International Journal of
Molecular Sciences

IMPACT FACTOR 4.9
Indexed in: PubMed
CITESCORE 8.1

Article

A Novel Affordable and Reliable Framework for Accurate Detection and Comprehensive Analysis of Somatic Mutations in Cancer

Rossano Atzeni, Matteo Massidda, Enrico Pieroni, Vincenzo Rallo, Massimo Pisu and Andrea Angius

Special Issue
Molecular Research of Multi-omics in Cancer

Edited by
Dr. Siddharth Pratap

MDPI

https://doi.org/10.3390/ijms25158044

International Journal of
Molecular Sciences

MDPI

Article

A Novel Affordable and Reliable Framework for Accurate Detection and Comprehensive Analysis of Somatic Mutations in Cancer

Rossano Atzeni [1,†], Matteo Massidda [2,†], Enrico Pieroni [1], Vincenzo Rallo [3], Massimo Pisu [1] and Andrea Angius [3,*]

[1] Center for Advanced Studies, Research and Development in Sardinia (CRS4), 09050 Pula, Italy; ratzeni@crs4.it (R.A.); ep@crs4.it (E.P.); massimo@crs4.it (M.P.)
[2] Department of Medical, Surgical and Experimental Sciences, University of Sassari, 07100 Sassari, Italy; mmassidda@uniss.it
[3] Istituto di Ricerca Genetica e Biomedica (IRGB), Consiglio Nazionale delle Ricerche (CNR), Cittadella Universitaria di Cagliari, 09042 Monserrato, Italy; vincenzo.rallo@irgb.cnr.it
* Correspondence: andrea.angius@cnr.it
† These authors contributed equally to this work.

Abstract: Accurate detection and analysis of somatic variants in cancer involve multiple third-party tools with complex dependencies and configurations, leading to laborious, error-prone, and time-consuming data conversions. This approach lacks accuracy, reproducibility, and portability, limiting clinical application. *Musta* was developed to address these issues as an end-to-end pipeline for detecting, classifying, and interpreting cancer mutations. *Musta* is based on a Python command-line tool designed to manage tumor-normal samples for precise somatic mutation analysis. The core is a Snakemake-based workflow that covers all key cancer genomics steps, including variant calling, mutational signature deconvolution, variant annotation, driver gene detection, pathway analysis, and tumor heterogeneity estimation. *Musta* is easy to install on any system via Docker, with a Makefile handling installation, configuration, and execution, allowing for full or partial pipeline runs. *Musta* has been validated at the CRS4-NGS Core facility and tested on large datasets from The Cancer Genome Atlas and the Beijing Institute of Genomics. *Musta* has proven robust and flexible for somatic variant analysis in cancer. It is user-friendly, requiring no specialized programming skills, and enables data processing with a single command line. Its reproducibility ensures consistent results across users following the same protocol.

Keywords: cancer; somatic mutations; mutational patterns; mutational signatures; somatic variant detection; machine learning; precision medicine

Citation: Atzeni, R.; Massidda, M.; Pieroni, E.; Rallo, V.; Pisu, M.; Angius, A. A Novel Affordable and Reliable Framework for Accurate Detection and Comprehensive Analysis of Somatic Mutations in Cancer. *Int. J. Mol. Sci.* **2024**, *25*, 8044. https://doi.org/10.3390/ijms25158044

Academic Editor: Gennady Verkhivker

Received: 10 June 2024
Revised: 11 July 2024
Accepted: 22 July 2024
Published: 24 July 2024

1. Introduction

The large amount of data coming from cancer genomes, via high-throughput next-generation sequencing (NGS) technology, has recently revolutionized the studies in oncology, enabling comprehensive analyses of somatic variants and addressing the need for affordable data analysis tools, oriented to their identification and interpretation [1]. In fact, both single nucleotide variants (SNVs) and small insertions and deletions (INDELs) play a critical role in the initiation and progression of cancer [2]. Unfortunately, the specific somatic mutations that drive tumor growth exhibit high variability across different cancer types, and even within individual tumors [3]. Hence, it is vital to accurately identify and interpret these mutations for comprehensive understanding of cancer genome characterization, clinical genotyping, and treatment strategies [4].

However, the accurate identification and interpretation of somatic mutations in cancer samples remain a major challenge due to the inherent complexity and heterogeneity of cancer genomes [5]. This complexity varies among different tumor types, and even among

patients with the same cancer type [6]. Additionally, detecting low allele frequency variants present in a small fraction of cancer cells is difficult using conventional sequencing methods [5]. Sequencing artifacts, such as errors during library preparation, sequencing, or data processing, can lead to false-positive or false-negative results, affecting the accuracy of somatic mutation detection [7]. Furthermore, cross-contamination between tumor and normal tissues during sample collection, processing, or sequencing can introduce spurious results. Another challenge is intratumor heterogeneity, which refers to the extensive genetic diversity among different tumor cells within the same tumor. Intratumor heterogeneity can arise due to clonal evolution or spatial heterogeneity, and can impact the accuracy of somatic mutation calling and structural and functional interpretation of mutations [5,6].

To address these challenges, researchers have developed various bioinformatics tools for processing and analyzing cancer genomic data, each with its strengths and limitations [8]. These tools typically encompass multiple analysis steps, including read alignment, variant calling, variant annotation, detection of driver genes, pathway analysis, assessment of tumor heterogeneity, and deconvolution of mutational signatures [8].

However, using third-party software for detecting and analyzing mutations in cancer poses further challenges and problems, such as inaccuracies, difficulties in reproducibility, and limited clinical applicability [8]. The complex dependency trees and configuration requirements of these tools can often exacerbate these issues. All these aspects highlight the necessity of user-friendly and streamlined pipelines that offer reproducible, reliable and accurate results. Developing such pipelines is crucial for standardizing the analysis of cancer genomic data and ensuring result reproducibility across different datasets and research groups [8].

To overcome these limitations, we present Mutation and Somatic Tumor Analysis (*Musta*): a novel, affordable, and reliable end-to-end pipeline for detecting, classifying, and interpreting somatic mutations in cancer. *Musta* is designed to provide a standardized framework that offers accurate detection and comprehensive analysis of somatic variants while being at the same time user-friendly and cost-effective.

2. Results

Musta is currently used for cancer sample data analysis at the CRS4-NGS Core. Its reliability has been extensively tested on published data from "The Cancer Genome Atlas" repository. Each cohort counts hundreds of samples in distinct patients. Furthermore, *Musta* was also tested on published data from the genome archive of Beijing Institute of Genomics, with 23 samples of liver cancer from the same patient [9].The performance evaluations of *Musta* were conducted on a Dell server with dual Intel Xeon Gold 6238R processors, each featuring 28 cores at 2.2 GHz, a 38.5 M cache, and 128 GB of RAM. The server, configured with a total storage capacity of 40 TB in a RAID 5 setup, provided ample resources for robust analyses. The server runs CentOS Linux release 7.9.2009, with Docker version 20.10.11 and GNU Make version 4.3. During evaluations, each job in the Snakemake workflow efficiently utilized 4 out of the 56 available cores on the server.

2.1. Evaluating Musta in a Hepatocellular Carcinoma (HCC) Dataset

In the initial evaluation of *Musta*, we focused on its performance in analyzing the HCC dataset. A set of 23 tumor biopsies, along with a tumor-adjacent matched normal sample (N1), were sequenced at an average depth of 74.4× [9]. The samples were systematically distributed across the tumor tissue slice, categorized into four quadrants (A–D), with a central sample (Z1). Furthermore, the 23 sequenced samples were evenly distributed within the tumor, with 12 samples positioned on the periphery and 11 samples on the inside (see Figure 1a in [9]).

2.1.1. Detection

To evaluate the performance of *Musta*'s Detection module on HCC datasets, we executed the *Musta* detect command with all variant callers enabled. The results were then

compared with the findings of the original study [9]. To provide a more comprehensive representation and facilitate interpretation, data from comparative results are summarized in plots and diagrams in Figures 1 and 2. Initially, a purely quantitative assessment is conducted by comparing the total numbers and counts of "pass" variants, along with the execution times for each variant caller and sample (Supplementary Table S1). Additionally, Supplementary Table S1 includes data on the amount of non-swapped physical and virtual memory used, the number of MB read and written, and CPU usage time.

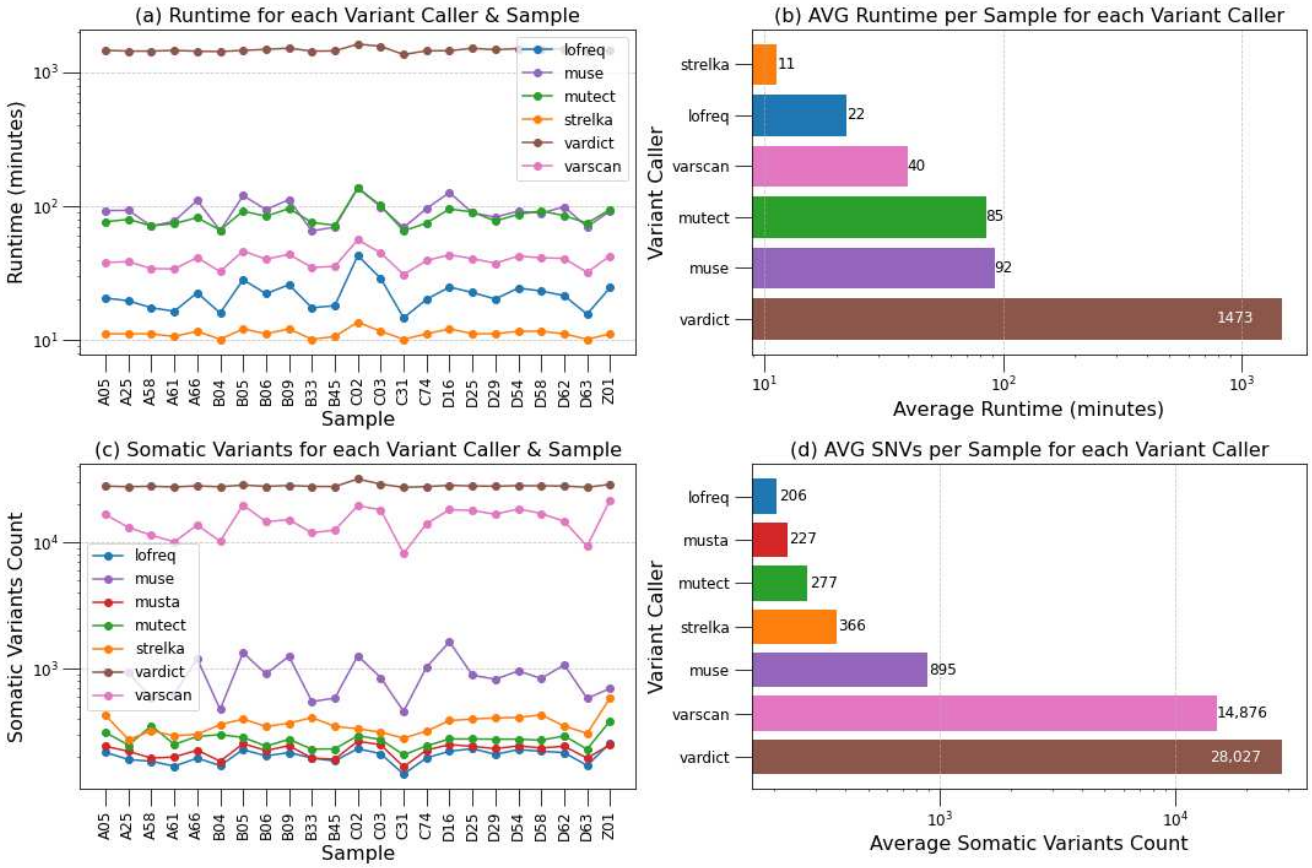

Figure 1. Performance evaluation of *Musta*'s detection module on HCC datasets. This figure provides a comprehensive comparison of the variant calling results from the *Musta* Detection module and the original study by [9]. (**a**,**b**) display the execution times for each variant caller across different samples: (**a**) highlighs the runtime of each variant caller for each sample while (**b**) shows the average runtime for a single sample. (**c**,**d**) illustrate the total counts of *pass* variants reported by each variant caller: (**c**) exhibits the number of variants called by each variant caller for each sample while (**d**) shows the average number of called variants in a single sample per variant caller.

