Identifying Novel Sources of Non-canonical Tumor Antigens via the Hybrid de novo Transcriptome Assembly Pipeline

Identifying Novel Sources of Non-canonical Tumor Antigens via the Hybrid de novo Transcriptome Assembly Pipeline

Jirapat Techachakrit

A Thesis in the Field of Bioengineering and Nanotechnology

for the Degree of Master of Liberal Arts in Extension Studies

Harvard University

May 2022

Abstract

Traditionally, tumor-specific antigens (TSAs) are believed to result solely from mutations occurring within specific cancer types. Peptides resulting from these mutations can be applied as therapeutic agents which engineer the body's immune system so that it can identify and remove tumorigenic cells with greater efficiency. These TSA-based therapeutics are known as neoantigen vaccines. Neoantigen vaccines have been developed for personalized treatment of cancer patients, resulting in the overall increased patient survival rates. However, the discovery process of neoantigen vaccines is considered inefficient, often resulting in delays in production, and subsequently, in the delivery to critical cancer patients. Despite developments of sophisticated algorithms, the discovery process of mutation-based neoantigenic TSAs continues to elude researchers with its rarity. The difficulties in the identification process may be attributed to the dynamic nature of the mutational landscape in each tumor. This dynamic nature makes identifying a viable neoantigenic mutation a possibility only to some cancer patients. Furthermore, the rate at which a tumor mutates also varies tumor-to-tumor and patient-to-patient, making a streamline processing of neoantigen vaccine largely impractical. This Thesis serves as a new frontier towards improving TSAs identification and widening the discoverable landscape by identifying novel classes of tumor-specific antigens arising from alternative, non-mutational sources. The work done in this Thesis utilized open-source bioinformatic tools in conjunct with in-house python scripts. The resulting tool is called the novel Hybrid *de novo* Transcriptome Assembly Pipeline ("hybrid *de novo*").

The novel hybrid *de novo* pipeline investigates non-mutational tumorigenic landscapes in carcinomas, sarcomas, and neoplasms in the lung. These landscapes are compared against the landscapes of a healthy human tissue panel (comprising of various tissues of the body) in order to identify the lung TSAs. The procedure largely analyzes total RNA-seqs of the cancer and healthy tissues to identify the presence of any non-canonical transcripts in the non-canonical transcriptional frames. These non-canonical transcriptional frames, which have previously been of little interest in cancer treatments field, have resurfaced as the prime suspect of potential sources for novel TSA isoforms. Utilizing the novel hybrid *de novo* pipeline, we were able to identify a total of 20 novel, non-canonical TSAs existing only within the lung cancer patient samples (N = 100), all of which were shared amongst the patients with varying degrees of frequency. We also identified an additional 11 novel isoforms that have high expressions in cancer samples, and low expressions in healthy tissues (tumor-associated antigens – TAAs). Summarily, the application of the novel hybrid *de novo* pipeline in TSAs and TAAs discoveries, in conjunct to traditional pipeline (mutational-based) will improve the overall chance for neoantigen vaccine discovery and clinical translation. Furthermore, the presence of shared TSAs and TAAs show a significant potential for stratifying neoantigen vaccines in patients with similar tumor-genomic make up. If the process for patient genome profiling can be streamlined, groups of cancer patients whose tumor expresses shared TSAs/TAAs will benefit from ready-to-use 'generic' neoantigen vaccines. The concept of a 'generic' drug for use in personalized medicine will largely reduce the overall time required from lab-to-patient while maintaining the precision associated with traditional personalized medicine.

Author's Biographical Sketch

Jirapat "JT" Techachakrit is an international member of the diverse community of students at Harvard Extension School. Her field of study at Harvard is Bioengineering and Nanotechnology. Her special interests are in immunology, oncology, immunotherapy, systems biology, genetic engineering, nano-drug delivery, translational medicine, and bioinformatics.

Even though JT believes that being educated is a fundamental human right, there was no equality in Thailand. In 2012, with more questions than answers provided at home, JT found herself in Washington state in the USA, funded by the international exchange student scholarship. Her first trip to the United States yielded a near-decade of being on the frontier of research and higher education. Even now, JT seeks to apply her experiences (both as a person growing up with two drastically different cultures and as a survivor of many waves of abuse of socioeconomic segregation, racism, sexism, ageism, and disability) to shrink the gaps in educative inequality. She hopes that with better education and better access to education, more people could help represent and alleviate the pressure of other socioeconomic inequalities that prevent millions from having access to their basic needs and rights.

Dedication

I dedicate this Thesis to my two co-mentors, Dr. Sira Sriswasdi, (Ph.D., University of Pennsylvania), and Dr. Sujata Bhatia, (M.D., Ph.D., University of Pennsylvania), and their respective committees from both Harvard University and the Thesis' surrogate institution, Chulalongkorn University during the coronavirus pandemics.

Through the continued guidance from Dr. Sriswasdi and Dr. Bhatia, I have composed and refined this Thesis. They had offered their expertise, provided learning opportunities, skills, and valuable advice when I most needed them.

Acknowledgments

This project could not have been completed without the support of my family, especially those that have already passed on. Their strength serves as my own in times of difficulties, and their struggles have built foundations that allowed me the privilege to pursue science at its forefront. Thank you to my other family, Dr. David Michael "Mike" Payne and Judge Sam, his husband, for your continued encouragement and support.

I want to thank my friends: Boat, Zane, Gunt, Nutt, Kite, Win, BB, Jesse, Reza, Baiploo, Rehab, Cecilie, Eller, Amaria, Andrew, and many others for their academic advice and moral support. Special thanks to Rita and Bryan Jameson, whose perseverance and stories in fighting cancer continue to inspire my work; may she rest in peace. I am privileged to have met and befriended such diverse and passionate people worldwide. I could not hope to dream such big dreams without the support I have received. Thank you for being a part of my story.

Table of Contents

List of Figures

List of Equations

Chapter I.

Introduction

Background

Over the past two decades, the emergence of cancer immunotherapy has rapidly advanced in the field of oncology worldwide. Cancer immunotherapy is a class of treatment that harnesses the power of the immune system in order to prevent, control, and eliminate cancer cells. Cancer immunotherapy has been considered another effective option for cancer treatments, joining surgeries, cytotoxic chemotherapy, and cellular irradiation. Cancer immunotherapy comes in many forms, including immune checkpoint inhibitors (ICIs), cancer vaccines, adoptive cell transfer, and tumor-infecting viruses (Rizvi, 2017; Voelker, 2020). Amongst these, the ICIs have been most widely used for commercial cancer treatments, resulting in dramatic decreases in subsets of malignancies.

However, the anti-tumor effects of ICIs are generally dependent on the presence of the immune checkpoint receptors, which can be under-expressed in some cancer subtypes (Oiseth and Aziz, 2017) under immune-suppressive tumor microenvironment (TME). A TME is a major obstacle to the success of ICIs therapy since the tightly-packed compound of cancer cells does not allow for tumor-infiltrating lymphocytes (TILs) to reach immune checkpoint receptors. The lack of these receptors renders ICIs unusable, allowing cancer cells to evade the immunological recognition and escape (Bodey et al., 2000). As ICIs are only effective against specific cancer profiles, researchers seek to improve cancer response via multimodal therapy with ICIs and cancer vaccines. Clinical trials of these multimodal therapies provided evidence of positive priming of cancer cells via cancer vaccines. The combination immunotherapy approach can turn a ‘cold’ (immunologically low-response tumor) tumor ‘hot’ (immunologically high-response tumor) by targeting antigens expressed exclusively on cancer cells, allowing for TILs to enter the TME while also priming the T-cell immune responses (Collins et al., 2018).

Evolution of Multimodal Immunotherapies

In 2010, the first autologous cell-based cancer vaccine, sipuleucel-T, was approved by the United States Food and Drug Administration for the treatment of metastatic prostate cancer. Sipuleucel-T, which contains antigen-presenting cells (APCs) primed with a recombinant fusion protein, PA2024, revealed the first instances in precision immune engineering. In order to stimulate 'memory,' PA2024 makes use of granulocyte-macrophage colony-stimulating factor (GM-CSF), inducing the activation of APCs for the downstream immuno-processing of its fused prostatic acid phosphatase (PAP) antigen (Kantoff et al., 2010).

Cancer vaccines such as sipuleucel-T engage the immune response by generating antigen-specific T-cells in combination therapy. The recognition process of cancer vaccine-induced T-cells is initiated by the binding of naïve T-cell receptors to foreign antigen from the vaccine contained in the major histocompatibility complex (MHC) class II of the human APCs. Upon recognizing a non-self antigen, primed T-cells undergo multiple proliferation phases to form a large pool of effector cells that recognize the same antigen presented on other cells of the body. After expansion, the newly formed pool of primed, effector T-cells facilitates the identification and removal of cancer cells with other lymphatic cells, such as the natural killer (NK) cells (Figure 1).

In 2017, a phase I study of sipuleucel-T and an ICI ipilimumab resulted in a statistical increase in serum antibodies specific to PAP and PA2024 above the level achieved with monotherapy of either medication (Scholz, et al. 2017). These results suggest the importance of antigen-specific T-cells undergoing the ‘prime-expand-facilitate’ induced by the cancer vaccine, which improved the median survival rates of patients enrolled.

