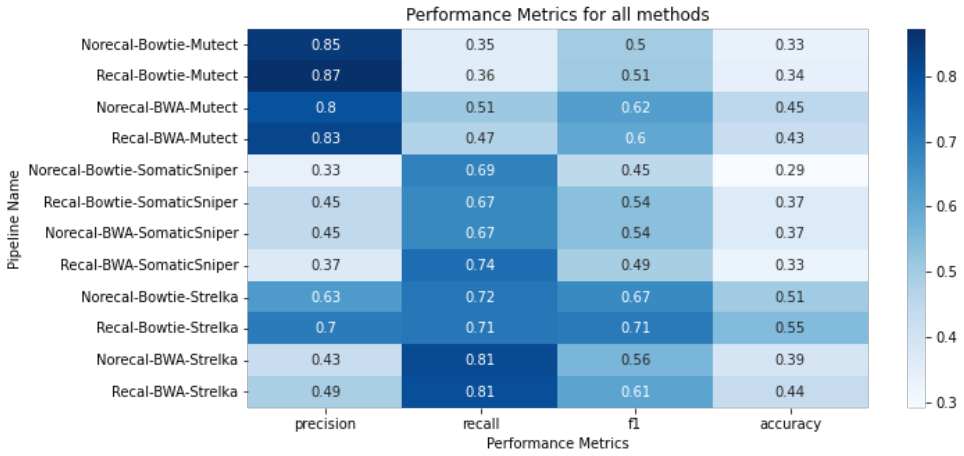


Figure 3.5: Precision, Recall, F1-Score and Accuracy Measurement of each pipeline

Mapping Algorithm Impact (BWA - Bowtie)

- Precision:
  - For Mutect and SomaticSniper, Bowtie consistently exhibits higher precision compared to BWA.
  - In Strelka, BWA configurations show a decrease in precision, suggesting a sensitivity to the choice of mapping.
- Recall:
  - BWA consistently outperforms Bowtie in terms of recall for all variant callers.
  - The difference is most prominent in Mutect, where BWA configurations exhibit significantly higher recall values.
- F1 Score:
  - F1 scores are generally balanced between Bowtie and BWA, with slight variations depending on the variant caller.
  - Mutect and SomaticSniper show more consistent F1 scores with BWA.

- Accuracy:
  - Accuracy is relatively consistent between Bowtie and BWA, with minor variations.
  - The impact of the choice of mapping is less pronounced compared to the recalibration effect.

Recalibraiton Effect

- Precision:
  - Across all tools, pipelines with recalibration tend to show slightly higher precision values.
  - Notable improvement is observed in SomaticSniper and Strelka configurations.
- Recall:
  - Recalibration generally contributes to increased recall, especially noticeable in Bowtie configurations.
  - Strelka pipelines benefit the most from recalibration in terms of recall.
- F1 Score:
  - Similar to precision and recall, recalibration tends to positively influence F1 scores.
  - Strelka benefits the most from recalibration, showing a significant improvement in F1 scores.
- Accuracy:
  - Recalibration contributes to a modest increase in accuracy across all pipelines.
  - Notable improvements are seen in Strelka configurations, aligning with trends observed in precision and recall.

The masurement of twelve distinct pipelines, each varying in base recalibration conditions, mapping algorithm choice (Bowtie or BWA), and variant caller (Mutect, SomaticSniper, Strelka), reveals some useful information. Recalibration emerges as an important factor, consistently enhancing precision, recall, and F1 scores across various tools. Particularly noteworthy is the substantial impact of recalibration on Strelka configurations, where significant improvements in precision and recall are observed. The choice between Bowtie and BWA introduces detailed differences, with BWA demonstrating higher recall and notable sensitivity in Strelka. These findings highlight

the complex interaction between pipeline configurations and variant calling metrics, showing the need for consideration of tool-specific requirements. The overall performance analysis underscores the significance of recalibration as a pivotal step in enhancing the accuracy and reliability of NGS data analyses, while also acknowledging the subtle yet influential role of mapping selection in shaping outcomes.