In the overall variant caller analysis, distinctive characteristics emerge, as clearly depicted by the plots for runtime (Figure 1a,b) and the number of somatic variants (Figure 1c,d). LoFreq stands out for its consistency and reliable results, featuring quick runtimes but potentially limited sensitivity due to a lower count of reported variants. MuSe, despite variable runtimes, contributes to a comprehensive overview with its moderate to high number of reported variants. Mutect2 proves stable in runtimes, achieving a balance between precision and sensitivity with a moderate count of reported variants. Strelka2 excels in quick execution, offering a high quantity of variants but requiring careful filtering. VarDict, albeit slower, provides an extensive set of variants, emphasizing the trade-off between time and results. VarScan2 demonstrates versatility with moderate runtimes and a balance between precision and sensitivity, offering a well-rounded choice. The consensus strategy prioritizes variants identified by multiple callers, ensuring a more conservative

selection. However, when integrating permissive and restrictive callers, there is a potential drawback: the rejection of a significant portion of variants called by the permissive ones. It is important to note the variation in execution times: Strelka2, LoFreq, and VarScan2 complete their analysis in a few tens of minutes, Mutect2 and MuSe take about an hour and a half, and VarDict requires over 24 h. Excluding VarDict, the Detection module typically takes about 4 h to analyze a single sample.

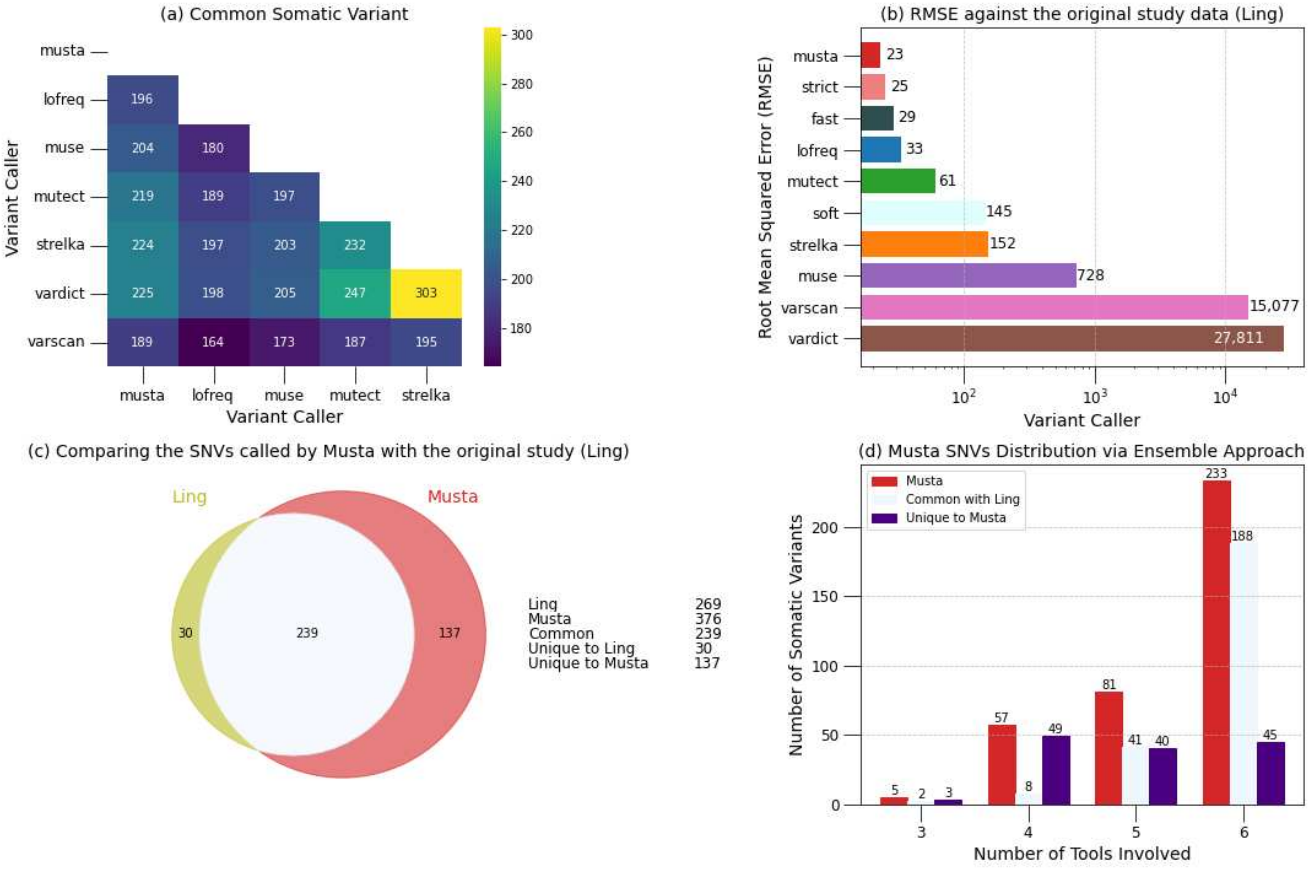

Figure 2. *Musta* against original study data (Ling [9]). This figure shows a comparison of the number of somatic variants validated by Ling and the results from *Musta*. (**a**) heatmap highlights common somatic variants between variant callers and their contribution in *Musta* results. (**b**) root mean square error (RMSE) for all variant callers against Ling results, demonstrating *Musta*'s precision. (**c**) Venn diagram highlighting the concordance between *Musta* and Ling analysis, showing nearly 90% overlap, with *Musta* identifying 137 unique variants and Ling 30. (**d**) underscores *Musta*'s robustness, showing that nearly 99% of somatic mutations were identified by at least four out of six variant callers, indicating the high quality and comprehensive coverage of *Musta*'s ensemble approach.

In general, individual variant callers share more mutations with *Musta* than with other variant callers, confirming *Musta*'s superiority as an ensemble approach (see Figure 2a and Supplementary Table S2). Mutect2 and Strelka2 are exceptions, sharing a significant portion of the identified somatic mutations. Additionally, VarDict demonstrates the highest concordance with Strelka2. Let us now examine the number of somatic variants per sample, identified and validated by [9], and the comparison with *Musta*'s results (Supplementary Table S3). For a comprehensive validation, and to demonstrate the efficacy of the ensemble approach, we also compare the results of individual variant callers and of combinations of them. From the calculation of the root mean square error (RMSE) for all variant callers, concerning Ling's results, it becomes apparent that *Musta* exhibits the lowest RMSE, indicating the highest precision (Figure 2b). Interestingly, *Musta*'s precision is enhanced when employing all variant callers compared to the combinations offered by

the options: –strict (run only Mutect2, LoFreq, Strelka2), –soft (VarScan2, VarDict, MuSe), and –fast (LoFreq, VarScan2, Strelka2). After this initial quantitative analysis, we turn to a qualitative analysis to check for agreement on the mutations detected. The Venn diagram in Figure 2c highlights the concordance between *Musta* and Ling analysis to be nearly 90%. From the same diagram analysis, it emerges that *Musta* identified 137 variants unseen by Ling analysis, while there are only 30 mutations identified in the original study not captured by *Musta*. The robustness and reliability of *Musta* are further underscored by Figure 2d, where it is evident that nearly 99% of the somatic mutations were identified by at least four out of six Variant Callers, indicating not only the comprehensive coverage provided by *Musta*, but also the high quality of its results. This highlights the solidity of its ensemble approach in capturing a wide range of somatic mutations, thus confirming its reliability in the analysis of cancer samples.

2.1.2. Classification

After identifying somatic mutations in tumor samples, the subsequent classification of these mutations is vital for understanding their biological significance and clinical implications, particularly in identifying driver mutations pivotal for cancer progression. *Musta* employs two variant annotation options: the Ensemble Variant Effect Predictor (VEP) and GATK's Funcotator. Their results are summarized in Figure 3 and Supplementary Table S4, which also include detailed metrics such as the amount of non-swapped physical and virtual memory used, the number of MB read and written, and CPU usage time, providing a comprehensive overview of the resource consumption.

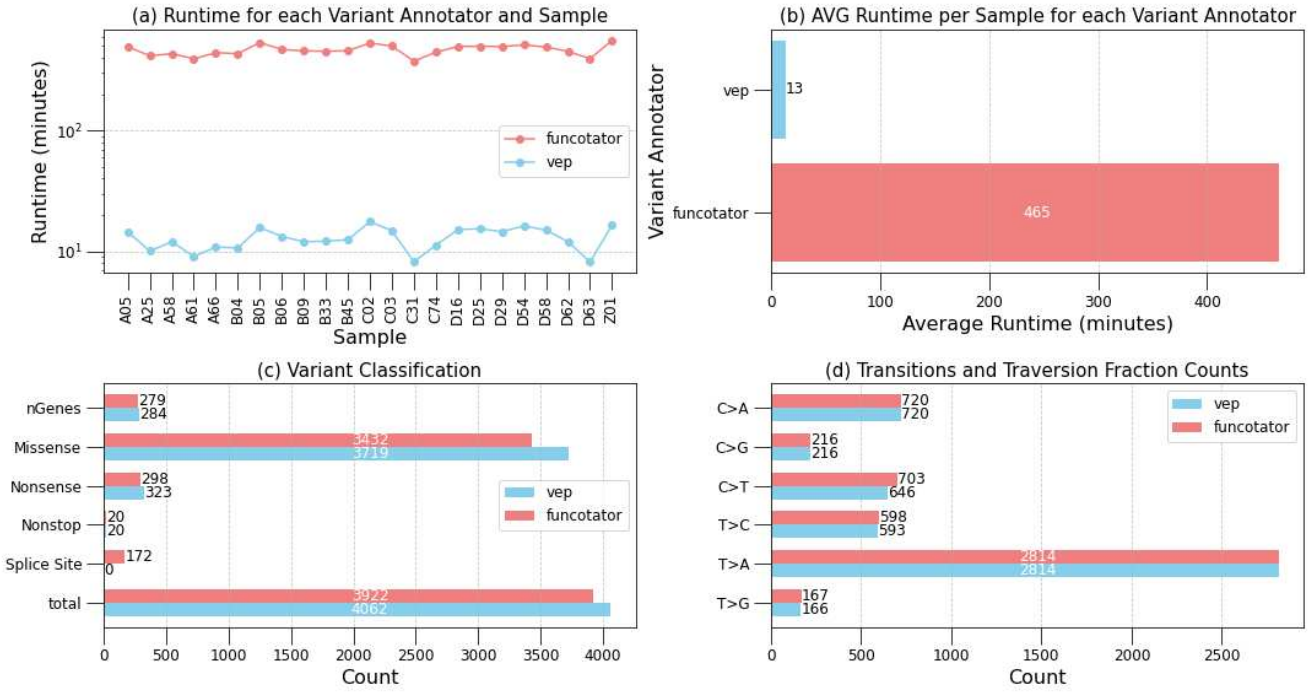

Figure 3. Performance evaluation of *Musta*'s classification module on HCC datasets. This figure summarizes the results and efficiency of the Ensemble Variant Effect Predictor (VEP) and GATK's Funcotator used in the *Musta* framework. (**a**,**b**): Runtime efficiency comparison. VEP annotates a sample in 15 min, while Funcotator takes over 7 h. (**c**,**d**): Quantitative outcomes. Funcotator identifies slightly fewer genes than VEP, but offers more detailed classifications per gene.

One striking difference between the two variant annotation tools is their runtime efficiency (see Figure 3a,b). VEP completes annotation for a single sample in approximately 15 min, demonstrating rapid processing. In contrast, Funcotator requires significantly more time, with an average runtime of over seven hours per sample. This substantial difference

may influence the choice of annotation tool based on specific analysis requirements and available computational resources. When comparing the quantitative classification outcomes (Figure 3c,d), Funcotator tends to identify slightly fewer genes compared to VEP. However, Funcotator provides a richer variety of classifications per gene, albeit with only a marginal difference. This suggests that while Funcotator may identify fewer genes overall, it offers more detailed classifications for each identified gene.