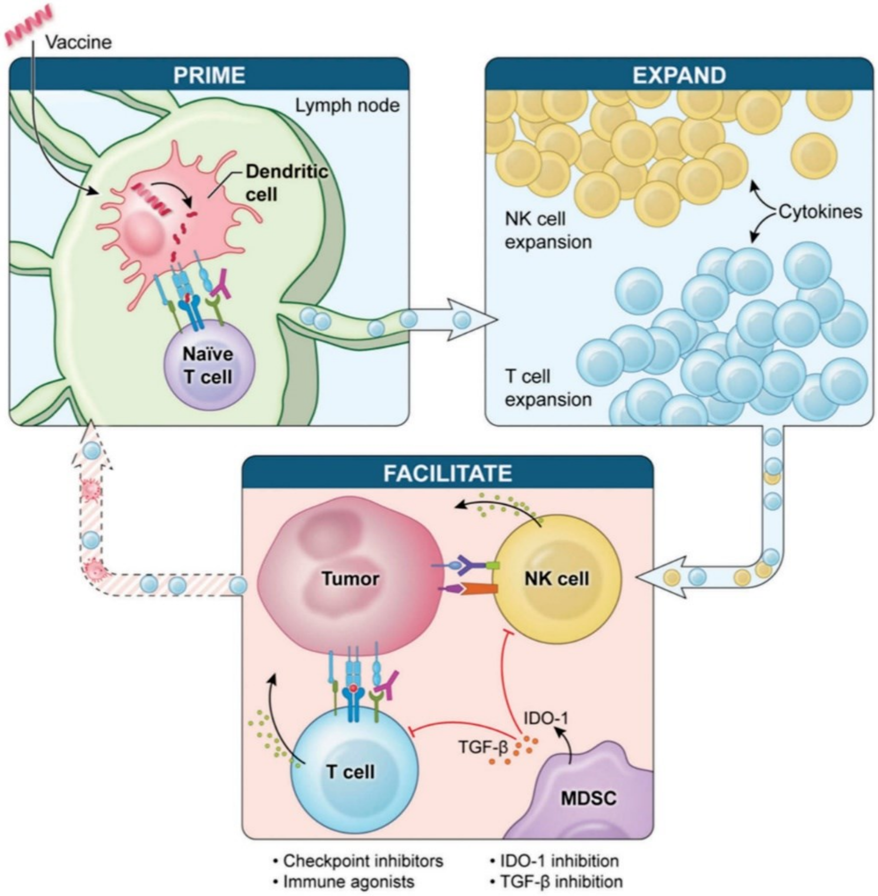


Figure 1: Multimodal immunotherapy can prime, expand, and facilitate the anti-tumor response

(Collins, et al. 2018)

Non-synonymous Mutated Neoantigen Vaccine

Unlike sipuleucel-T, which is a cell-based cancer vaccine that is cultured *in vitro* as pre-primed, engineered APCs; the modern cancer vaccines employed 'internal' immune-engineering by inoculating cancer peptides to the patients directly. This modern method induces immune responses *in vivo*, reducing the need for high-level inoculation of live cells, thus easing difficulties in production, safety controls, and costs (Bassani-Sternberg et al., 2016).

With the refinement in modern cancer vaccine production, researchers now have the freedom to explore multitudes of vaccine candidates at a more modest cost, resulting in the development of ‘neoantigen’ class of vaccine for the use of T-cell based immunogenic priming. Neoantigens are special, non-autologous peptides that are presented on the surface of MHC class II of the APCs. The use of these antigens provides great specificity and accuracy for targeting cancer cells (Day et al., 2009). However, not all peptides are considered neoantigens. The criteria for being neoantigenic is dependent on multiple factors (Guo, Lei, and Tang, 2018; Wei et al., 2019; Hodge et al., 2020). First, peptides must be able to undergo APC uptake upon inoculation, where they are enzymatically lysated into smaller fragments. If the peptide fragments maintain strong kinetic affinities with the MHC class II protein, they will be presented on the surface of the APCs. Only after being presented on the APC are the peptides considered neoantigenic. The MHC-peptide complex is then recognized by the T-cell receptor and goes on to activate naïve T-cells (Oiseth and Aziz, 2017; Ribatti, 2017).

Being based on loose peptides, neoantigens alone are poor inducers of the adaptive immune response necessary for T-cell activations. In order to induce effective

T-cell priming, adjuvants are needed to attract immune cells to the site of injection. Adjuvants also promote cell-mediated trafficking of antigens to draining lymph nodes, triggering APCs activation (Paston et al., 2021). Following priming, expansion, and facilitation of T-cells, the pool of effector cells will maintain a self-recognizing biochemical feedback loop (Ochsenbein, 2002) to avoid harming healthy cells, but will effectively induce cellular eradication upon the recognition of non-self antigens (Zugazagoitia et al., 2016; Ribatti, 2017).

The number of peptide-based cancer vaccines being explored has increased since the establishment of neoantigens' role in immune-priming. The search for potent cancer-identifying neoantigens revolves around finding mutated antigenic peptides that present only on cancer cells, also known as tumor-specific antigens (TSAs). Modern neoantigen vaccines are based entirely on TSAs, delivering high specificity against tumors presenting such peptides (Anagnostou et al., 2017).

On the other hand, TSAs are excessively rare. Since the mutations that generate TSAs are considered patient-specific, TSA identifications are performed patient-wise. As is with many personalized medicines, the process for neoantigen discovery and synthesis is slow (Schumacher and Schreiber, 2015). In practice, researchers use rapid-synthesis of multiple TSAs candidates with varying expression in cancer to provide for a broad-coverage effect against tumors to compensate for the torpid turnaround time. This understanding that TSAs are 'patient-specific' mutagenic antigens that vary from patient-to-patient dates back to 1957, when R.T. Prehn and J. M. Main published an article called 'Immunity to Methylcholanthrene-induced Sarcomas.' This article, stating "that immunity to syngeneic tumors was theoretically impossible; the tumor was a part of the self and therefore could not arouse an immune response" (Prehn and Main, 1957) provided ground concepts for many neoantigen vaccine discovery techniques and computational tools in the present day (Gubin et al., 2015).

Methods in TSAs discoveries

As patient-specific therapies go, the discovery of the patient's own non-synonymous somatic mutations is crucial in neoantigen vaccine development (Ott et al., 2017; Hilf et al., 2019; Sun et al., 2019; Fang et al., 2020). Researchers searched for these mutations within the patients' protein-coding regions in the form of RNA sequencing analyses. Most often, the mutations discovered for neoantigen vaccines are from single-nucleotide variations (SNVs) and insertion-deletion (INDELS) mutational events. To identify these mutations, bioinformatic algorithms are used in conjunction with next-generation sequencing (NGS), Deep sequencing, proteogenomic analyses, immunoprecipitation, and mass spectrometry (MS) to validate TSA candidates. RNA sequencing analyses and validation are often performed before downstream MS variant calling and MHC-binding assessment (Hodge et al., 2020).

The process of MS variant calling involves heavy-duty processing in identifying candidate peptides and matching these peptides to those in the cancer proteome. The use of MS in neoantigen discoveries is so prevalent due to the ability of MS to directly compare peptide spectra between cancerous tissues to those in healthy tissues. In practice, however, one cannot easily isolate peptides of interest from cancerous tissues due to the scarce availability of cancer tissues and the large volume of MHC-specific antibodies needed for immunoprecipitation.

Since MS sensitivity to identify TSAs is conditional on the abundance of peptides for the spectra, peptides with low abundances cannot be detected (Hodge et al., 2020). Since low abundance peptides may escape undetected, the use of MS alone in the discovery process for neoantigens is impractical. In this sense, utilizing computationally analyzed RNA sequencing data alongside MS variant calling and MHC-binding assessment is now the standard workflow for neoantigen vaccines (Figure 2).

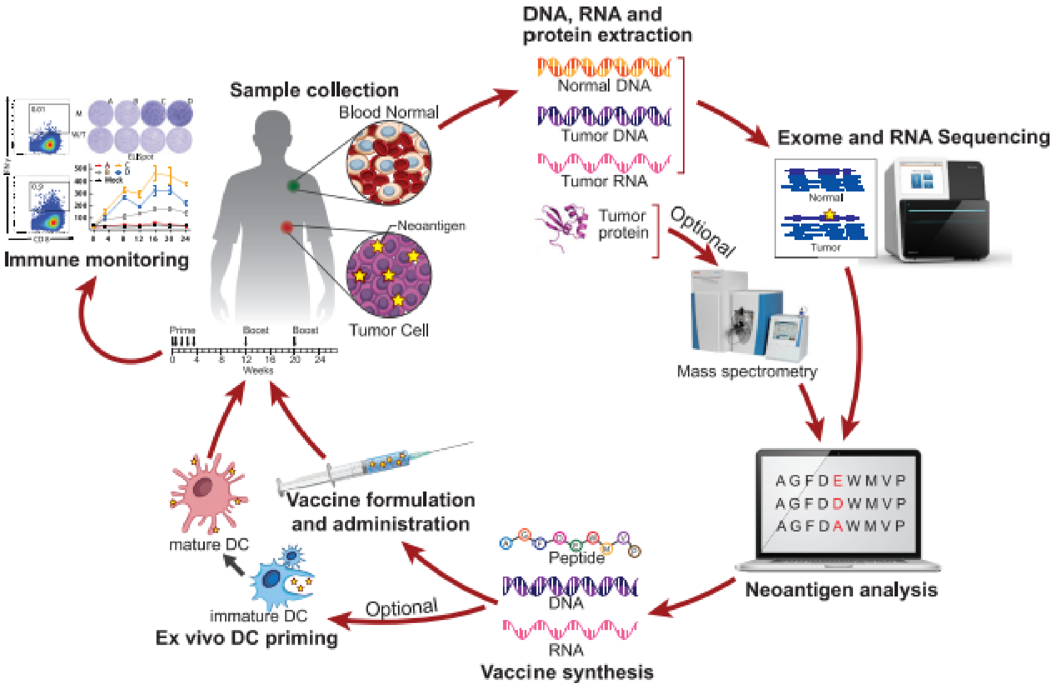


Figure 2: A standard workflow for neoantigen peptide discovery

(Hodge et al., 2020)

Still, the yield for immunogenic TSAs originating from SNVs and INDELs is low even amidst cancers with a high tumor mutational burden (TMB). TMB refers to the number of somatic gene mutations present in the tumor, varying across different cancer types. Tumors such as the non-small-cell lung cancer (NSCLC) and melanoma are considered to have high TMB (Alborelli et al., 2020; Stein et al., 2019). However, meta-analysis from Bassani-Sternberg and their team showed infrequent presence of immunogenic TSAs in melanoma despite its high mutational burden compared to other cancer types (Bassani-Sternberg et al., 2016). In a cross-analysis of 13 different SNVs neopeptide studies, only 53 of the 1,948 SNVs tested were shown to elicit T-cell responses (Bjerregaard et al., 2017), averaging to merely two immunogenic TSAs per tumor. These statistics expectedly declined in classes of cancer with lower TMB, such as those in acute myeloid leukemia (AML). The lack of usable immunogenic TSAs in most classes of cancer prompted for additional sources of TSAs (Sahin et al., 2017).