3.4. Heatmap Visualization

3.4.1. Jaccard Heatmap

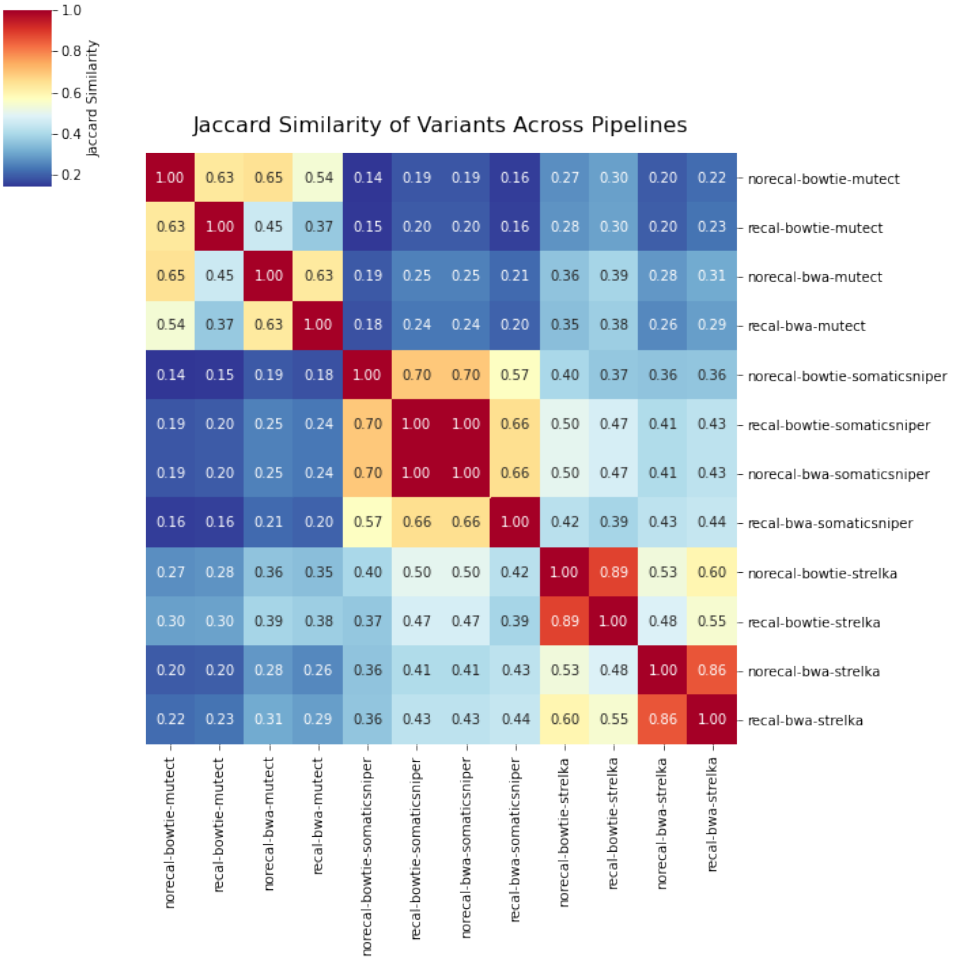


Figure 3.6: Jaccard Heatmap Representation of variant data

The Jaccard heatmap was generated with Matplotlib [2] and Sklearn [3] to visualize the similarity between different pipelines based on their variant profiles. The process involved transforming the variant data into a binary matrix, where each row represented a unique variant, and each column represented a different pipeline. The binary values in the matrix indicated the presence (1) or absence (0) of a variant in the indexed pipeline.

The Jaccard similarity was then computed between each pair of pipelines using the binary matrix. The Jaccard similarity coefficient measures the proportion of shared variants between two pipelines relative to the total number of distinct variants present in

both pipelines. A coefficient of 1 indicates identical variant profiles, while 0 indicates no shared variants.