Furthermore, a qualitative analysis reveals that both tools classify the same set of genes as frequently mutated genes (FLAG), as shown in Figure 4a,b, indicating a high level of agreement in identifying genes relevant to hepatocellular carcinoma (HCC). These findings are consistent with previous studies in the literature, which have already identified these top genes as strongly correlated with hepatocellular carcinoma [10–18]. This convergence reinforces the reliability of both tools for cancer analysis. Given the high level of agreement and robustness observed, the choice between VEP and Funcotator ultimately depends on user preferences. Factors such as runtime requirements, computational resources, familiarity with each tool's functionalities, and ease of access to additional files may influence the decision-making process.

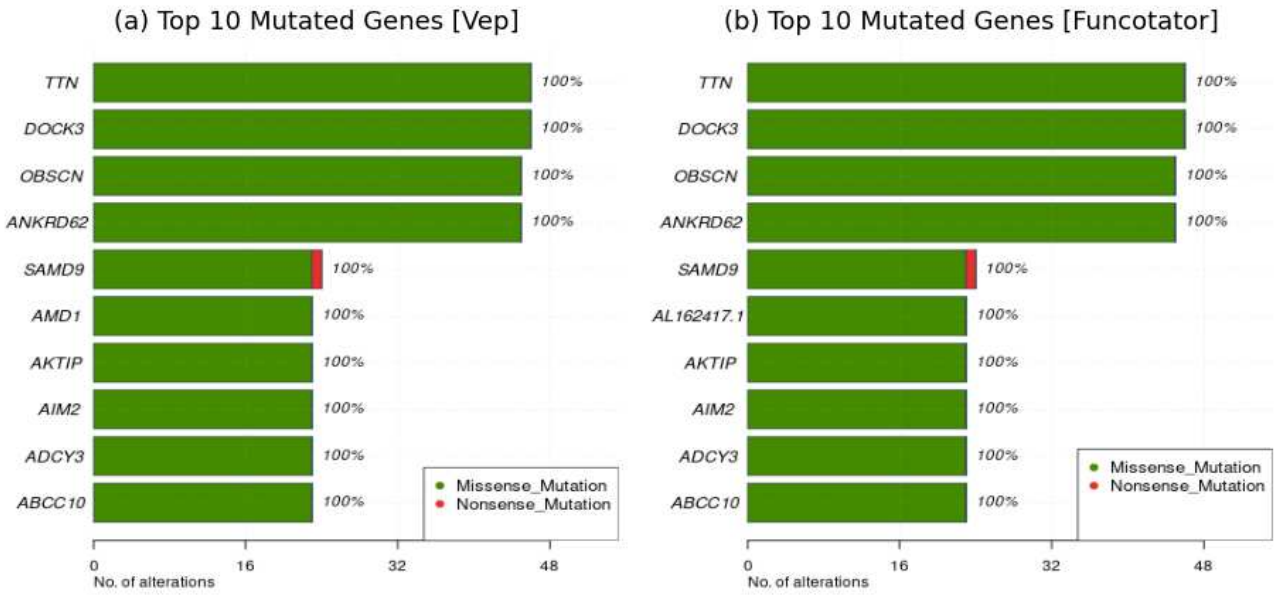

Figure 4. Comparison of frequently mutated genes (FLAG) classified by VEP and Funcotator. Both tools classify the same set of genes as frequently mutated genes (FLAG), indicating high agreement in identifying genes relevant to hepatocellular carcinoma (HCC).

2.1.3. Interpretation

In this section, the outcomes of the Interpretation module are examined (refer to Supplementary Table S5 for details on non-swapped physical and virtual memory usage, MB read and written, and CPU usage time), focusing on the comparative analysis between the Hepatocellular Carcinoma (HCC) and the Liver Hepatocellular Carcinoma (LIHC) datasets [19] obtained from "The Cancer Genome Atlas" repository. The aim is to ascertain the concordance and consistency of results across these datasets, notwithstanding inherent disparities in sample size and diversity.

The comparison outcomes are illustrated in Figures 5–7 However, before delving into the interpretation of the results, it is essential to acknowledge a significant inherent bias: the LIHC dataset comprises 365 samples from distinct patients, whereas the HCC dataset consists of only 23 samples from a single patient, representing a singular case of hepatocellular carcinoma. Consequently, it is expected that the LIHC results exhibit greater diversity and wider distribution compared to those of HCC.

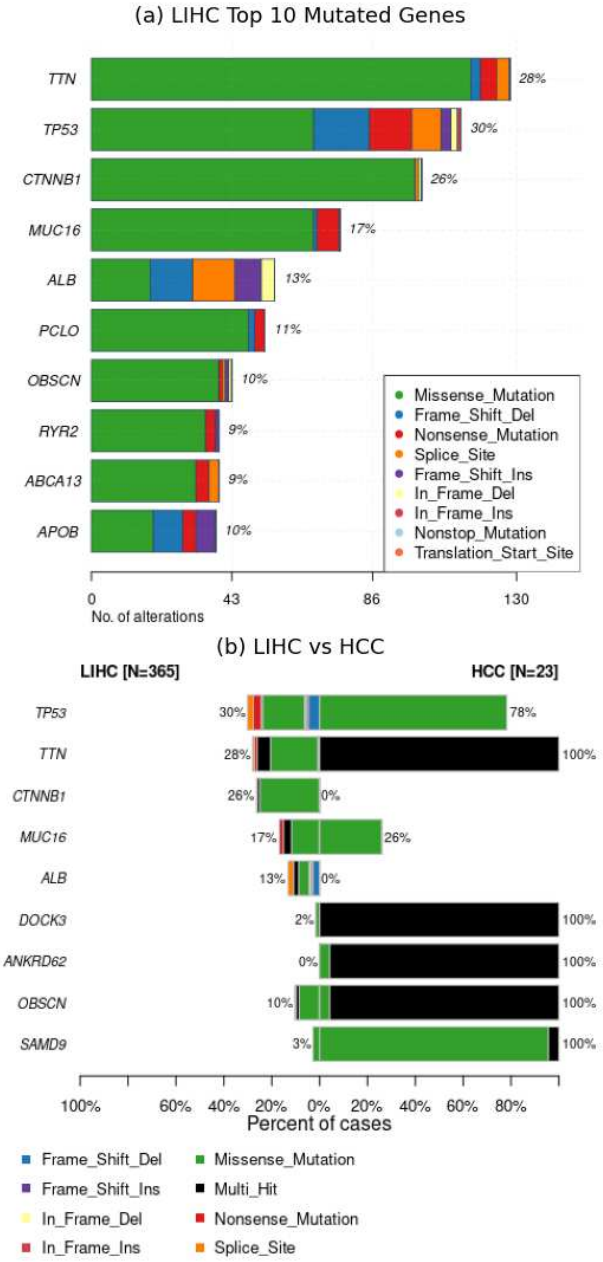

Figure 5. Performance evaluation of *Musta*'s interpretation module on HCC and LIHC datasets: frequently mutated genes. The (**a**) shows the Top 10 mutated genes in TCGA-LIHC dataset, while (**b**) illustrates a comparison between Top 10 mutated genes in TCGA-LIHC and HCC (Ling) datasets. *TTN* gene is identified as the most mutated gene in both datasets, aligning with literature on Hepatocellular Carcinoma. The HCC dataset shows mutations in all samples, while the LIHC dataset exhibits variable numbers of samples with mutations, reflecting greater diversity.

An initial positive observation is evident from Figure 5a,b, where the gene *TTN* emerges as the most mutated gene in both datasets. This finding aligns with existing literature [10], which strongly associates *TTN* mutations with Hepatocellular Carcinoma. It is notable that in the HCC dataset, genes are mutated across all samples, whereas in the LIHC dataset, the number of samples with mutated genes varies, reflecting the diversity inherent in a cohort of multiple patients.

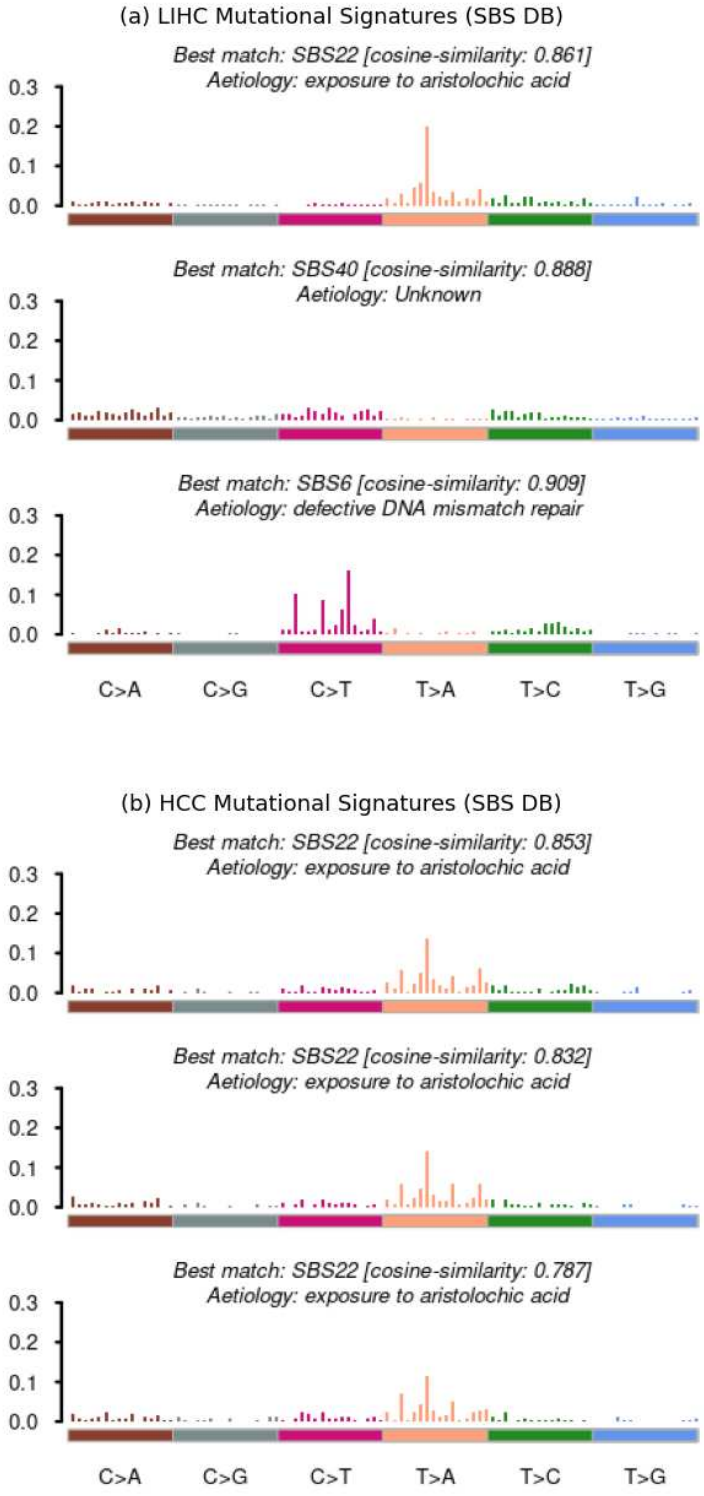

Figure 6. Performance evaluation of *Musta*'s interpretation module on HCC and LIHC datasets: mutational signatures from the SBS databases. Both datasets feature SBS22, linked to Hepatocellular Carcinoma, as the first signature.

The decisive evidence comes from the subsequent plots, which compare the Mutational Signatures extracted from the two datasets, from the SBS databases. In both datasets, the first signature is SBS22, associated with exposure to aristolochic acid and significantly

correlated with Hepatocellular Carcinoma (Figure 6a,b), As expected, three distinct signatures were extracted from the LIHC dataset, while only one was extracted from the HCC dataset.