Origins of MHC-bound ‘non-canonical’ Antigens in Normal and Cancer Cells

A recent study has shown MHC-presentation of targetable neoantigens originating from non-canonical sources within the cancerous mouse and human genomes (Laumont et al., 2018). These neoantigens do not contain mutated residue but are expressionally deficient in normal tissues, presenting only in the cancer genome. Laumont and his team also defined another class of peptides, which had been previously focused on by targeted therapy, called the tumor-associated antigens (TAAs). These peptides were employed only as cancer biomarkers due to their high expression in cancer tissues, but they also contain non-negligible expressions in healthy tissues.

The sources of these enigmatic, non-canonical peptides may be examined through the immunopeptidomes (Istrail et al., 2004). An immunopeptidome is a collection of recognizable peptides presented by the MHC molecule. The immunopeptidome from healthy human tissue is primarily composed of degraded proteins (retirees), defective ribosomal products (DRiPs), or short-lived proteins (SLiPs). The presentation of these peptides in normal cells represents the health of the cell and allows T-cells to determine whether a cell requires mediated removal and recycling (Dersh, Hollý, and Yewdell, 2021).

Interestingly, the majority of DRiPs are found from non-canonical translation events. DRiPs can arise from non-canonical translation initiation sites at the 'CUG' and other near-cognate (single-nucleotide difference from canonical 'AUG') start codons, or from non-canonical translation initiation factor 2A (EIF2A) instead of EIF2α-GTP-Met-tRNAiMet complex (Starck et al., 2012). Further, non-canonical translation initiation has been shown to be significantly enhanced by stress, such as viral infections or environmental distress. The enhanced non-canonical translation preferentially generates peptides from the 3' to 5' untranslated regions (UTR) of the mRNA as well as alternative reading frames within the known coding regions (Starck et al., 2016). Despite the contribution of non-canonical translation to the antigenic immunopeptidome, the event constitutes a small fraction of the total cellular translation. The apparent paradox highlights a key feature of immunosurveillance, a process in which the host immune system recognizes and eliminates tumors – that the classical computational proteome poorly reflects the landscape of recognizable immunopeptidomes (Yewdell, Dersh, and Fåhraeus, 2019).

Given the rich population of non-canonical translations in the generation of MHC class I binding peptides, carcinogenesis can be thought of as a stressor. Carcinogenesis is the process with which cancer is formed from a normal cell, and is suggested to result from the dysregulation and aberrant translation of irregular peptides under duress for survival (Dersh, Hollý, and Yewdell, 2021).

Compared to healthy cells, which are constantly regulated by proliferative checkpoints, tumors contain mutations that allow for avoidance of these checkpoint signals, resulting in uncontrolled proliferation. The aberrant growths in cancer cells give rise to both biochemical and physical stressors, such as hypoxia, nutrient deficiency, and chronic toxin exposure due to the lack of space and poor ventilation of the environment. In turn, the selective pressure within the TME naturally leads to the accumulation of genetic and epigenetic alterations within the surviving cancer cells. The surviving cancer cells can then evolve with enhanced non-canonical translational events (Sriram, Bohlen, and Teleman, 2018).

To date, only a handful of studies have described targeting tumor immunosurveillance via non-canonical antigens. Despite successful personalization of non-synonymous mutation neoantigens targeting TSAs from various tumors (melanoma and glioblastoma), the computational prowess for identifying non-synonymous mutations, and subsequently, the neoantigens, falls short in malignancies with fewer mutations (Sahin et al., 2017; Hilf et al., 2019).

Current Approaches in TSAs and TAAs Discovery

As non-canonical translational events are enhanced in physically and biologically stressed environments, tumors generated from these translational events provide insight into new-generation of antigen discoveries. The multi-level non-canonical translational events in tumors suggested TAAs as another viable target for cancer immunotherapy. Non-canonical TAAs have the potential to be much less expressionally present in normal tissues compared to cancer tissues since non-canonical transcriptional events evidently occur less frequently in healthy cells.

These TAAs arise mostly from genetic and epigenetic amplification or post-translational modification of peptides, which can originate from both the canonical and non-canonical translational frames. They also have the tendency for expression that are higher and preferential to cancer cells, which create another potential source for cancer neoantigen that could effectively target cancer cells without synonymous mutations most often associated with TSA landscape.

Despite the theoretical lowered specificity for cancer cell targeting compared to TSA-based neoantigens, TAA-based antigen discoveries have been reported to have a much greater volume and discoverable landscape. This increased discoverable landscape improves the likelihood that the antigens can be shared amongst patients and thus, reduces the cost and time for neoantigen vaccine productions. These alternative translational events have hinted at the presence of other types of neoantigens.

It has been surmised that some classes of neoantigens should provoke greater immunogenic responses than others (Hodge et al., 2020). For example, a TSAs arising from SNV mutation is characterized by a single-nucleotide polymorphism, which can have highly similarity to the wild-type peptide and conform to the supposed immunogenicity of the antigen will have narrow-impact personalization as cancer neoantigen vaccine. On the other hand, a more exotic class of neoantigens immunopeptidome not arising from mutations but from non-canonical translating events lacking in the normal, healthy proteome could result in a broader-scale immunogenic effect.

Further, despite the synonymous SNV/INDELS classes of neoantigens being classified as 'personal' (the mutation is specific to the person's tumor), other classes of neoantigens, such as driver neoantigens, may be shared amongst patients. However, non-canonical neoantigens arising from other sources were previously overlooked due to difficulties in the detection and identification of their origins. However, insights into other potential sources of neoantigens previously described in the literature (1.) gene fusions (Wei et al., 2019), 2.) splice sites creation (Jayasinghe et al., 2018), 3.) alternative splicing (Kahles et al., 2018), 4.) intron retention (Smart et al., 2018), 6.) non-coding RNA (Laumont et al., 2018) from 5' to 3' UTR (Chong et al., 2020), and 7.) RNA editing and modifications (Christofi and Zaravinos, 2019) have been recognized as an important field of study for diversifying the pool for neoantigen identifications within the tumor genetic landscape.

Advances in next-generation sequencing (NGS), MS, and novel computational tools initiated the extensive search for novel sources of neoantigens. Laumont and colleagues have successfully utilized an alignment-free RNA workflow ("k-mer profiling) to identify immunogenic neoantigens from non-coding transcripts from RNAseq data in various cancer cells or neoantigen identifications within the tumors' genetic landscape. K-mer profiling has vastly improved mapping precision and variant calling accuracy over the classical MS searches and traditional alignment-based methods (Laumont et al., 2018). The translated peptides from the RNA transcriptome from k-mer profiling have been validated through MS for presentation match on MHC class I and further selected for in vivo studies. Specifically, the study has identified 1,875 MHC-peptides on CT26 (murine colon carcinoma) and 783 on EL-4 (murine lymphoma) cell lines, showing a largely expanded immunopeptidome over the SNV/INDELS computational pipelines (Laumont et al., 2018).

Another study suggested intron-retention to be a source of cancer neoantigen. Potential neoantigens derived from tumor-specific introns have been computationally identified from RNAseq and validated via MHC class I immunopeptidome MS data from various cancer cell lines (Smart et al., 2018). Failure of splicing machineries or splice-site mutation cause introns to be included in mature mRNA. The superposition of tumor-specific introns neoepitope suggests that the aberrant splicing events in tumor models can generate abnormal transcripts, which can be translated into immunogenic, MHC-bound peptides. In fact, Smart and colleagues have shown that the identified intron retention events in melanoma tissue were able to increase the neoantigen load up to 70% compared to neoantigens of the same tissue solely considering the canonical somatic mutations.

Potential Sources of Novel Neoantigens

Realistically, TSAs and TAAs discovered through mutations within the canonical reading frames make up only 1 percent of the patients' genome (Lachmann et al., 2018; Chong et al., 2020). It is believed that the rest of the genome, which is the non-coding regions, contains unusable 'junk DNA' lacking instructions for cellular protein synthesis. However, the idea that shared TSAs and TAAs may exist beyond the canonical translation frame was suggested in three independent reports (Andreev et al., 2015; Gerashchenko et al., 2012; Starck et al., 2016). These reports reveal that specific stress conditions can shut down the canonical translation, all the while increasing the non-canonical 5'UTR translation of stress-privileged transcripts. Furthermore, reports of the effect of pro-inflammatory cytokines, such as the type I interferon and tumor necrosis factor α (TNF-α), show an increased number of potential non-canonical translations (Prasad et al., 2016).

Researchers continue to discover the unexplored potential for TSAs and TAAs discoveries within these non-coding regions, finding evidence of transcription and translation events independent of genetic mutations (Barvík et al., 2017, Laumont et al., 2018). As the identification of non-canonical antigens remains an elusive area of research with little consensus over the most viable, most immunogenic sources for TSAs and TAAs, the search for more powerful computational algorithms and MS variant calling techniques continue (Laumont et al., 2018). The additional landscape from non-canonical transcription and translation events further presses for the need of an antigen discovery pipeline with greater flexibility.

As antigen repertoire grows, traditional sequencing and MS variant calling workflows require a renewed understanding of cancer biology to utilize the non-canonical reading frames for immunotherapy successfully.