The resulting Jaccard similarity matrix was visualized as a heatmap using seaborn's clustermap function [1]. The heatmap displayed pairwise Jaccard similarity values between pipelines, with higher values indicating greater similarity. The color intensity represented the strength of the similarity, with warmer colors indicating higher similarity and cooler colors indicating lower similarity.

As it is clearly seen in the heatmap, variant calling algorithms appear as the configuration that affects the similarity score the most. Thanks to the heat coloring, we can observe that 3 different variant calling algorithms form clusters in 3 different regions. It is also possible to obtain information from regions outside these clusters. The scores of SomaticSniper configured pipelines at the intersection with Strelka are significantly higher than the scores at the intersection with Mutect. From this we can conclude that SomaticSniper is more similar to Strelka.

3.4.2. Dice Similiarity Heatmap

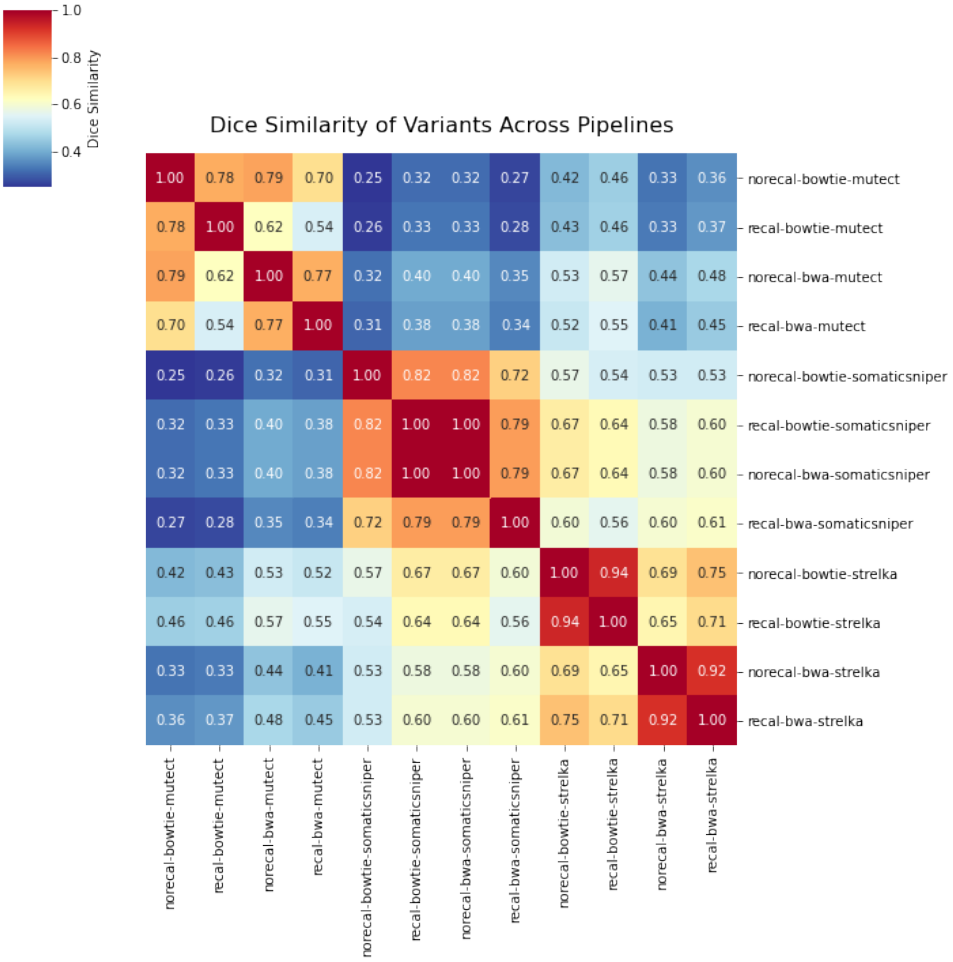


Figure 3.7: Dice Heatmap Representation of variant data

The heatmap depicting Dice similarity was generated with Matplotlib [2] and Sklearn [3] libraries to evaluate the resemblance among distinct pipelines based on their variant profiles. This involved the conversion of variant data into a binary matrix, where each row corresponded to a unique variant, and each column represented a specific pipeline. The binary values denoted whether a variant was present (1) or absent (0) in a given pipeline.