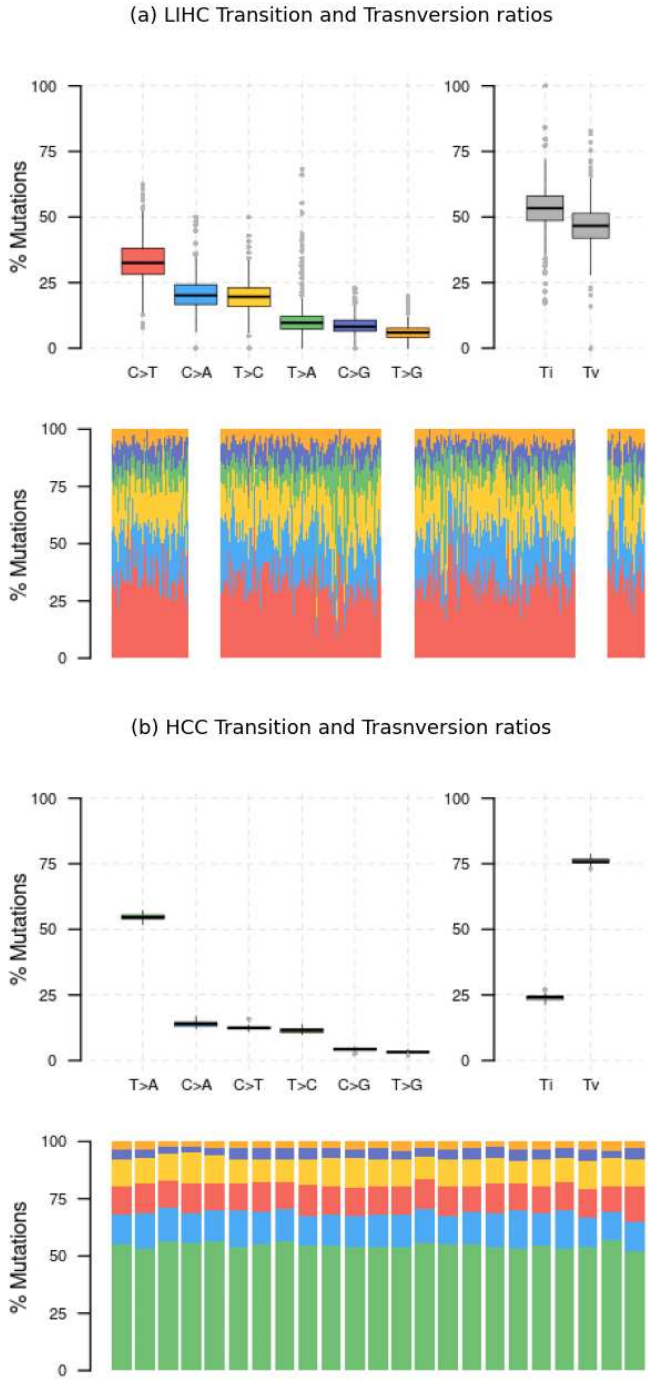

Figure 7. Performance evaluation of *Musta*'s interpretation module on HCC and LIHC datasets: transition and trasnversion ratios. Transition and transversion ratios are more uniformly distributed in the LIHC dataset. In contrast, the HCC dataset shows a predominance of T>A mutations.

This aligns with the observations in Figure 7a,b, where the LIHC dataset shows a more uniform distribution of Transition and Transversion ratios, whereas the HCC dataset exhibits a clear predominance of T>A mutations, overshadowing the others during signature extraction.

In conclusion, the comparative analysis reveals notable concordance between the HCC and LIHC datasets, despite differences in sample size and diversity. As we have often pointed out, and it is still worth reiterating, these findings enhance our understanding of the mutational landscape of Hepatocellular Carcinoma and underscore the importance of robust interpretation methods in genomic analysis.

2.2. Evaluating the Scalability and Portability of Musta

Musta leverages Docker, Snakemake, and Conda to ensure maximum scalability and portability. Tested on CentOS 7.9.2009, Ubuntu 20.04 and, later, Windows 11 Professional (with WSL 2 and Docker Desktop) and macOS, *Musta* demonstrates comprehensive cross-platform compatibility.

For scalability, *Musta* adopts a conservative approach to core management, always leaving one core free to prevent overloads. This requires a minimum dual-core system. Performance, memory usage, and storage utilization are influenced by the tools in the Detection and Classification modules and the number of samples. On an 4-core, 16 GB RAM system, the Detection module runs effectively with the '–fast' option, and the Classification module with VEP annotations. Systems with 16–128 GB RAM and 4–56 cores can incorporate more variant callers and annotators. *Musta*'s flexibility allows users to add variant caller results without regenerating VCFs, enhancing resource management and user convenience. However, intermediate files can occupy up to 50% of the total size of input BAM files, so adequate disk space is necessary.

Finally, an internet connection is essential for downloading packages, updates, and dependencies, ensuring smooth installation and operation of *Musta*.

3. Discussion

Musta, both in tests and routine analysis, has proven to be a robust and flexible pipeline for accurate detection and comprehensive analysis of somatic variants in cancer. Its ease of installation and setup enables users, even without specific skills in computational programming, to process cancer data using a single command line. Furthermore, *Musta* encourages and simplifies the definition of protocols, thus adhering to an analysis style that adopts best data processing practices [8], and thereby enabling the transfer and reproducibility of analysis and results.

Currently, *Musta* is being utilized for cancer sample data analysis at the CRS4-NGS Core. Its reliability has been extensively tested on the Liver Hepatocellular Carcinoma (LIHC) dataset [19] obtained from TCGA Research Network (https://www.cancer.gov/tcga, accessed on 21 July 2024). Each cohort comprises hundreds of samples from distinct patients. Additionally, *Musta* has been tested on published data from the genome archive of the Beijing Institute of Genomics, including 23 liver cancer samples from the same patient. All software tools incorporated into *Musta* have been carefully chosen after an in-depth literature review and practical testing, with the selection guided by the need to meet reliability, reproducibility, and confidence in results, while also ensuring ease of use. The Dockerized version of *Musta* ensures consistent and reproducible results across multiple platforms, making it a valuable tool for project collaborations. Moreover, *Musta* promotes the good practice of standardization, precise protocol definition, and seamless transferability to other users, ensuring consistency and reliability across analyses.

In summary, *Musta* represents a significant advancement in the field of somatic variant analysis in cancer, offering a reliable, flexible, and easily reproducible approach for researchers engaged in precision oncology research.

4. Materials and Methods

Musta is a comprehensive end-to-end pipeline developed to streamline the detection, classification, and interpretation of somatic mutations in cancer samples. *Musta*'s foundation is a user-friendly Python-based command-line tool that easily handles matched tumor-normal samples and that simplifies the analysis process for researchers, regardless

of their programming expertise. Researchers can initiate the entire analysis by executing a single command line, making *Musta* accessible and efficient even for those without specific programming skills.

4.1. Overview

The *Musta* framework efficiently organizes cancer sample processing tools into three distinct analysis modules: detection, classification, and interpretation (Figure 8). Researchers have the flexibility to run each module independently, or combine all three modules for a more comprehensive analysis workflow. The *Musta* framework is designed with a layered architecture (Figure 9): the core is a Snakemake-based workflow [20], encapsulated in a Python framework and running in a Docker container [21]. A user-friendly Command-Line Interface (CLI) enables users to issue commands and provide input data.

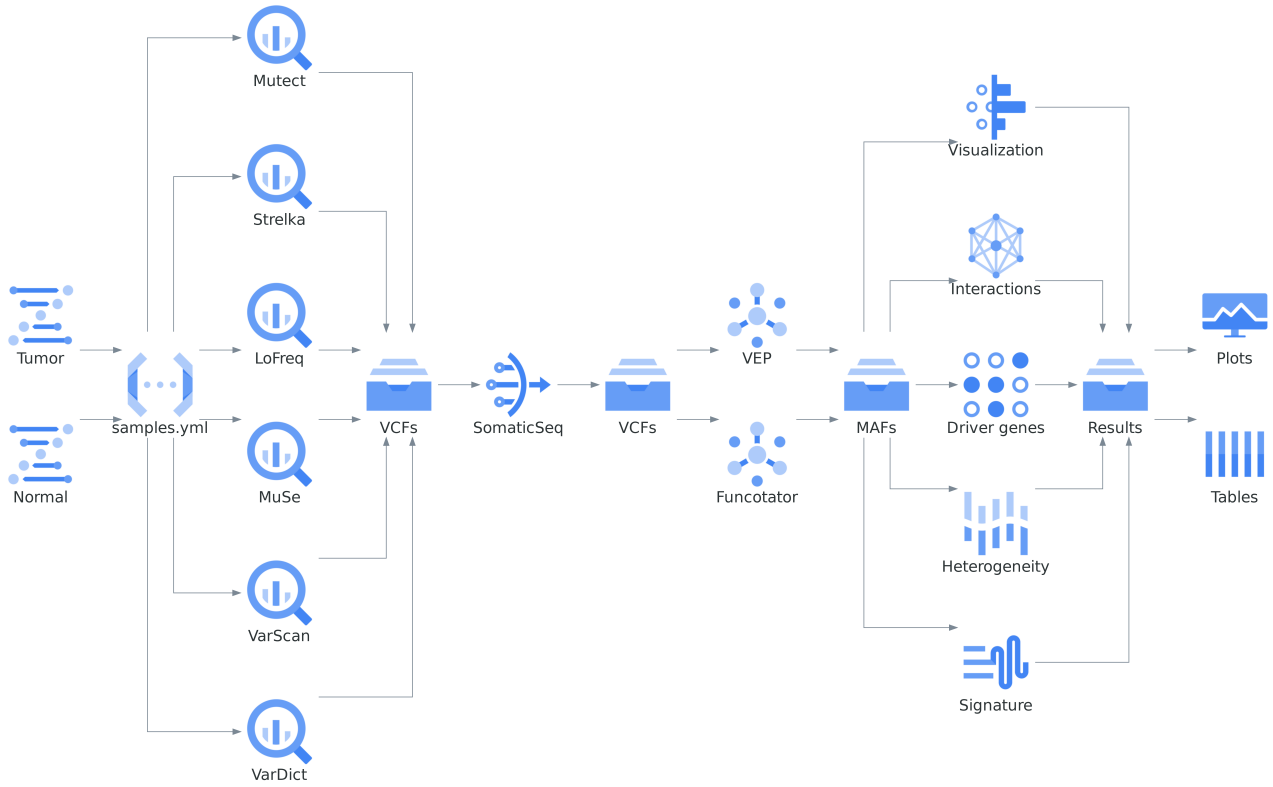

Figure 8. Overview of the *Musta* framework for cancer sample analysis. This figure illustrates the workflow of the *Musta* framework, which efficiently organizes cancer sample processing tools into three distinct analysis modules: detection, classification, and interpretation. The process begins with input BAM files, which are detailed in the samples.yaml file. Each paired normal and tumor BAM file is sent to one of the six variant callers in the detection module. The VCF files generated by each variant caller are then processed by the Ensemble step (SomaticSeq), which produces a consensus VCF. This consensus VCF is subsequently passed to the Classification module, where two Variant Annotators (VEP and Funcotator) generate an annotated MAF file. This annotated MAF file serves as the input for the final step, the Interpretation module, which generates plots and tables to facilitate data analysis and visualization.

The source code and the latest version of *Musta* framework is freely available at https://github.com/next-crs4/musta (accessed on 21 July 2024). A simple Makefile bootstraps *Musta*, taking care of the installation, configuration and running modules and allowing the execution of the entire pipeline or any individual module depending on the starting data. The detection module accepts BAM files as input, the classification module works with VCF files and the interpretation module accepts MAF files.

By running `make bootstrap`, the *Musta* Docker image is built and the executable file `musta` is conveniently linked in the user's bin path. With this setup, you can easily access *Musta* commands using the `musta` executable. The command `musta --help` will show a brief usage help.

The basic structure of the *Musta* command is as follows:

```
musta COMMAND --workdir WORKING-DIR --samples-file  SAMPLES-FILE [options]
```

Here is an overview of these components:

- `COMMAND`: This can be one of the following: `detect`, `classify`, or `interpret`. It selects the specific module the user wants to run. The command `musta COMMAND --help` will show a brief usage help for each command.
- `--workdir WORKING-DIR`: This parameter designates the working directory, which is the destination folder for analysis. This is the folder where (i) the Snakemake pipeline will be deployed, (ii) analysis logs will be stored, and (iii) all analysis outputs will be generated.
- `--samples-file SAMPLES-FILE`: this parameter points to a YAML file that lists the datasets the user wants to analyze.