While the standardized approach to neoantigen discovery relies on the identification of the non-synonymous mutation in tumors, the Thesis utilizes a 'hybrid', analysis of traditional RNA processing algorithms with *de novo* recognition of isoforms of transcripts that exists outside of the capture range of traditional algorithms. The novel “hybrid *de novo*” pipeline was formulated while considering the various and potentially unknown mechanisms from which non-canonical antigens may arise.

A *de novo* concept of antigen discovery is not new. However, given evidence that TSAs and TAAs can originate from elusive sources, the ‘hybrid’ concept of our discovery pipeline makes for an improved tool for capturing wide ranges of antigens and isoforms of previously less-known sources both canonical- and noncanonically. In order to investigate the correlation between tumor formation, growth, and escape with non-canonical transcription and translation events, we have generated three separate computational methods for discoveries. These differing methods comprise the overall novel hybrid *de novo* pipeline, which is based on multimodal prediction models combined with traditional mapping of the transcriptome to pre-existing genome annotations.

The novel hybrid *de novo* pipeline is a tool that may be key to translating alternative-source TSAs and TAAs discoveries into a more practical clinical setting. Since current vaccine-to-patient turnaround time remains poor due to laborious experimental and regulatory barriers, stratification of cancer vaccine proposed a more practical solution to clinical translation.

In contrast to targeting tumor-specific, private TSAs and TAAs, the concept of ‘shared’ immunogenic antigens from non-canonical sources makes cancer vaccine more applicable to a wider range of patients. Despite shared antigens being an underexplored topic, with shared TSAs being considered impossible, the vast alternative transcriptional and translational landscape which contains overlapping regions can prove to be promising (Laumont et al., 2018). The shared TSAs and TAAs found could be central for generating high-precision, high-stratification cancer treatments. The shared TSAs and TAAs can be further applied systemically for global-scale computational proteogenomic antigen discoveries to deliver ‘generic’ shared antigen-based cancer vaccines.

The discovery of shared TSAs and TAAs with the novel hybrid *de novo* pipeline may also reveal the underlying mechanism in which cancers thrive. The aberrant transcriptional and translational events, the dysregulation of proliferative checkpoints, and the ability for immune escape may be explained through exploring the expression of these antigens in cancer.

Chapter II.

Material and Methods

For the scope of this Thesis, we explore the use of the novel hybrid *de novo* pipeline in discovering shared TSAs and TAAs amongst cancer tissue samples of various lung squamous cell carcinoma patients from the Cancer Genome Atlas (TCGA) database. As mentioned previously, while SNVs and INDELs can generate personalized neoantigens, which are likely to be TSAs, traditional antigen discovery pipelines lack the ability to process aberrant peptides from alternative sources (both canonical and non-canonical). The novel hybrid *de novo* pipeline has the ability to capture a much wider range of these aberrant peptides. As a proof-of-concept for the wide-capture range of the novel hybrid *de novo* pipeline, we have generated 'global' (healthy) and 'local' (cancer-specific) databases. Each database contains its own sets of peptides from canonical and non-canonical sources. In addition, peptides from the 'local' database are cross-examined with those within the 'global' database to identify for TSAs and TAAs with strong expressions.

Both databases are generated from the total RNA sequencing data of multiple tumor- and health-tissue atlases (TCGA, NCBI, and ONCOBOX). Since the data used in this study contain human genetic information, they are protected under the NIH-controlled access data download and management. Reports of data storage, security and privacy protocols are submitted to the dbGaP committee prior to data approval. Data download permission was granted via the eRA Commons platform. Pre-anonymized patient data are further encrypted and de-identified to provide local security. RNA sequencing data were classified into normal and tumor and stored separately.

Section I: Generation of Global and Tissue-specific Reference Databases

I.I Sources of Human Medullary Thymic Epithelial Cells (mTECs)

For this study, we have elected the use of healthy medullary thymic epithelial cells (mTECs) to represent the repertoire of T-cell recognizable self-antigens in the cancer differential expression studies. In the human immune system, mTECs represent a set of unique stromal cell populations within the thymus. These specialized cells are key to establishing lymphatic central tolerance and T-cell maturation, a process in which any self-reactive immature T-cells are eliminated or redirected. Since mTECs can recognize self- and non-self antigens from semi-random somatic rearrangements of the TCRs (via VDJ rearrangements), they are believed to contain the largest collection of T-cell recognizable self-antigens (Larouche et al., 2020).

In this step, raw RNA sequencing data (as FASTQ files) from mTECs were obtained via various databases (NCBI-ANTE and ONCOBOX) with access permission following the sequence read archive (SRA) guidelines from NCBI servers. Despite some of the data being open-sourced, all data were anonymized based on dbGaP's two-step removal of personal information and identifiers. Each sample is given a study-specific identification number and stored in an individualized directory within the Quality Network Appliance Provider (QNAP).

I.II Preparation of Data Security for the ‘Global’ Human Tissue Atlas

In addition to the mTECs self-antigen repertoire, a comprehensive ‘global’ database containing multitudes of healthy human tissues was also generated. The need for this large-scale global transcriptome database is due to the ambiguity in a cancer’s true primary site. Even though the cancer model used in this study has a primary site in the lung, the true sites of origin for each cancer sample may vary due to events of metastasis, clinical diagnosis, and removal. When combined with the mTECs transcriptome repertoire, the global database provides for a more comprehensive and complete outlook of what is considered ‘normal’ in T-cell mediated immune response against cancer.

The Global Human Tissue Atlas (global atlas) is a multi-tissue transcriptome database generated from 22 types of healthy human tissues via the novel hybrid *de novo* pipeline. In this data preparation step, raw RNA sequencing data (as FASTQ files) from NCBI-ANTE and ONCOBOX were obtained with access permission following the SRA guidelines from NCBI servers. Our global atlas, including the mTECs, contains a total of 168 samples: adrenal gland (6 samples), bladder (5 samples), bone marrow (11 samples), brain (9 samples), cervix (4 samples), colon (12 samples), esophagus (11 samples), kidney (8 samples), liver (11 samples), lung (8 samples), mammary gland (5 samples), medullary thymic epithelial cells (mTECs) (3 samples), ovary (4 samples), pancreas (8 samples), prostate (6 samples), skeletal muscle (6 samples), skin (6 samples), small intestine (9 samples), stomach (15 samples), thyroid gland (6 samples), tonsil (7 samples), uterus (2 samples), and whole blood nuclear cell (WBNCs) (6 samples). Despite some of the data being open-sourced, all data were anonymized based on dbGaP's two-step removal of personal information and identifiers. Each sample is given a study-specific identification number and stored in an individualized directory within the Quality Network Appliance Provider (QNAP).

I.III Quality Control of Input RNA Sequences

RNA sequencing has become one of the most transient sources of data for biotechnology, engineering, and bioinformatics for the purpose of transcriptome profiling, and in our case, novel transcript and peptide discoveries. Due to the intrinsic design of current high-throughput Next-generation Sequencing (NGS) technologies, RNA sequences' quality may be compromised for their throughput, thus, requiring vigorous quality control (QC) procedures to ensure reproducibility in experimental results (Sheng et al., 2017). Two common issues in raw RNA sequencing data are the contamination of samples with genomic information from another species, and RNA-seq-specific issues such as ribosomal RNA (rRNA) residual, RNA degradation, and varying read coverages (Zhou et al., 2018).

In order to ensure clean data for analysis, RNA sequencing results (in FASTQ file format) were subjected to QC based on (1) sequencing-quality assessment and trimming and (2) detection of internal contaminants, and (3) detection of external contaminants using RNA-QC-chain following default settings (Figure 3). RNA-QC-chain utilizes a three-sequential workflow that first trims low sequencing-quality reads, which is followed by an rRNA filter, in which rRNA fragments are identified, extracted, and used to further identify the contaminating species, and lastly, multiple metrics are provided for the evaluated FASTQ data. The trimmed outputs of RNA-seq data are used for downstream processing.