Dice similarity, renowned for its role in gauging the similarity between two sets, was computed for each pair of pipelines using the binary matrix. This similarity metric examines the overlap between sets, yielding a score ranging from 0 to 1, with 1 signifying complete similarity and 0 indicating no overlap.

The resultant Dice similarity matrix was visualized as a heatmap using seaborn's clustermap function [1]. In alignment with prior heatmaps, this visualization illustrated pairwise Dice similarity values between pipelines. Elevated values indicated heightened similarity, while lower values indicated reduced overlap. The color intensity on the heatmap conveyed the strength of similarity, with warmer hues representing higher similarity and cooler hues denoting lower similarity.

Dice heatmap tends to have higher values overall compared to Jaccard heatmap, as Dice penalizes unmatched elements more than Jaccard. Jaccard heatmap may shows sharper distinctions between areas of high and low similarity due to its focus on the ratio of shared elements. (Jha) When examining the formulas of the three coefficients—Jaccard, Overlap, and Dice—it becomes apparent that the Jaccard coefficient imposes a more significant penalty on differences compared to the overlap coefficient. Conversely, the Dice coefficient places a slightly higher emphasis on commonalities than differences.[4]

3.5. Principal Component Analysis

The PCA (Principal Component Analysis) was performed on the variants obtained from different pipelines to explore the underlying patterns and relationships within the data. First, the variant data from each pipeline output were organized into a matrix format, where rows represent unique variants, and columns represent different pipelines. Each cell in the matrix was assigned a binary value (0 or 1) based on the presence or absence of a variant in the corresponding pipeline.

Next, the matrix was transposed, resulting in a new matrix where rows represent different pipelines, and columns represent unique variants. This transposition was necessary for the PCA analysis, as it allows the identification of principal components that capture the maximum variance across different pipelines.

The PCA was then applied to this transposed matrix, reducing the dimensionality to two principal components for visualization. The resulting scatter plot displayed each pipeline as a point in a two-dimensional space, with the x and y axes representing the first and second principal components, respectively.

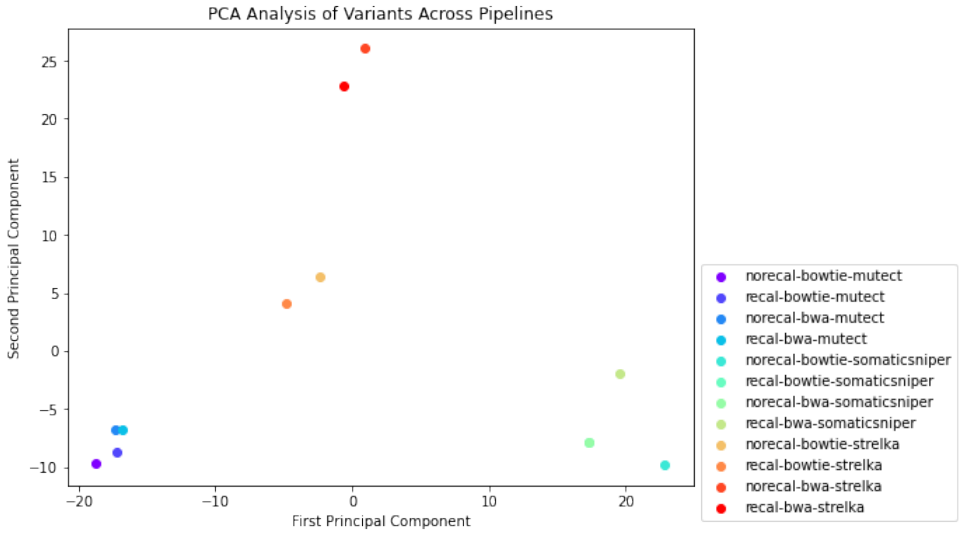


Figure 3.8: Principal Component Analysis

In the scatter plot, each pipeline was assigned a distinct color, providing a visual representation of how variants across different pipelines cluster in the reduced feature space. The distance between points indicates the degree of similarity or dissimilarity between pipelines based on their variant profiles.