For detailed usage instructions and user options, including the complete directory structure of the destination folder and the required YAML structure for the datasets file, please refer to the *Musta* documentation available at https://next-crs4.github.io/musta (accessed on 21 July 2024).

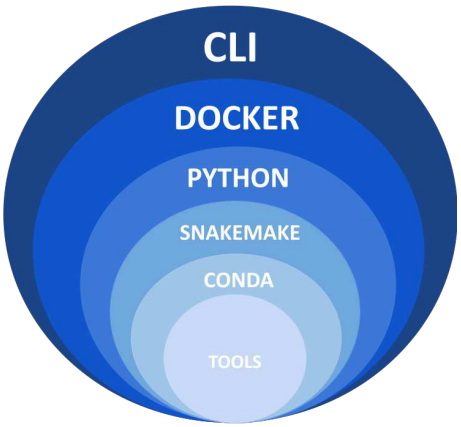

Figure 9. Layered architecture of the *Musta* framework. The *Musta* framework is built with a layered architecture, ensuring efficient and organized processing of cancer samples. At its core is a Snakemake-based workflow (version 7.15), encapsulated within a Python framework (version 3.8) and executed in a Docker container (version 20 and later). The Snakemake rules instantiate the necessary Conda (version 4.12) environments to run the individual tools, ensuring compatibility and reproducibility. Users interact with the system through a user-friendly Command-Line Interface (CLI), enabling easy command execution and data input.

4.1.1. Snakemake Core

At the heart of *Musta*'s architecture resides a Snakemake-based workflow. Snakemake, a powerful and versatile workflow content manager, offers modularity, scalability, and reproducibility. Within the Snakemake environment, the workflow is orchestrated through a set of rules that efficiently connect input datasets to their corresponding outputs. Snakemake smartly schedules rules, ensuring they run only when their necessary input files are available. It can also handle multiple rules simultaneously, making it scalable on different systems, allowing users to efficiently run the workflow on local machines, including high-performance computing clusters and cloud-based platforms. Furthermore, Snakemake exhibits resilience by offering error recovery capabilities. In the event of an

interruption during pipeline execution due to an error, Snakemake identifies missing output files, empowering users to resume the execution of failed jobs from the last available correct results, once the issues are resolved. Another notable advantage of Snakemake is its ability to manage software dependencies using Conda. This implies that Snakemake can create dedicated independent virtual environments in which to execute analysis pipelines, ensuring that all required software dependencies are present and in the correct versions. In the context of Snakemake, the *Musta* workflow consists of two core files:

- `Snakefile`: This file represents the workflow's core. It contains all the rules and commands that define the sequence of tasks and dependencies within the pipeline. Each rule specifies how to create output files from input files or other rules, and may invoke commands, scripts, or generate output directly. The Snakefile essentially orchestrates the entire analysis process.
- `config.yaml`: The config.yaml file complements the Snakefile by providing essential input files and parameters for the workflow. It specifies the data sources, settings, and configurations needed for *Musta* to execute the analysis correctly. This file helps to configure the workflow according to the user's specific requirements and data. Furthermore, through boolean flags in the config.yaml, allow users to specify which module to execute and which tools to enable.

The *Musta* Snakemake workflow is freely available at https://github.com/solida-core/musta (accessed on 21 July 2024). Users who are experienced with Snakemake can run the workflow independently, outside of the *Musta* framework, by editing the `Snakefile` and the `config.yaml` and running Snakemake commands.

4.1.2. Python framework

Through the integration of the Snakemake API library [22], the Python tool serves as the conductor of the Snakemake pipeline. It effectively manages configuration, orchestrates execution, and interprets user commands received through the CLI. In the following is reported an overview of the Python tool's functionality:

- Pipeline download. Initially, it downloads the pipeline into the working directory (`WORKING-DIR`).
- Environment configuration. The tool sets up internal paths within the Docker container environment.
- Edit `samples.yaml`. It edits the `samples.yaml` file, replacing user-local paths with Docker volume paths.
- Edit `config.yaml`. It manipulates the `config.yaml` file to control execution by toggling module and tool flags and constructing the workflow of rules.
- Workflow execution. Finally, this tool initiates the Snakemake workflow, ensuring the execution of the defined tasks.

In summary, the Python tool acts as the orchestrator of the Snakemake pipeline, configuring the environment, handling file paths, and executing the workflow based on user-defined parameters and commands received through the CLI.

4.1.3. Docker Container

Musta is conceived for an easy installation on any computational platform and any operating system through the Docker platform. Within the *Musta* architecture, the Docker container acts as a boundary layer, separating the outer CLI layer from the inner Python framework. This segregation allows a clear distinction between user interaction and core functionality. Key to this setup is the use of Docker volumes. Both the input data and the working directory are mounted as Docker volumes, providing access to the Python framework inside the container. Importantly, this arrangement is entirely transparent to the user, who will continue to work with familiar local file paths. This approach simplifies data management and ensures that results are readily accessible for further analysis. As previously mentioned, the `make bootstrap` command facilitates the creation of the *Musta*

Docker image. Users can confirm the existence of this image by executing the `docker image` command. For users familiar with Docker, direct access to the *Musta* Python tool is possible using a command structured as follows:

```
docker run [docker volumes options] musta:Dockerfile musta [musta option]
```

However, managing Docker volume mounts manually can be cumbersome and time-consuming. To streamline this operation, we have developed a user-friendly CLI that resides at a higher layer in the *Musta* framework.

4.1.4. Command Line Interface

The Command Line Interface (CLI) is based on a Bash script and aims to provide a seamless user experience by abstracting the presence of the Docker container, allowing users to interface directly with the Python tool. Here is a breakdown of the CLI's functionality:

- Input file and dataset check. The CLI performs an initial check on input files and datasets. It ensures that they exist, are in the correct format, and that all required accessory files are present. For example, it verifies that BAM files are indexed (with corresponding BAI files for each BAM), checks that VCF files are in compressed (GZ) format and indexed, and confirms the completeness of reference files.
- Constructing the Docker command. After verifying input files, the CLI constructs the Docker command. For each input file, volumes are mounted in Docker, and the list of arguments for the *Musta* Python tool is assembled. Finally, the Docker RUN command is prepared.

The CLI streamlines the process by handling these tasks automatically, making it easier for users to work with *Musta* and Docker without the need for detailed Docker knowledge. It simplifies the interaction with the *Musta* Python tool, allowing users to focus on their analysis rather than Docker container management.

4.2. Workflow Modules

The workflow for detection and analysis of somatic mutations in cancer usually begins with the preprocessing of raw sequencing data and culminates in the identification of somatic mutations. The goal is to generate a comprehensive and annotated list of mutations that facilitates informed clinical decision-making [8]. Although there is currently no universally accepted bioinformatics pipeline or set of tools for cancer analysis, several common elaboration steps are typically required and widely accepted. Downstream analysis includes, among others, driver gene identification, pathway analysis, deconvolution of mutational signature analysis, and estimation of tumor heterogeneity [5,23].

The preprocessing step aims to remove low-quality reads, adapter sequences, and other artifacts from the raw sequencing data, because overall they could affect downstream analyses. This step involves quality control, trimming, filtering, and alignment to a reference genome [24–27]. To simplify *Musta*, we have omitted the preprocessing step from the pipeline. Users are encouraged to generate their preprocessed BAMs, following the GATK Best Practices [27]. For user convenience, we recommend our dedicated Snakemake pipeline available at https://github.com/solida-core/dima (accessed on 21 July 2024). This resource will guide users through the necessary steps for preprocessing their data effectively.

4.2.1. Detection

The process of identifying somatic mutations in cancer begins with the comparison of aligned tumor sequencing data to a corresponding normal or matched reference dataset. This comparison makes it possible to distinguish between mutations that are tumor-specific (somatic mutations) and those present in the patient's germline (germline mutations). Variants are subsequently filtered according to several criteria such as read depth, quality score, allele frequency and functional annotation, to remove false positives and retain true positives while minimizing false negatives [28].

The landscape of available somatic variant detection tools is very broad, including both commercial and open-source solutions [29–33]. However, the concordance among these tools about variant callers is often low [34].Dramatically, different somatic variant callers may yield divergent results when applied to the same dataset. This is because each caller has its own strengths and limitations [29–33], thus resulting in such a low concordance among them. Consequently, identifying a single best variant caller that can consistently outperform others on different datasets is impractical [31,32]. To address this issue, ensemble approaches have been introduced to harmonize the results generated by multiple somatic variant callers [35–38]. To create an effective ensemble approach for somatic mutation detection [39], two key issues need to be addressed:

1. Selecting diverse and accurate component callers. Choosing an optimal number of component callers while ensuring diversity is crucial. Base learners must balance high accuracy with diversity to build a robust ensemble [40,41]. Diversity is essential because the benefit of an ensemble diminishes if all callers perform similarly. Conversely, too much diversity can lead to contradictory results, so a balanced selection of diverse component callers is essential.
2. Combining individual caller results:
    (a) Simple approaches like majority voting [42] and consensus approaches [35,43] rely on majority decisions or consensus among individual caller results. However, these do not take into account the quality of each caller's results, although weighted approaches are possible.
    (b) Complex machine learning-based methods such as stacking, Bayesian approaches, decision trees, and deep learning [44–49] leverage prediction results or metrics from individual callers as input features. Machine learning algorithms are used to combine these features, offering increased robustness against noise and errors. However, these methods often demand more computational resources and may be less interpretable.

In summary, the selection of diverse and accurate component callers, as well as the choice between simple and complex ensemble methods, plays a key role in improving somatic mutation detection accuracy while taking into account computational complexity and interpretability. After an initial in-depth literature review, practical testing, and screening, we selected six commonly used somatic variant callers: MuTect2 (v4.3.0) [50], VarScan2 (v2.4.4) [51], VarDict (v2019.06.04) [52], Strelka2 (v2.9.10) [53], LoFreq (v2.1.5) [54], and MuSE (v1.0) [55]. To improve the sensitivity and specificity of somatic mutation detection by leveraging the strengths of each caller, we selected SomaticSeq (v3.8.0) [47] as the ensemble approach for the *Musta* framework.

SomaticSeq integrates the VCF outputs from these six variant callers and processes them to produce a single consensus VCF, thus providing a more accurate and reliable set of somatic mutations. Compared to other ensemble and consensus tools—such as SomaticCombiner [56], NeoMutate [46], BAYSIC [45] and Moss [57]—SomaticSeq stands out for its consistent updates and maintenance, as well as its unique ability to accept input not only from all six variant callers selected for the *Musta* framework, but also from JointSNVMix [58], SomaticSniper [59], and Scalpel [60]. Additionally, it offers the flexibility to input any arbitrary VCF file(s) from caller(s) that we did not explicitly incorporate, paving the way for future improvements to *Musta*. These characteristics contribute to a richer, more flexible, and user-adaptive pipeline.

While technically feasible to develop a pipeline based on the other aforementioned ensemble and consensus tools, such an approach presents challenges that could compromise the universality and accessibility we aimed to achieve with *Musta*, without offering significant advantages. Each of these other tools has strengths and relevance to specific aspects of variant calling. For example, SomaticCombiner integrates a new variant allelic frequency (VAF) adaptive majority voting approach, which maintains sensitive detection for variants with low VAFs. NeoMutate incorporates seven supervised machine learning

(ML) algorithms to exploit the strengths of multiple variant callers using a non-redundant set of biological and sequence features. BAYSIC performs an unsupervised, fully Bayesian latent class analysis to estimate false positive and false negative error rates for each input method. Moss, in addition to VCF files, also takes the original BAM files of the tumor and normal samples as input. However, the downstream analysis flexibility achieved with SomaticSeq is far more effective for a general approach to the problem. Overall, SomaticSeq's adaptability and comprehensive input acceptance make it the superior choice for developing a robust and versatile variant calling pipeline, enhancing both the utility and ease of use for researchers.