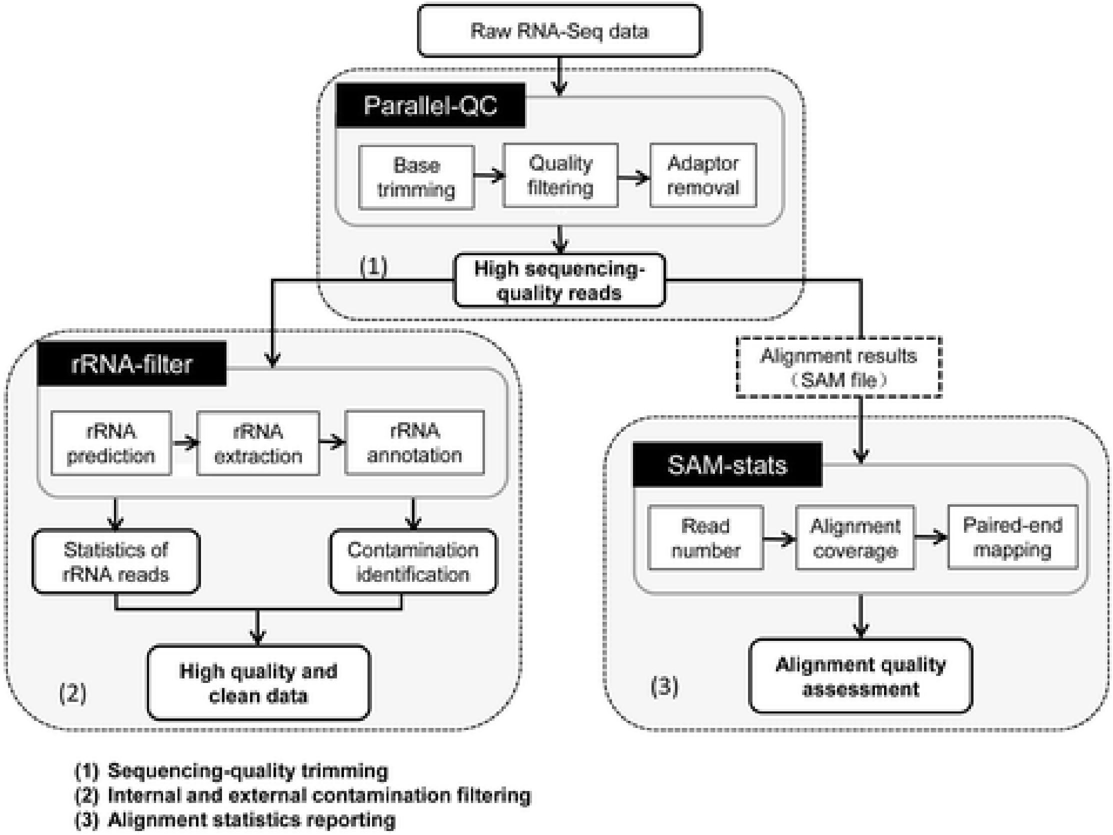
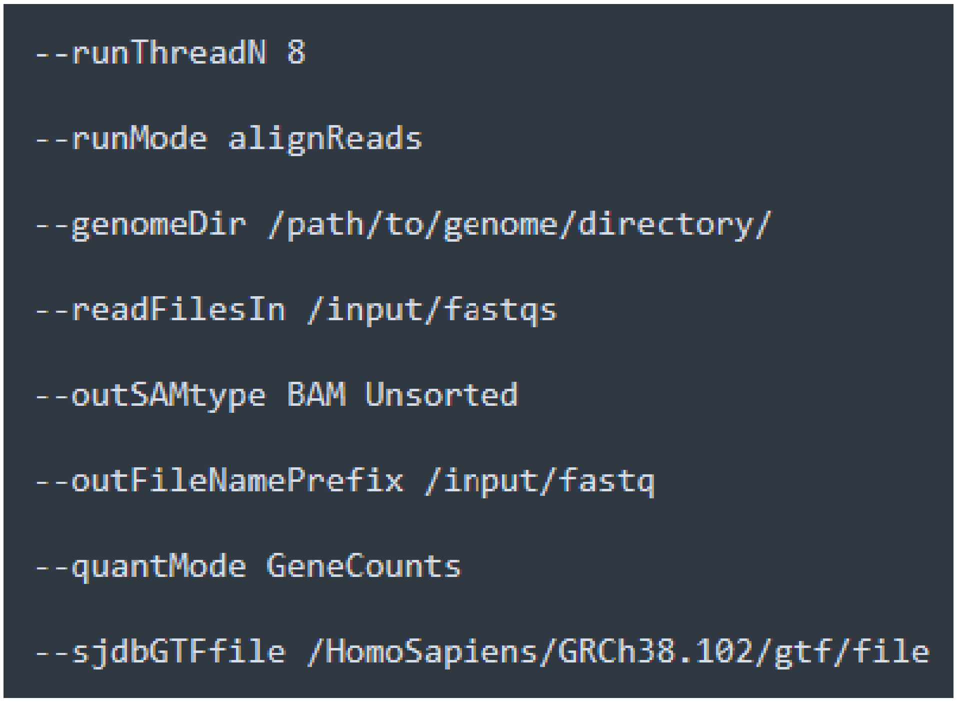

Raw RNA-Seq data
Parallel-QC
Base trimming
Quality filtering
Adaptor removal
(1)
High sequencing-quality reads
rRNA-filter
rRNA prediction
rRNA extraction
rRNA annotation
Statistics of rRNA reads
Contamination identification
(2)
High quality and clean data
Alignment results （SAM file）
SAM-stats
Read number
Alignment coverage
Paired-end mapping
(3)
Alignment quality assessment
(1) Sequencing-quality trimming
(2) Internal and external contamination filtering
(3) Alignment statistics reporting

Figure 3: The workflow of RNA-QC-chain

(Zhou et al., 2018)

I.IV RNA Sequence Alignment with the Human Genome

In order to determine the position from which the RNA sequencing read has originated from within the human genome, the RNA reads are aligned to the reference genome using the STAR (Spliced Transcript Alignment to a Reference) aligner (Dobin et al., 2013). The STAR alignment was executed with STAR version 2.7.9a on an Intel® Xeon® CPU E5-2630 v4, 2.20 GHz, 98 GB RAM supercomputer under the Linux CentOS7 operating system with modified settings (Figure 4).



```
--runThreadN 8

--runMode alignReads

--genomeDir /path/to/genome/directory/

--readFilesIn /input/fastqs

--outSAMtype BAM Unsorted

--outFileNamePrefix /input/fastq

--quantMode GeneCounts

--sjdbGTFfile /HomoSapiens/GRCh38.102/gtf/file
```

Figure 4: Settings of the STAR aligner for NCBI-ANTE and ONCOBOX read alignment *(Techachakrit, 2022)*

STAR aligner is an RNA-seq alignment tool designed by Alexander Dobin and colleagues to align non-contiguous sequences directly to the reference genome. In contrast to traditional alignment methods, which are short-read mappers which align short reads to a database or splice junctions, STAR aligner utilizes their two-steps algorithms, namely (1) the seed search and (2) the clustering/stitching/scouring step (Dobin et al., 2013). During seed search, the STAR aligner sequentially searches for a maximal mappable prefix (MMP) by way of identifying the read sequence *R*, the read location *I,* and the reference genome *G*. The MMP is thus defined as the longest substring (maximal mappable length (MML)) of the read sequence at location *i* that matches exactly one or more substrings of *G*.

Equation 1: Maximal Mappable Prefix (MMP)

$$R_i, R_{i+1}, \ldots, R_{i+MML-1}$$

The first seeds identified were mapped to a donor splice site; then, the MMP search is repeated again in the unmapped portion for an acceptor splice site.

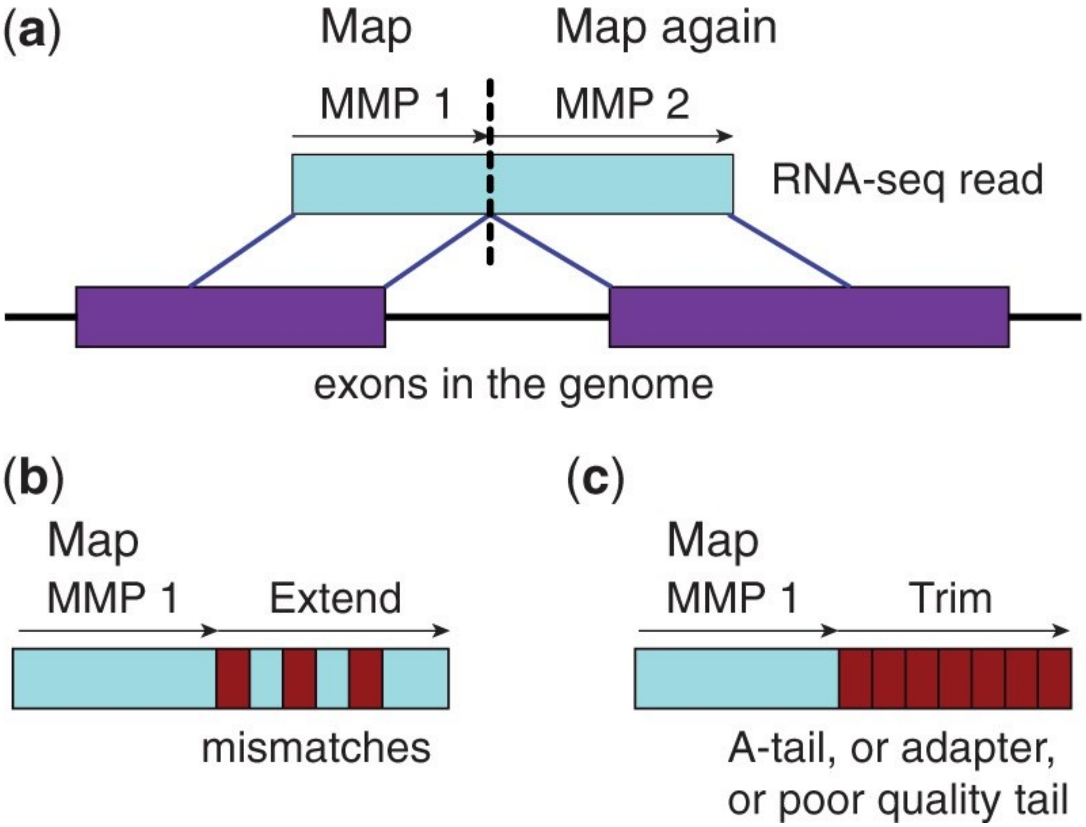


Figure 5: A Schematic representation of the MMP in the first step of STAR algorithms *(a) splice site junction detection, (b) mismatches, (c) tails. (Dobin et al., 2013)*

The second phase of the STAR aligner utilizes a stitching mechanism to compose together an entire read sequence by attaching all the previously generated seeds together, as shown in Figure 5. In order to stitch together seeds in the proper order, the seeds are clustered together by proximity to select a set of anchor seeds, which other seeds were stitched on to. This stitching mechanism represents a single sequence, allowing for possible genomic gaps and overlaps between the inner ends of the reads. All seeds that are mapped within a genomic window around the anchors are then stitched together, with the assumption of a local linear transcription model. This approach greatly increases the sensitivity of the STAR aligner as compared to other alignment tools (Figure 6).

Due to its high-throughput turn-around rates, we have elected to use STAR aligner as our aligning tool. In an attempt to generate the global atlas containing nearly a terabyte of data, STAR aligner excels in both speed and sensitivity. The computational tool exhibits the lowest false-positive rates with high sensitivity when compared with other RNA sequencing alignment algorithms in the field, such as TopHat2 (Kim et al., 2013), GSNAP aligner (Wu et al., 2010), RUM aligner (Grant et al., 2011), and MapSplice (Wang et al., 2010) as shown in Figure 6.