- Analysis
  - As seen in the table, pipelines with similar structures tend to converge. The fact that the Mutect pipeline group is closer to each other than the other groups shows that the Mutect variant calling algorithm is more insensitive to the choices of base recalibration and mapping algorithms and gives similar results in different configurations.
  - We can also observe this in the SomaticSniper algorithm, but the increase in the distance between the group elements tells us that the similarity rate decreases.
  - As for the Strelka algorithm, we see that the change in the base recalibration setting does not lead to large similarity differences, but we observe that the change in the mapping algorithm has a great impact on the results and divides the group into two.

4. Discussion

The comprehensive analysis of variant calling pipelines using the Comparative Sequencing Analysis Platform (CoSAP) has provided valuable insights into the performance of different configurations. The exploration focused on Read Trimming, Read Mapping, Duplicate Removal-Base Calibration, Variant Calling, and Variant Annotation steps, employing various mappers (BWA, Bowtie) and variant callers (Mutect, SomaticSniper, Strelka).

The comparison of variant calls made by Mutect, SomaticSniper, and Strelka across different setups revealed distinctive patterns. Recalibration consistently enhanced precision and reduced false positives across all tools. Notably, Strelka showed sensitivity to mapping algorithms, with BWA resulting in higher variant counts, possibly indicating increased sensitivity. These findings underscore the impact of both recalibration and mapping choices on variant calling precision and accuracy.

The analysis of confusion matrices further emphasized the positive influence of recalibration on precision and recall. The Mutect pipeline demonstrated improved performance with recalibration, while SomaticSniper exhibited sensitivity to mapping choices. Strelka configurations showcased varying impacts, emphasizing the need for tool-specific considerations in pipeline design.

The examination of performance metrics highlighted some useful information. Recalibration consistently improved precision, recall, and F1 scores, with Strelka benefiting the most. Mapping algorithm choice influenced recall, with BWA generally outperforming Bowtie. The intricate interplay between recalibration and mapping emphasizes the need for a tailored approach based on specific tool requirements.

The PCA analysis provided a visual representation of variant profiles across different pipelines. Clustering patterns indicated that Mutect demonstrated higher insensitivity to configuration changes, while SomaticSniper and Strelka exhibited more nuanced responses. This insight reinforces the importance of considering tool-specific characteristics in pipeline optimization.

Future research endeavors could explore additional factors influencing variant calling, such as library preparation methods, sequencing platforms, and the impact of reference genome choices. Investigating the scalability of pipelines with larger datasets and diverse genomic regions could provide insights into real-world applicability. Moreover, fine-tuning pipeline parameters based on specific genomic contexts and experiment types could further enhance performance.

In conclusion, this study contributes to the understanding of variant calling pipeline performance, bringing light on the different effects of recalibration and mapping choices. The findings underscore the necessity of tailored approaches for different variant calling tools and provide a foundation for optimizing pipelines in diverse genomic analyses.

Bibliography

[1] Seaborn Development Team. “Seaborn: statistical data visualization.” [Online] Available: https://seaborn.pydata.org, Accessed: [18.12.2023].

[2] Matplotlib Development Team. “Matplotlib: Visualization with Python.” [Online] Available: https://matplotlib.org, Accessed: [16.12.2023].

[3] Scikit-learn. “Machine Learning in Python - scikit-learn.” [Online] Available: https://scikit-learn.org/stable/, Accessed: [18.12.2023].

[4] Jayant Jha, Similarity Coefficients: A Beginner’s Guide to Measuring String Similarity. ” [Online] Available: https://medium.com/@igniobydigitate/similarity-coefficients-a-beginners-guide-to-measuring-string-similarity-d84da77e Accessed: [20.12.2023]