SomaticSeq uses an ensemble strategy based on machine learning to integrate the individual caller results. It employs a random forest classifier that takes into account various features extracted from each variant caller's output, such as allelic depths, base quality scores, and mapping qualities. These features are used to train the classifier to distinguish true somatic mutations from false positives. The trained classifier then combines the predictions of the component callers and generates a final set of high-confidence somatic mutations. SomaticSeq into our analysis pipeline, will improve the reliability and completeness of somatic mutation analysis in cancer genomes.

In the context of Snakemake workflow, the Detection module involves rules that, starting from BAM files, automatically performs variant calling in tumor-normal matched mode, depending on provided input files for each sample. The rules are responsible for efficiently preparing the input files for each variant caller selected, executing the variant calling and the ensembles processes and retrieving the resulting VCF files. This includes handling tasks such as format conversions and file compression and decompression, as needed, and the indexing of the resulting VCF files for further downstream analysis. The basic structure of the *Musta* command to invoke Detection module is:

```
musta detect --workdir WORKING-DIR --samples-file  SAMPLES-FILE [options]
```

By default, the detection module within *Musta* is configured to perform the variant calling using all six of the mentioned variant callers. However, *Musta* provides users with the flexibility to customize their analysis according to their specific needs in the following ways:

- Excluding specific variant callers. If users wish, *Musta* allows specific variant callers to be excluded from the analysis. This customization ensures that only the desired variant callers are effectively applied to the data.
- Selecting preset combinations. Alternatively, users can choose from preset combinations of variant callers. *Musta* offers predefined sets of variant callers that have been optimized for specific analysis scenarios:
  `--strict` run only restrictive variant callers: Mutect2, LoFreq, Strelka2.
  `--soft` run only permissive variant callers: VarScan2, VarDict, MuSe.
  `--fast` run only fast variant callers: LoFreq, VarScan2, Strelka2, MuSe.

4.2.2. Classification

Following the variant calling process, where somatic mutations are identified, it is imperative to annotate these variants with functional information to gain insights into their potential impact on the genome. At a fundamental level, gene annotation is performed to determine whether the variant affects the protein-coding sequence of a gene. It specifically identifies whether the variant is synonymous or non-synonymous, and assesses its impact on splice sites and other specific genomic regions. This initial gene-level annotation is crucial for understanding the potential functional consequences of the variant itself. To enhance the annotation process, the identified variants are compared and annotated using established and widely used databases, such as dbSNP [61], gnomAD [62], ClinVar [63], and COSMIC [64]. These databases house a vast repository of curated genomic information, enabling researchers to determine whether a particular variant has been previously reported in the germline or has been associated with cancer-related genomic changes. This

comparison is critical in assessing the significance of the variant and its potential relevance to the specific tumor or tumor-associated disease under study.

To streamline the annotation process, researchers have several widely used and freely available annotation tools at their disposal. These tools, such as the Ensemble Variant Effect Predictor (VEP) [65], ANNOVAR [66], SnpEff [67], and Funcotator from GATK, play a key role in annotating variants. They use diverse annotation algorithms and leverage comprehensive genomic resources to provide detailed functional annotations for the identified variants. With the help of these tools, researchers can efficiently annotate large-scale datasets, taking into account variant type, location, and potential functional impact. In a comparative study [68], these variant annotation tools were evaluated using a ground-truth set of 298 variants from a medical exome database. Notably, VEP exhibited the highest accuracy, annotating 297 variants with HGVS nomenclature. In contrast, ANNOVAR showed the highest discrepancy, with 20 variants showing inconsistencies. Another study [69] reported a robust 92.6% concordance (100/108 variants) between SnpEff and VEP.

Based on the findings of these comparative studies, we have selected VEP (v106) as the primary variant annotator for the *Musta* pipeline. However, to enhance the flexibility and robustness of our pipeline, we recognize the unique capabilities and advantages of other annotation tools. Therefore, we have integrated Funcotator (v4.3.0) as an option within the pipeline, allowing users to choose and compare annotations generated by both VEP and Funcotator. This approach empowers users to tailor their annotation strategy to their specific research needs and preferences, ensuring that *Musta* can handle a variety of datasets with different characteristics. Additionally, we are committed to further expanding the annotation capabilities of *Musta*. The roadmap for a future release includes the integration of SnpEff, providing users with even more flexibility and options in their analyses. By incorporating multiple annotation tools, *Musta* aims to offer a comprehensive, adaptable, and user-friendly solution for somatic variant annotation, capable of accommodating diverse datasets and evolving research requirements.

By default, the classification module in *Musta* is configured to perform variant annotation using VEP, but users have the flexibility to opt for Funcotator alone or in combination with VEP. The fundamental structure of the *Musta* command for invoking the Classification module is:

```
musta classify --workdir WORKING-DIR --samples-file SAMPLES-FILE [options]
```

After annotating somatic mutations using both VEP and Funcotator, the resulting annotated Variant Call Format (VCF) files are further transformed into the Mutation Annotation Format (MAF). In fact, the MAF format is accepted as the standard one for storing both somatic and germline variants stemming from extensive cancer sequencing studies [70]. Once the variants are in the MAF format, the pipeline moves to the final stage of interpreting these variants through downstream analyses. These analyses are vital for unraveling the functional significance and potential clinical implications of the annotated somatic mutations.

4.2.3. Interpretation

The downstream analysis of somatic mutation data commonly involves the utilization of multiple independent software tools, employing various computational and statistical approaches. In the *Musta* pipeline, this analysis is significantly facilitated by incorporating the Maftools R package (v2.10) [71]. Maftools is a specialized bioinformatics tool designed for the comprehensive analysis and interpretation of somatic variants stored in the MAF format. It offers a wide range of downstream analysis and visualization modules that are extensively used in cancer genomic studies, enabling researchers to perform driver gene identification, pathway analysis, mutational signature analysis, enrichment analysis, and association analysis, among others [5,23]. Moreover, Maftools addresses the challenge of visualizing complex and heterogeneous data, providing researchers with an array of visualization functions to generate publication-quality images such as oncoplots, lollipop plots, and oncoprints. By incorporating Maftools into the *Musta* pipeline, we offer to

researchers the possibility to effectively interpret somatic variants, derive meaningful insights, conduct in-depth analyses, and communicate their findings more efficiently, ultimately enhancing our understanding of the genomic landscape in cancer research.

1. Variant visualization. Within the *Musta* pipeline framework, Maftools presents a range of graphical representations that assist in detecting mutation patterns and recurring characteristics within the dataset.
    - Summary plots provide an at-a-glance summary by showcasing variant counts per sample and their distribution based on classification. They offer a high-level overview of the mutation landscape within the dataset.
    - Onco plots illustrate mutations across samples, revealing distribution patterns. They are definitely valuable in cancer patient studies, providing a comprehensive mutational view.
    - Lollipop plots take the interpretation a step further by depicting mutations on protein structures. This visual representation helps researchers to gain a better understanding of the precise locations of mutations and their potential impact on protein structure and thus function.
    - Transitions and transversions plots categorize mutations into transitions and transversions, providing insights into the mutational spectrum within the dataset. Understanding these mutation types is crucial for unraveling mutational patterns.
    - Rainfall plots are a powerful tool for visualizing mutation-rich areas within cancer genomes. They are especially useful for identifying hypermutated regions, a phenomenon known as kataegis [23,72]. These plots help pinpoint areas of intense genomic alteration.
    - Oncostrip allows researchers to zoom in on specific genes, simplifying the exploration of features like mutual exclusivity. This focused view aids in uncovering relationships and interactions among genes of interest.
2. Somatic interactions. Recent advances in cancer genomics research have shown that many disease-causing genes in cancer are often mutated in a mutually exclusive manner [73,74]. Identification of such gene sets can reveal de novo pathways and underlying mechanisms of tumorigenesis. For this, *Musta* performs a Fisher's exact test on all combinations of genes, to detect such mutually exclusive or co-occurring sets of genes.
3. Detecting cancer driver genes. Cancer driver genes provide selective growth advantage to cancer cells when mutated [3]. Several mathematical approaches have been developed to identify such driver genes [5,75–77]. In the context of the *Musta* framework, detection of such associated genes is based on the oncodriveCLUST algorithm [78]: the concept is that a majority of the mutations in oncogenes are clustered around mutational hotspots, whereas mutations on passenger genes are randomly distributed.
4. Pfam domains. In each type of cancer, specific protein domains are notably enriched with mutations [79,80]. The process of identifying and categorizing protein domains based on their mutation frequency serves the dual purpose of discerning the predominant domain affected within a particular cancer cohort. This approach further aids in pinpointing highly disrupted pathways and protein families that share similar functions, offering insights into the intricacies of deregulated mechanisms.
5. Tumor heterogeneity. Tumors are generally heterogeneous, composed of multiple clones and undergoing continuous evolution [81]. Heterogeneity can be inferred by clustering and classifying variants into sub clones, according to their allele frequencies [82,83]. Although clustering of variant allele frequencies gives us a fair idea on heterogeneity, it is also possible to measure the extent of heterogeneity in terms of a numerical value. MATH score is a simple quantitative measure of intra-tumor heterogeneity, which expresses the width of the variant allele frequency (VAF) distribution [84]. High MATH scores are found to be associated with poor prognosis and survival [85].

6. Mutational signature analysis. As cancer progresses, it develops a characteristic mutational pattern that unveils the underlying mutagenic processes at play. This revealing pattern can be deciphered through dimensional reduction techniques like non-negative matrix factorization (NMF) [23]. By decomposing a matrix containing nucleotide substitutions categorized into 96 substitution classes, based on their surrounding bases, specific mutational signatures, unique to each cancer type, emerge. These mutational signatures offer a blurred picture of the intricate mutational landscape of cancers. However, by cross-referencing them with validated signatures, we can achieve a clearer and more focused picture. This further enriches our understanding of the distinct processes that drive cancer progression.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/ijms25158044/s1.

Author Contributions: R.A. and M.M. developed the software, R.A., M.M. and V.R. performed the data processing and somatic calling and evaluation tests. R.A., M.M., E.P., M.P. and A.A. conceived the study. R.A. and M.M. drafted the manuscript. All authors reviewed the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Fondo di Beneficenza of Intesa Sanpaolo S.p.A., B/2020/0094, Italy and PRIN 2022WBLA3Z Ministry of University and Research.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The HCC datasets analyzed for this study can be found at the genome sequence archive of Beijing Institute of Genomics under accession id PRJCA000091: https://bigd.big.ac.cn/bioproject/browse/PRJCA000091 (accessed on 21 July 2024). *Musta* is publicly released at the following GitHub repository: https://github.com/next-crs4/mustapy.

Acknowledgments: Rossano Atzeni performed his activity in the framework of the International PhD in Innovation Sciences and Technologies at the University of Cagliari, Italy. Rossano Atzeni, Enrico Pieroni, and Massimo Pisu were partially supported by the Sardinian Regional Authorities.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflicts of interest.