The GRCh38.102 human genome assembly and gene annotation were obtained as FASTA files and GTF files, respectively, from the open-source Ensembl genome database at the European Bioinformatics Institute (Yates et al., 2020). Each NCBI-ANTE and ONCOBOX sample was searched and mapped with the most extended matching sequence within the reference genome and searched again for unmapped regions of reads. The separate MMPs were then stitched together to generate complete read alignment for each sample.

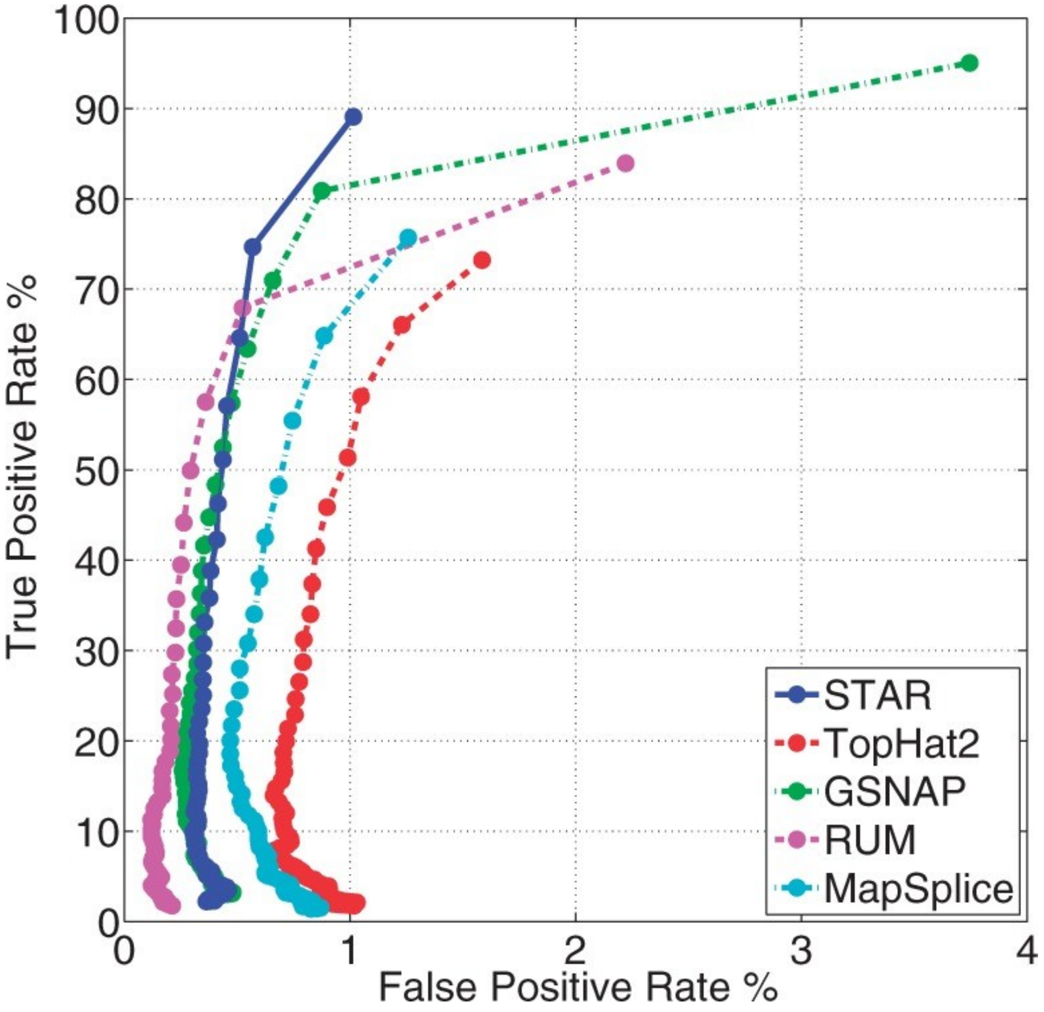
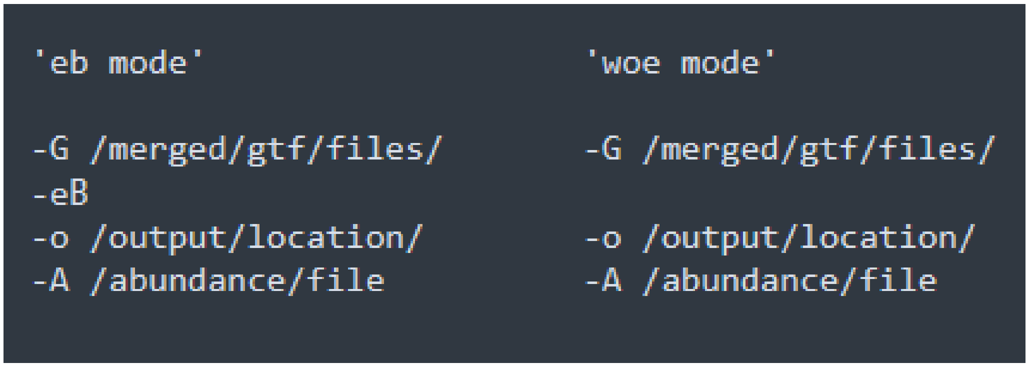


Figure 6: Comparison of true-positive rates vs. false-positive rates of differing tools

Comparison of true-positive rates (as a percentage) vs. false-positive rates (percentage) for simulated RNA-seq data using STAR, TopHat2, GSNAP, RUM, and MapSplice tools. (Dobin et al., 2013)

I.V RNA Read Assembly and Merging

The resulting spliced read alignments, outputted as unsorted binary BAM files, were sorted based on reference position using SAMtools and assembled via StringTie. SAMtools is a set of utilities which facilitates interactions with DNA and RNA read alignments in the SAM, BAM and CRAM formats (Li et al., 2009). To pass the unsorted BAM files into StringTie for read assembly and merging, SAMtools version 1.11 with subcommand ‘sort’ was performed under default settings. Following sorting, StringTie, a high-efficiency read assembler version 2.1.4 (Pertea et al., 2015) was used under default settings to perform the primary assembly. The GTF results from the primary assemblies were merged to form a global reference for the secondary *de novo* assembly. The secondary assemblies were performed with and without the expression estimation mode (‘eb’ and ‘woe’ respectively), which limits the processing of read alignments to estimate the coverage of transcript with the merged global reference GTF (Figure 7).



```
'eb mode'                     'woe mode'

-G /merged/gtf/files/         -G /merged/gtf/files/
-eB
-o /output/location/          -o /output/location/
-A /abundance/file            -A /abundance/file
```

Figure 7: StringTie secondary assemblies, following the ‘eb’ and ‘woe’ protocols *(Techachakrit, 2022)*

StringTie is a transcript assembling tool that can simultaneously assemble reads and estimate their expression simultaneously. StringTie first groups the reads into clusters, which are then turned into splice graphs. The splice graphs are then used in iterative extraction of the heaviest path, construction of flow network, and computation of maximum flow for abundance estimation (Figure 8).

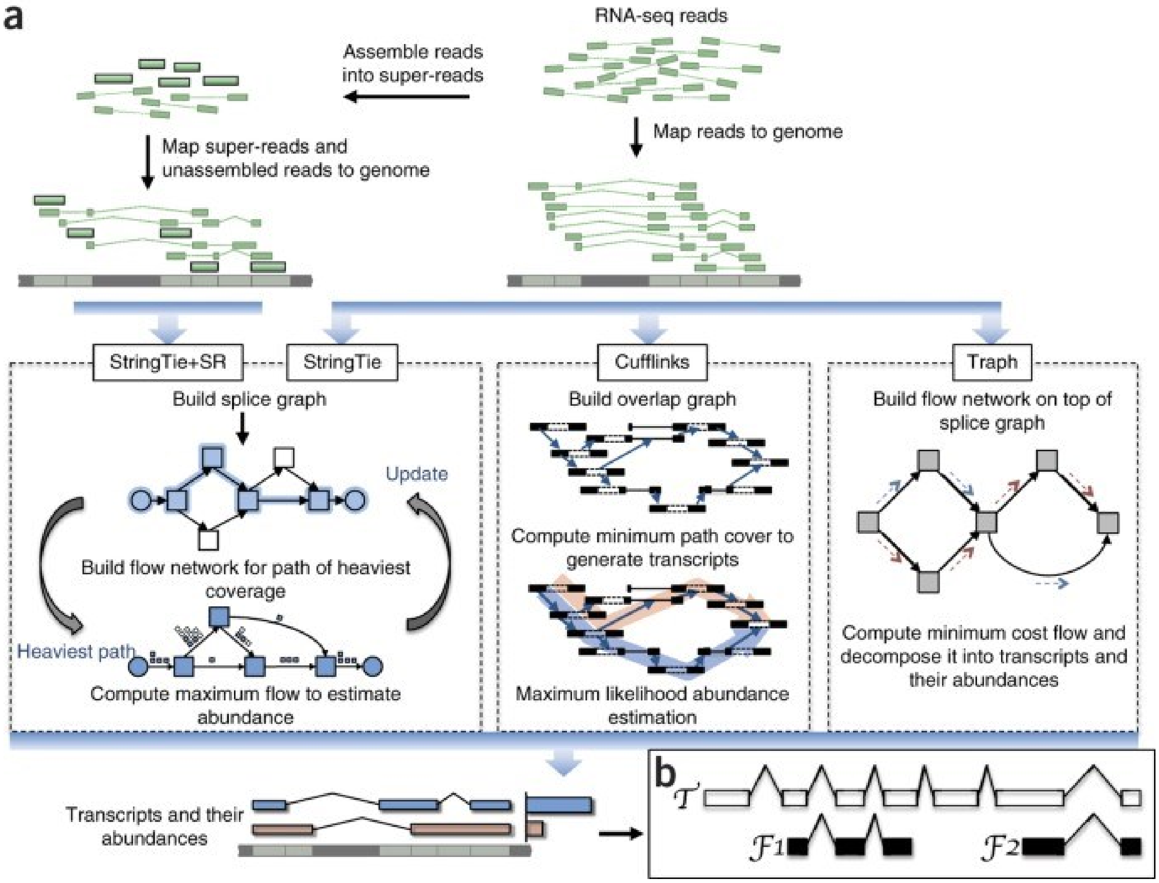

a
RNA-seq reads
Assemble reads into super-reads
Map super-reads and unassembled reads to genome
Map reads to genome
StringTie+SR
StringTie
Cufflinks
Traph
Build splice graph
Update
Build flow network for path of heaviest coverage
Heaviest path
Compute maximum flow to estimate abundance
Build overlap graph
Compute minimum path cover to generate transcripts
Maximum likelihood abundance estimation
Build flow network on top of splice graph
Compute minimum cost flow and decompose it into transcripts and their abundances
Transcripts and their abundances
b
$\mathcal{T}$
$\mathcal{F}1$
$\mathcal{F}2$