References

1. Mardis, E.R. A decade's perspective on DNA sequencing technology. *Nature* **2011**, *470*, 1483–1489. [CrossRef] [PubMed]
2. Martincorena, I.; Campbell, P.J. Somatic mutation in cancer and normal cells. *Science* **2015**, *349*, 198–203. [CrossRef] [PubMed]
3. Vogelstein, B.; Papadopoulos, N.; Velculescu, V.E.; Zhou, S.; Diaz, L.A., Jr.; Kinzler, K.W. Cancer genome landscapes. *Science* **2013**, *339*, 1546–1558. [CrossRef] [PubMed]
4. Garraway, L.A. Genomics-driven oncology: Framework for an emerging paradigm. *J. Clin. Oncol.* **2013**, *31*, 1806–1814. [CrossRef] [PubMed]
5. Lawrence, M.S.; Stojanov, P.; Polak, P.; Kryukov, G.V.; Cibulskis, K.; Sivachenko, A.; Getz, G. Mutational heterogeneity in cancer and the search for new cancer-associated genes. *Nature* **2013**, *499*, 214–218. [CrossRef] [PubMed]
6. Gerlinger, M.; Rowan, A.J.; Horswell, S.; Larkin, J.; Endesfelder, D.; Gronroos, E.; Swanton, C. Intratumor heterogeneity and branched evolution revealed by multiregion sequencing. *N. Engl. J. Med.* **2012**, *366*, 883–892. [CrossRef] [PubMed]
7. Shendure, J.; Ji, H. Next-generation DNA sequencing. *Nat. Biotechnol.* **2008**, *26*, 1135–1145. [CrossRef] [PubMed]
8. Jäger, N. Bioinformatics workflows for clinical applications in precision oncology. *Nat. Semin. Cancer Biol.* **2022**, *84*, 103–112. [CrossRef] [PubMed]
9. Ling, S.; Hu, Z.; Yang, Z.; Yang, F.; Li, Y.; Lin, P.; Chen, K.; Dong, L.; Cao, L.; Tao, Y.; et al. Extremely high genetic diversity in a single tumor points to prevalence of non-darwinian cell evolution. *Proc. Natl Acad. Sci. USA* **2015**, *112*, E6496–E6505. [CrossRef]
10. Li, Z.; Gao, H.; Zhang, X.; Liu, Q.; Chen, G. Mutational and transcriptional alterations and clinicopathological factors predict the prognosis of stage I hepatocellular carcinoma. *BMC Gastroenterol.* **2022**, *22*, 427. [CrossRef]
11. Shen, J.; Qi, L.; Zou, Z.; Du, J.; Kong, W.; Zhao, L.; Wei, J.; Lin, L.; Ren, M.; Liu, B. Identification of a novel gene signature for the prediction of recurrence in HCC patients by machine learning of genome-wide databases. *Sci. Rep.* **2020**, *10*, 4435. [CrossRef] [PubMed]
12. Wang, D.; Zhang, L.; Sun, Z.; Jiang, H.; Zhang, J. A radiomics signature associated with underlying gene expression pattern for the prediction of prognosis and treatment response in hepatocellular carcinoma. *Eur. J. Radiol.* **2023**, *167*, 111086. [CrossRef]

13. Wang, Q.; Zhai, Y.Y.; Dai, J.H.; Li, K.Y.; Deng, Q.; Han, Z.G. SAMD9L inactivation promotes cell proliferation via facilitating G1-S transition in hepatitis B virus-associated hepatocellular carcinoma. *Int. J. Biol. Sci.* **2014**, *10*, 807–816. [CrossRef] [PubMed]
14. Bian, X.; Shi, D.; Xing, K.; Zhou, H.; Lu, L.; Yu, D.; Wu, W. AMD1 upregulates hepatocellular carcinoma cells stemness by FTO mediated mRNA demethylation. *Clin. Transl. Med.* **2021**, *11*, e352. [CrossRef] [PubMed]
15. Zhou, X.; Huang, J.M.; Li, T.M.; Liu, J.Q.; Wei, Z.L.; Lan, C.L.; Zhu, G.Z.; Liao, X.W.; Ye, X.P.; Peng, T. Clinical Significance and Potential Mechanisms of ATP Binding Cassette Subfamily C Genes in Hepatocellular Carcinoma. *Front. Genet.* **2022**, *13*, 805961. [CrossRef] [PubMed]
16. Zhang, Y.; Qiu, Z.; Wei, L.; Tang, R.; Lian, B.; Zhao, Y.; He, X.; Xie, L. Integrated analysis of mutation data from various sources identifies key genes and signaling pathways in hepatocellular carcinoma. *PLoS ONE* **2014**, *9*, e100854. [CrossRef] [PubMed]
17. Zheng, P.; Xiao, W.; Zhang, J.; Zheng, X.; Jiang, J. The role of AIM2 in human hepatocellular carcinoma and its clinical significance. *Pathol. Res. Pract.* **2023**, *245*, 154454. [CrossRef] [PubMed]
18. Mroweh, M.; Roth, G.; Decaens, T.; Marche, P.N.; Lerat, H.; Macek Jílková, Z. Targeting Akt in Hepatocellular Carcinoma and Its Tumor Microenvironment. *Int. J. Mol. Sci.* **2021**, *22*, 1794. [CrossRef] [PubMed]
19. Ellrott, K.; Bailey, M.H.; Saksena, G.; Covington, K.R.; Kandoth, C.; Stewart, C.; Hess, J.; Ma, S.; Chiotti, K.E.; McLellan, M.; et al. MC3 Working Group; Cancer Genome Atlas Research Network. Scalable Open Science Approach for Mutation Calling of Tumor Exomes Using Multiple Genomic Pipelines. *Cell Syst.* **2018**, *28*, 271–281. [CrossRef]
20. Mölder, F.; Jablonski, K.P.; Letcher, B.; Hall, M.B.; Tomkins-Tinch, C.H.; Sochat, V.; Forster, J.; Lee, S.; Twardziok, S.O.; Kanitz, A.; et al. Holtgrewe M, Rahmann S, Nahnsen S, Köster J. Sustainable data analysis with Snakemake. *F1000Research* **2021**, *10*, 33. [CrossRef]
21. Docker: Lightweight Linux Containers for Consistent Development and Deployment. Available online: https://www.docker.com (accessed on 21 July 2024).
22. The Snakemake API Reference. Available online: https://snakemake.readthedocs.io/en/v7.0.0/api_reference/snakemake.html (accessed on 22 July 2024).
23. Alexandrov, L.B.; Nik-Zainal, S.; Wedge, D.C.; Aparicio, S.A.; Behjati, S.; Biankin, A.V.; Bignell, G.R.; Bolli, N.; Borg, A.; Borresen-Dale, A.L.; et al. Signatures of mutational processes in human cancer. *Nature* **2013**, *500*, 415–421. [CrossRef] [PubMed]
24. Andrews, S. FastQC: A Quality Control Tool for High Throughput Sequence Data. Available online: http://www.bioinformatics.babraham.ac.uk/projects/fastqc (accessed on 22 July 2024).
25. Li, H.; Durbin, R. Fast and accurate short read alignment with Burrows—Wheeler transform. *Bioinformatics* **2009**, *25*, 1754–1760. [CrossRef] [PubMed]
26. Krueger, F. Trim Galore. A Wrapper Tool around Cutadapt and FastQC to Consistently Apply Quality and Adapter Trimming to FastQ Files. Available online: https://www.bioinformatics.babraham.ac.uk/projects/trim_galore/ (accessed on 22 July 2024).
27. Van der Auwera, G.A.; Carneiro, M.O.; Hartl, C.; Poplin, R.; Del Angel, G.; Levy-Moonshine, A.; DePristo, M.A. From FastQ data to high-confidence variant calls: The Genome Analysis Toolkit best practices pipeline. *Curr. Protoc. Bioinform.* **2013**, *43*, 11.10.1–11.10.33. [CrossRef] [PubMed]
28. Hwang, S.; Kim, E.; Lee, I.; Marcotte, E.M. Systematic comparison of variant calling pipelines using gold standard personal exome variants. *Sci. Rep.* **2015**, *5*, 17875. [CrossRef] [PubMed]
29. Xu, C. A review of somatic single nucleotide variant calling algorithms for next-generation sequencing data. *Comput. Struct. Biotechnol.* **2018**, *16*, 15–24. [CrossRef] [PubMed]
30. Kroigard, A.B.; Thomassen, M.; Laenkholm, A.V.; Kruse, T.A.; Larsen, M.J. Evaluation of nine somatic variant callers for detection of somatic mutations in exome and targeted deep sequencing data. *PLoS ONE* **2016**, *11*, e0151664. [CrossRef] [PubMed]
31. Roberts, N.D.; Kortschak, R.D.; Parker, W.T.; Schreiber, A.W.; Branford, S.; Scott, H.S.; Glonek, G.; Adelson, D.L. A comparative analysis of algorithms for somatic SNV detection in cancer. *Bioinformatics* **2013**, *29*, 2223–2230. [CrossRef] [PubMed]
32. Wang, Q.; Jia, P.; Li, F.; Chen, H.; Ji, H.; Hucks, D.; Dahlman K.b.; Pao, W.; Zhao, Z. Detecting somatic point mutations in cancer genome sequencing data: A comparison of mutation callers. *Genome Med.* **2013**, *5*, 91. [CrossRef]
33. Kim, S.Y.; Speed, T.P. Comparing somatic mutation-callers: Beyond Venn diagrams. *BMC Bioinform.* **2013**, *14*, 189. [CrossRef]
34. O'Rawe, J.; Jiang, T.; Sun, G.; Wu, Y.; Wang, W.; Hu, J.; Bodily, P.; Tian, L.; Hakonarson, H.; Johnson, W.E.; et al. Low concordance of multiple variant-calling pipelines: Practical implications for exome and genome sequencing. *Genome Med.* **2013**, *5*, 28. [CrossRef]
35. Goode, D.L.; Hunter, S.M.; Doyle, M.A.; Ma, T.; Rowley, S.M.; Choong, D.; Ryland, G.L.; Campbell, I.G. A simple consensus approach improves somatic mutation prediction accuracy. *Genome Med.* **2013**, *5*, 90. [CrossRef] [PubMed]
36. Chiara, M. Gioiosa, S.; Chillemi, G.; D'Antonio, M.; Flati, T.; Picardi, E.; Zambelli, F.; Horner, D.S.; Pesole, G.; Castrignanó, T. CoVaCS: A consensus variant calling system. *BMC Genom.* **2018**, *19*, 120. [CrossRef] [PubMed]
37. Liu, Z.K.; Shang, Y.K.; Chen, Z.N.; Bian, H. A three-caller pipeline for variant analysis of cancer whole-exome sequencing data. *Mol. Med. Rep.* **2017**, *15*, 2489–2494. [CrossRef] [PubMed]
38. Rashid, M.; Robles-Espinoza, C.D., Rust, A.G.; Adams, D.J. Cake: A bioinformatics pipeline for the integrated analysis of somatic variants in cancer genomes. *Bioinformatics* **2013**, *29*, 2208–2210 [CrossRef] [PubMed]
39. Zhou, Z.-H. *Ensemble Methods: Foundations and Algorithms*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2012.
40. Hansen, L.K.; Salamon, P. Neural network ensembles. *IEEE Trans. Pattern Anal.* **1990**, *12*, 993–1001. [CrossRef]
41. Brown, G.; Wyatt, J.; Harris, R.; Yao, X. Diversity creation methods: A survey and categorisation. *Inf. Fusion* **2005**, *6*, 5–20. [CrossRef]

42. Ewing, A.D.; Houlahan, K.E.; Hu, Y.; Ellrott, K.; Caloian, C.; Yamaguchi, T.N.; Bare, J.C.; P'ng, C.; Waggott, D.; Sabelnykova, V.Y.; et al. Combining tumor genome simulation with crowdsourcing to benchmark somatic single-nucleotide-variant detection. *Nat. Methods* **2015**, *12*, 623–630. [CrossRef] [PubMed]
43. Callari, M.; Sammut, S.J.; De Mattos-Arruda, L.; Bruna, A.; Rueda, O.M.; Chin, S.F.; Caldas, C. Intersect-then-combine approach: Improving the performance of somatic variant calling in whole exome sequencing data using multiple aligners and callers. *Genome Med.* **2017**, *9*, 35. [CrossRef] [PubMed]
44. Kim, S.Y.; Jacob, L.; Speed, T.P. Combining calls from multiple somatic mutation-callers. *BMC Bioinf.* **2014**, *15*, 154. [CrossRef]
45. Cantarel, B.L.; Weaver, D.; McNeill, N.; Zhang, J.; Mackey, A.J.; Reese, J. BAYSIC: A Bayesian method for combining sets of genome variants with improved specificity and sensitivity. *BMC Bioinform.* **2014**, *15*, 104. [CrossRef]
46. Anzar, I.; Sverchkova, A.; Stratford, R.; Clancy, T. NeoMutate: An ensemble machine learning framework for the prediction of somatic mutations in cancer. *BMC Med. Genom.* **2019**, *12*, 63. [CrossRef] [PubMed]
47. Fang, L.T.; Afshar, P.T.; Chhibber, A.; Mohiyuddin, M.; Fan, Y.; Mu, J.C.; Gibeling, G.; Barr, S.; Asadi, N.B.; Gerstein, M.B.; et al. An ensemble approach to accurately detect somatic mutations using SomaticSeq. *Genome Biol.* **2015**, *16*, 197. [CrossRef]
48. Ainscough, B.J.; Barnell, E.K.; Ronning, P.; Campbell, K.M.; Wagner, A.H.; Fehniger, T.A.; Dunn, G.P.; Uppaluri, R.; Govindan, R.; Rohan, T.E.; et al. A deep learning approach to automate refinement of somatic variant calling from cancer sequencing data. *Nat. Genet.* **2018**, *50*, 1735–1743. [CrossRef] [PubMed]
49. Sahraeian, S.M.E.; Liu, R.; Lau, B.; Podesta, K.; Mohiyuddin, M.; Lam, H.Y.K. Deep convolutional neural networks for accurate somatic mutation detection. *Nat. Commun.* **2019**, *10*, 1041. [CrossRef] [PubMed]
50. Cibulskis, K.; Lawrence, M.S.; Carter, S.L.; Sivachenko, A.; Jaffe, D.; Sougnez, C.; Gabriel, S. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. *Nat. Biotechnol.* **2013**, *31*, 213–219. [CrossRef] [PubMed]
51. Koboldt, D.C.; Zhang, Q.; Larson, D.E.; Shen, D.; McLellan, M.D.; Lin, L.; Miller, C.A.; Mardis, E.R.; Ding, L.; Wilson, R.K. VarScan 2: Somatic mutation and copy number alteration discovery in cancer by exome sequencing. *Genome Res.* **2012**, *22*, 568–576. [CrossRef] [PubMed]
52. Lai, Z.; Markovets, A.; Ahdesmaki, M.; Johnson, J. VarDict: A novel and versatile variant caller for next-generation sequencing in cancer research. *AACR Annu. Meeting* **2015**, *44*, e108. [CrossRef] [PubMed]
53. Kim, S.; Scheffler, K.; Halpern, A.L.; Bekritsky, M.A.; Noh, E.; Källberg, M.; Chen, X.; Kim, Y.; Beyter, D.; Krusche, P.; et al. Strelka2: Fast and accurate calling of germline and somatic variants. *Nat. Methods* **2018**, *15*, 591–594. [CrossRef]
54. Wilm, A.; Aw, P.P.K.; Bertrand, D.; Yeo, G.H.T.; Ong, S.H.; Wong, C.H.; Khor, C.C.; Petric, R.; Hibberd, M.L.; Nagarajan, N. LoFreq: A sequence-quality aware, ultra-sensitive variant caller for uncovering cell-population heterogeneity from high-throughput sequencing datasets. *Nucleic Acids* **2012**, *40*, 11189–11201. [CrossRef]
55. Fan, Y.; Xi, L.; Hughes, D.S.T.; Zhang, J.; Zhang, J.; Futreal, P.A.; Wheeler, D.A.; Wang, W. MuSE: Accounting for tumor heterogeneity using a sample-specific error model improves sensitivity and specificity in mutation calling from sequencing data. *Genome Biol.* **2016**, *17*, 178. [CrossRef]
56. Wang, M.; Luo, W.; Jones, K.; Bian, X.; Williams, R.; Higson, H.; Wu, D.; Hicks, B.; Yeager, M.; Zhu, B. SomaticCombiner: Improving the performance of somatic variant calling based on evaluation tests and a consensus approach. *Sci. Rep.* **2020**, *10*, 12898. [CrossRef]
57. Zhang, C.; El-Kebir, M.; Ochoa, I. Moss enables high sensitivity single-nucleotide variant calling from multiple bulk DNA tumor samples. *Nat. Commun.* **2021**, *12*, 2204. [CrossRef] [PubMed]
58. Roth, A.; Ding, J.; Morin, R.; Crisan, A.; Ha, G.; Giuliany, R.; Bashashati, A.; Hirst, M.; Turashvili, G.; Oloumi, A.; et al. JointSNVMix: A probabilistic model for accurate detection of somatic mutations in normal/tumour paired next-generation sequencing data. *Bioinformatics* **2012**, *28*, 907–913. [CrossRef] [PubMed]
59. Larson, D.E.; Harris, C.C.; Chen, K.; Koboldt, D.C.; Abbott, T.E.; Dooling, D.J.; Ley, T.J.; Mardis, E.R.; Wilson, R.K.; Ding, L. SomaticSniper: Identification of somatic point mutations in whole genome sequencing data. *Bioinformatics* **2012**, *28*, 311–317 [CrossRef] [PubMed]
60. Fang, H.; Bergmann, E.A.;Arora, K.; Vacic, V.; Zody, M.C.; Iossifov, I.; O'Rawe, J.A.; Wu, Y.; Jimenez Barron, L.T.; Rosenbaum, J.; et al. Indel variant analysis of short-read sequencing data with Scalpel. *Nat. Protoc.* **2016**, *11*, 2529–2548. [CrossRef] [PubMed]
61. Sherry, S.T.; Ward, M.H.; Kholodov, M.; Baker, J.; Phan, L.; Smigielski, E.M.; Sirotkin, K. DbSNP: The NCBI database of genetic variation. *Nucleic Acids Res.* **2001**, *29*, 308–311. [CrossRef] [PubMed]
62. Karczewski, K.J.; Francioli, L.C.; Tiao, G.; Cummings, B.B.; Alföldi, J.; Wang, Q.; Collins, R.L.; Laricchia, K.M.; Ganna, A.; Birnbaum, D.P.; et al. Variation across 14,1456 human exomes and genomes reveals the spectrum of loss-of-function intolerance across human protein-coding genes. *bioRxiv* **2019**, *531210*. [CrossRef]
63. Landrum, M.J.; Lee, J.M.; Riley, G.R.; Jang, W.; Rubinstein, W.S.; Church, D.M.; Maglott, D.R. ClinVar: Public archive of relationships among sequence variation and human phenotype. *Nucleic Acids Res.* **2014**, *42*, D980–D985. [CrossRef] [PubMed]
64. Forbes, S.A.; Beare, D.; Gunasekaran, P.; Leung, K.; Bindal, N.; Boutselakis, H.; Ding, M.; Bamford, S.; Cole, C.; Ward, S.; et al. COSMIC: Exploring the world's knowledge of somatic mutations in human cancer. *Nucleic Acids Res.* **2015**, *43*, D805–D811. [CrossRef]
65. McLaren, W.; Gil, L.; Hunt, S.E.; Riat, H.S.; Ritchie, G.R.; Thormann, A.; Flicek, P.; Cunningham, F. The ensembl variant effect predictor. *Genome Biol.* **2016**, *17*, 122. [CrossRef]

66. Wang, K.; Li, M.; Hakonarson, H. ANNOVAR: Functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Res.* **2010**, *38*, e164. [CrossRef] [PubMed]
67. Cingolani, P.; Platts, A.; Wang, L.L.; Coon, M.; Nguyen, T.; Wang, L.; Land, S.J.; Lu, X.; Ruden, D.M. A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. *Fly* **2012**, *6*, 80–92. [CrossRef]
68. Tuteja, S.; Kadri, S.; Yap, K.L. A performance evaluation study: Variant annotation tools - the enigma of clinical next generation sequencing (NGS) based genetic testing. *J. Pathol. Inf.* **2022**, *13*, 2153–3539. [CrossRef]
69. Yen, J.L.; Garcia, S.; Montana, A.; Harris, J.; Chervitz, S.; Morra, M.; West, J.; Chen, R.; Church, D.M. A variant by any name: Quantifying annotation discordance across tools and clinical databases. *Genome Med.* **2017**, *9*, 7. [CrossRef] [PubMed]
70. Mutation Annotation Format. Available online: https://wiki.nci.nih.gov/display/TCGA/Mutation+Annotation+Format (accessed on 21 July 2024).
71. Mayakonda, A.; Lin, D.C.; Assenov, Y.; Plass, C.; Koeffler, H.P. Maftools: Efficient and comprehensive analysis of somatic variants in cancer. *Genome Res.* **2018**, *28*, 1747–1756. [CrossRef] [PubMed]
72. Nik-Zainal, S.; Van Loo, P.; Wedge, D.C.; Alexandrov, L.B.; Greenman, C.D.; Lau, K.W.; Raine, K.; Jones, D.; Marshall, J.; Ramakrishna, M.; et al. The life history of 21 breast cancers. *Cell* **2012**, *149*, 994–1007. [CrossRef] [PubMed]
73. Leiserson, M.D.; Wu, H.; Vandin, F.; Raphael, B.J. CoMEt: A statistical approach to identify combinations of mutually exclusive alterations in cancer. *Genome Biol.* **2015**, *16*, 160. [CrossRef] [PubMed]
74. Yeang, C.H.; McCormick, F.; Levine, A. Combinatorial patterns of somatic gene mutations in cancer. *FASEB J.* **2008**, *22*, 2605–2622. [CrossRef] [PubMed]
75. Dees, N.D.; Zhang, Q.; Kandoth, C.; Wendl, M.C.; Schierding, W.; Koboldt, D.C.; Mooney, T.B.; Callaway, M.B.; Dooling, D.; Mardis, E.R.; et al. MuSiC: Identifying mutational significance in cancer genomes. *Genome Res.* **2012**, *22*, 1589–1598. [CrossRef] [PubMed]
76. Gonzalez-Perez, A.; Lopez-Bigas, N. Functional impact bias reveals cancer drivers. *Nucleic Acids Res.* **2012**, *40*, e169. [CrossRef]
77. Zapata, L.; Susak, H.; Drechsel, O.; Friedlander, M.R.; Estivill, X.; Ossowski, S. Signatures of positive selection reveal a universal role of chromatin modifiers as cancer driver genes. *Sci. Rep.* **2017**, *7*, 13124. [CrossRef] [PubMed]
78. Tamborero, D.; Gonzalez-Perez, A.; Lopez-Bigas, N. OncodriveCLUST: Exploiting the positional clustering of somatic mutations to identify cancer genes. *Bioinformatics* **2013**, *29*, 2238–2244. [CrossRef] [PubMed]
79. Yang, F.; Petsalaki, E.; Rolland, T.; Hill, D.E.; Vidal, M.; Roth F.P. Protein domain-level landscape of cancer-type-specific somatic mutations. *PLoS Comput. Biol.* **2015**, *11*, e1004147. [CrossRef]
80. Nehrt, N.L.; Peterson, T.H.; Park, D.; Kann, M.G. Domain landscapes of somatic mutations in cancer. *BMC Genom.* **2012**, *13* (Suppl. 4), S9. [CrossRef] [PubMed]
81. Nowell, P.C. The clonal evolution of tumor cell populations. *Science* **1976**, *194*, 23–28. [CrossRef] [PubMed]
82. Ding, L.; Ley, T.J.; Larson, D.E.; Miller, C.A.; Koboldt, D.C.; Welch, J.S.; Ritchey, J.K.; Young, M.A.; Lamprecht, T.; McLellan, M.D.; et al. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. *Nature*, **2012**, *481*, 506–510. [CrossRef]
83. Miller, C.A.; White, B.S.; Dees, N.D.; Griffith, M.; Welch, J.S.; Griffith, O.L.; Vij, R.; Tomasson, M.H.; Graubert, T.A.; Walter, M.J.; et al. SciClone: Inferring clonal architecture and tracking the spatial and temporal patterns of tumor evolution. *PLoS Comput. Biol.* **2014**, *10*, e1003665. [CrossRef] [PubMed]
84. Mroz, E.A.; Rocco, J.W. MATH, a novel measure of intratumor genetic heterogeneity, is high in poor-outcome classes of head and neck squamous cell carcinoma. *Oral Oncol.* **2013**, *49*, 211–215. [CrossRef]
85. Mroz, E.A.; Tward, A.D.; Hammon, R.J.; Ren, Y.; Rocco, J.W. Intra-tumor genetic heterogeneity and mortality in head and neck cancer: Analysis of data from the Cancer Genome Atlas. *PLoS Med.* **2015**, *12*, e1001786. [CrossRef]