Figure 8: Overview of StringTie algorithm in comparison to Cufflink and Traph *(Pertea et al., 2015)*

StringTie is used to assemble both of the canonical and non-canonical transcriptomes resulting from STAR alignment and is the basis of the novel hybrid *de novo* pipeline (Figure 9). First, a reference-based, human genome-guided assembly is used to reconstruct the read alignments to the genome, identifying clusters of reads representing the potential transcripts that have been previously annotated. Following the canonical transcriptome reconstruction, the STAR alignment reads are also assembled under the *de novo* transcriptome reconstruction, using the previously merged transcriptome library to avoid redundant isoforms. This multi-passes processing of StringTie generate a transcriptome that is fully missing from the annotated reference.

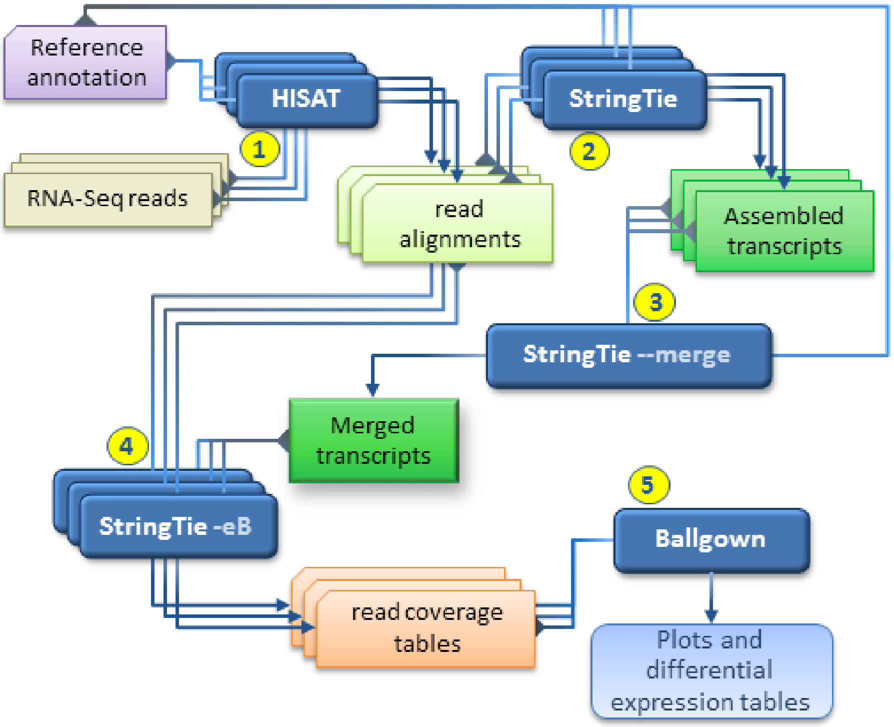


Figure 9: A schematic workflow for a general tandem StringTie-based *de novo* analysis *(Pertea et al., 2015)*

When combined with the two-step algorithms generated by tandem StringTie runs, we generate the output of the novel hybrid *de novo* transcriptome. The assembly allows for comprehensive identification of all transcripts present in a sample, including annotated genes, novel isoforms of annotated genes, and novel genes. The resulting transcriptomes represent the libraries of each RNA sequencing sample. These transcriptomes are merged to generate the transcriptome database, which considers redundant transcript structures across all samples.

I.VI Construction of Transcriptome-based Global Proteome Database

The newly generated global transcriptome database was annotated with GFFcompare following default settings to identify each transcript's relationship to the reference genome. GFFcompare is a utility tool that can be used to compare, merge, annotate and estimate the accuracy of GTF and GFF files as compared to the reference genome (Pertea et al., 2020).

Following annotations, the Transcriptome database was segregated into clusters based on their annotated output of strandedness using in-house python scripts: 1.) convertGTF_nt_sample_wise.py, 2.) GTF2FilteredList-eb.py, 3.) GTF2FilteredList-woe.py, and 4.) seqkit.py (see Appendix 2, 3, 4, and 5 respectively), then translated based on its annotations for translation (forward, reverse, six-frames) using in-house python scripts: translate_allORF.py and GTF2TPM (see Appendix 6 and 7 respectively), resulting in six final outputs (eb-pos, eb-neg, eb-unk, woe-pos, woe-neg, woe-unk, respectively). The scripts were generated on Python 3.8.9. The six clusters of final outputs are then merged to generate a single normal tissue master FASTA file containing all ORFs from all the samples in the global transcriptome-based proteome database.

For the scope of this Thesis, we have kept all of the translated results, as any possible reading frames may be present in tumors. In order to deal with the large datasets, the most likely reading frames were flagged for each transcript based on researching peptides via BLASTP for assistance in prioritization. The resulting master FASTA file was then converted into a Python dictionary in tandem with its transcript per million (TPM) values. The Python dictionary format allows for the large quantity of files to be crossed search for unique and non-unique results.

I.VII Construction of Tissue-specific Transcript Expression Database

To identify the downstream tissue-specific transcript expressions in cancer cells, we constructed a database of tissue-specific expression levels for the transcript identified (section I.V). The transcripts were quantified with Salmon version 1.8.0 following the default protocol for single- or paired-end (dependent on the particular sample's origin of RNA sequencing). Salmon is a transcript quantification tool that uses probabilistic two-phase inference algorithms, consisting of light-weight mapping followed by and online and offline phases which estimate and refine the resulting expression levels (Patro et al., 2017). The resulting nucleotide FASTA files (step I.V) were merged to act as a reference for the quantification. Duplicates within the nucleotide master FASTA file were marked and removed, and the master FASTA file was converted into a reference library with human genome GRCh38.p13 as its reference. The reference library was quantified against the original FASTQ files for read abundances. For the scope of this Thesis, the read abundances were not used in an isoform-level differential expression study but can be applied in tandem with computational tools such as Wasabi and Sleuth in R.

Section II: Identification of Sufficient Sample Size

II.I Sample-level Transcript Rarefication

In order to deliver an accurate global and tissue-specific database for the discovery of non-canonical tumor neoantigens, we calculated the number of samples required to capture substantial gene expression for analysis using our in-house python script: unique_counter.py (see Appendix 1). The Python script 'unique_counter.py' uses the rarefaction technique to subsample on the basis of sampling with replacement.

Rarefaction is a technique used to access species richness from random samplings in the field of ecology. The technique was developed by Howard L. Sanders in 1968. The calculation allows for the estimation of species richness for a given number (N) of samples in the sample size. The resulting rarefaction curves are plots of the number of species as a function of N samples, which grow at a steep tangent first before reaching a plateau where saturation is achieved (Sanders, 1968).

By applying the rarefaction technique to our transcript sampling, we are able to generate a saturation curve, estimating transcript homogeneity based on random sampling. By randomly sampling lower depths (known as subsampling) of specific numbers of RNA sequencing experiments and plotting the sample coverage of specific genes as a rarefaction curve, the saturation position would identify the numbers of samples required for a complete representation of sample coverage within the particular tissue. The information gained from these saturation points allowed us to understand the variability of transcript expression across samples from the same tissue. This has been utilized to identify as many unique transcripts as possible while distinguishing between false positives and rare transcripts, which may not be consistently detected in all samples.

Equation 2: Rarefaction estimation for the expected number of transcript species E(S)

N = total sample size

S = number of transcript species

n = standard sample size for comparison

N = number of individuals in the i

$$E(S) = \sum_{i=1}^{S} \left(1 - \left[\frac{\binom{N - N_i}{n}}{\binom{N}{n}} \right] \right)$$

Where:

$$\binom{N - N_i}{n} = \frac{(N - N_i)!}{(n!)([N - N_i] - n)!}$$

And:

$$\binom{N}{n} = \frac{N!}{n!(N - n)!}$$

Section III: Quantitative Analysis of Individual Tumor Samples

III.I Preparation of Data Security of the ‘Local’ Cancer Tissue Atlas

The ‘local’ cancer tissue atlas is a transcriptome database containing lung cancer transcripts processed by the novel hybrid *de novo* pipeline. The local cancer atlas represents all discoverable coding and non-coding transcripts from the cancer sample set within the limit of the novel hybrid *de novo* pipeline.

In this study, 100 patient-derived RNA sequencing data (as BAM files) of lung squamous cell carcinoma were obtained from the TCGA-LUSC project. In this data preparation step, raw BAM files from TCGA-LUSC were obtained with access permission following the dbGaP guidelines via the eRA Commons platform. The pre-anonymized data were re-encrypted and de-identified for local storage in an individualized QNAP. Additional cancer atlases were generated for experimental purposes, but were used only as rarefaction models within this study. These atlases include the TCGA-KIRP (cervical kidney renal papillary cell carcinomas) and the TCGA-KIRC (renal clear cell carcinomas), following the same protocol.

III.II Cancer RNA alignment with STAR aligner

In the TCGA database, the patients’ FASTQ files had been pre-aligned by the data donors in order to further improve data security resulting in downloadable BAM files. TCGA uses STAR aligner to align the cancer RNA reads to the human reference genome (version GRCh38.d1.vd1).

III.III RNA Read Assembly and Merging

The resulting spliced read alignments, outputted as unsorted binary BAM files, was sorted based on reference position using SAMtools and assembled via StringTie. SAMtools version 1.11 with subcommand ‘sort’ were performed under default settings. Following sorting, a primary assembly was performed with StringTie version 2.1.4 under default settings. The GTF results from the primary assemblies were merged to form a global cancer reference for the secondary *de novo* assembly. The secondary assemblies were performed with and without the expression estimation mode (‘eb’ and ‘woe’ respectively), which limits the processing of read alignments to estimate the coverage of transcript with the merged global reference GTF, similarly to step I.V. (Figure 7).

The resulting transcriptomes (TCGA-LUSC, TCGA-KIRP, and TCGA-KIRC) represent the libraries of each RNA sequencing sample. These transcriptomes are individually merged to generate the transcriptome database, which considers redundant transcript structures across each and all samples.

III.IV Construction of Transcriptome-based Global Proteome Database

The newly generated global transcriptome database was annotated with GFFcompare following default settings to identify each transcript's relationship to the human reference genome. Following annotations, the transcriptome databases were segregated into clusters based on their annotated output of strandedness using in-house python scripts: 1.) convertGTF_nt_sample_wise.py, 2.) GTF2FilteredList-eb.py, 3.) GTF2FilteredList-woe.py, and 4.) seqkit.py (see Appendix 2, 3, 4, and 5 respectively).

The resulting segregations were then translated based on thier annotations for translation (forward, reverse, six-frames) using in-house python scripts: translate_allORF.py and GTF2TPM (see Appendix 6 and 7 respectively), resulting in six final outputs (eb-pos, eb-neg, eb-unk, woe-pos, woe-neg, woe-unk). The scripts were generated on Python 3.8.9. The six clusters of final outputs were merged to generate a single normal tissue, master FASTA file containing all ORFs from all the samples in the global transcriptome-based proteome database.

For the scope of this Thesis, we have kept all of the translated results, as any possible reading frames may be present in the cancer samples. In order to deal with the large datasets, the most likely reading frames were flagged for each transcript based on researching peptides via BLASTP for assistance in prioritization. The resulting master FASTA file was then converted into a Python dictionary in tandem with its TPM values. The Python dictionary format allows for a large quantity of files to be crossed search for unique and non-unique results.

III.V Construction of Tissue-specific Transcript Expression Database

To identify the downstream tissue-specific transcript expression in cancer cells, we constructed a database of tissue-specific expression levels for the transcript identified in section III.IV. The transcripts were quantified with Salmon following protocols in step I.VII. The resulting nucleotide FASTA files (step III.IV) were merged to act as a reference for the quantification process. Duplicates within the nucleotide master FASTA file were marked and removed, and the master FASTA file was converted into a reference library with human genome GRCh38.p13 as its reference.

The reference library was quantified against the original FASTQ files for read abundances. For the scope of this Thesis, the read abundances were not used in an isoform-level differential expression study, but again, can be applied in tandem with computational tools such as Wasabi and Sleuth in R.

Section IV: Meta-analysis of Global Tumor Neoantigen Database

IV.I Identification of shared ORFs within TCGA-LUSC

In order to identify tumor-specific antigens in TCGA-LUSC, we utilized an in-house Python script: TPM_dict-LUSC.ipynb (see Appendix 8). TPM_dict-LUSC.ipynb was run on Jupyter Notebook version 6.2.0 (Kluyver et al., 2016), a web-based interactive computing platform based on Python, to generate the TCGA-LUSC master peptide dictionary ('the lung cancer dictionary').

The lung cancer dictionary contains nested information of each ORF, the transcript identified, the identity of origin in regards to the human genome, the patient derivation, and the cancer classification (the TCGA-LUSC in this case). By list comparisons in Python, we extracted all non-unique ORFs presented amongst the patient samples within the lung cancer dictionary. These non-unique ORFs are candidates for shared TSAs and TAAs, since they are shared amongst multiple patients within the sampling set. The resulting non-unique ORFs were extracted. The extracted results were parsed through a TPM check filter (TPM > 1) to assure sufficient expression before being counted and stored individually. The identified ORFs were later clustered in a transcript-wise fashion based on their transcript ID.

IV.II Identification of shared, TCGA-LUSC-specific ORFs

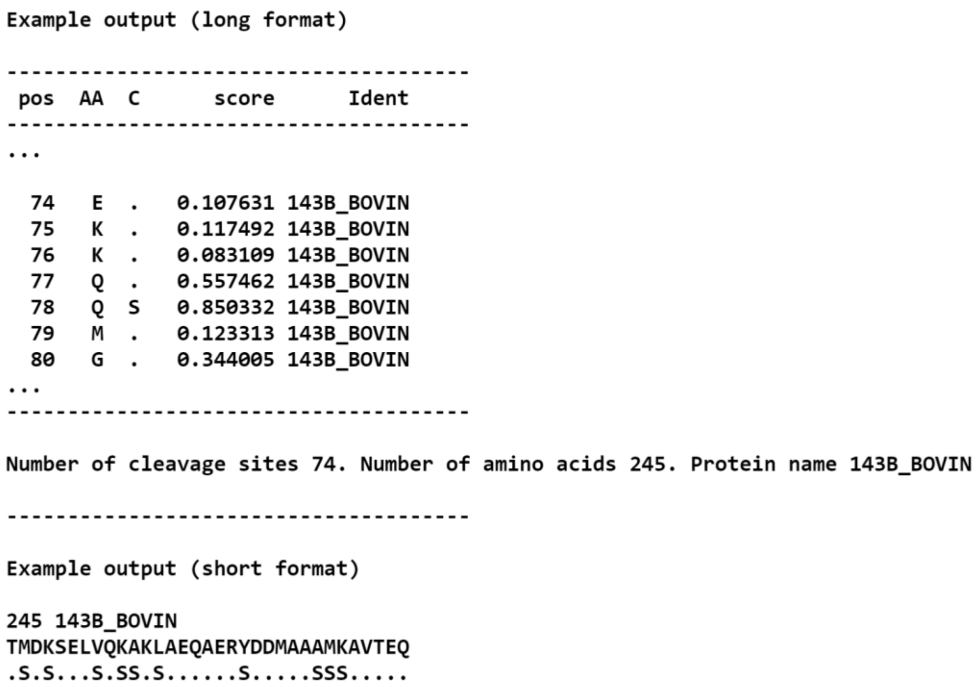
To identify ORFs which are specific to the sample set from TCGA-LUSC, we utilized the in-house Python script: TPM_dict-ONCOBOX.ipynb (see Appendix 9). TPM_dict-ONCOBOX.ipynb was run on Jupyter Notebook version 6.2.0 to generate the healthy tissue (NCBI-ANTE and ONCOBOX-based) master peptide dictionary ('the healthy tissues dictionary').

The healthy tissues dictionary contains nested information of each ORF, the transcript identified, the identity of origin in regards to the human genome, the patient derivation, and the tissue of origins (amongst the 22 tissues the ORFs were generated from). By using two lists comparison in Python, we cross-searched all non-unique ORFs resulting from section IV.I with the newly generated healthy tissue dictionary. Any shared ORFs that are unique to the lung cancer dictionary are considered candidates for TSAs, while those that are also present in the healthy tissues dictionary are considered candidates for TAAs. The resulting non-unique ORFs were separately extracted. The extracted results were parsed through a TPM check filter (TPM > 1) to assure sufficient expression before being counted and stored as 'TSA candidates' and 'TAA candidates', respectively. The identified ORFs were later clustered transcript-wise based on their transcript ID.

IV.III Analysis for potential antigens based on shared ORFs (TSA- and TAA-candidates)

The shared TSAs and TAAs candidates identified in section IV.II were further analyzed for their potential as neoantigenic peptides. Namely, the ORFs undergo *in silico* predictions for proteasomal cleavages based on the human proteasome using NetChop version 3.1 on the web-based server under threshold settings of 0.5.

NetChop is a web server- and command-line Linux-based neural network proteasome cleavage prediction tool, which was trained on human data, including novel sequence encoding scheme and 1260 publicly available MHC class I ligands (C-terminus cleavage sites) for improved prediction accuracies (Kesmir et al., 2002; Nielsen et al., 2005). NetChop intakes peptide FASTA files, which are the shared ORFs (TSA- and TAA-candidates) in this study. The identified cleaved peptides were outputted as tables containing residue numbers, amino acids, assigned prediction, prediction scores, and sequence names (Figure 10).



```
Example output (long format)

--------------------------------------
 pos  AA  C      score      Ident
--------------------------------------
...

   74   E  .   0.107631 143B_BOVIN
   75   K  .   0.117492 143B_BOVIN
   76   K  .   0.083109 143B_BOVIN
   77   Q  .   0.557462 143B_BOVIN
   78   Q  S   0.850332 143B_BOVIN
   79   M  .   0.123313 143B_BOVIN
   80   G  .   0.344005 143B_BOVIN
...
--------------------------------------

Number of cleavage sites 74. Number of amino acids 245. Protein name 143B_BOVIN

--------------------------------------

Example output (short format)

245 143B_BOVIN
TMDKSELVQKAKLAEQAERYDDMAAAMKAVTEQ
.S.S...S.SS.S......S.....SSS.....
```

Figure 10: Example output for NetChop version 3.1

(Kesmir et al., 2002; Nielsen et al., 2005)

Chapter III.

Results

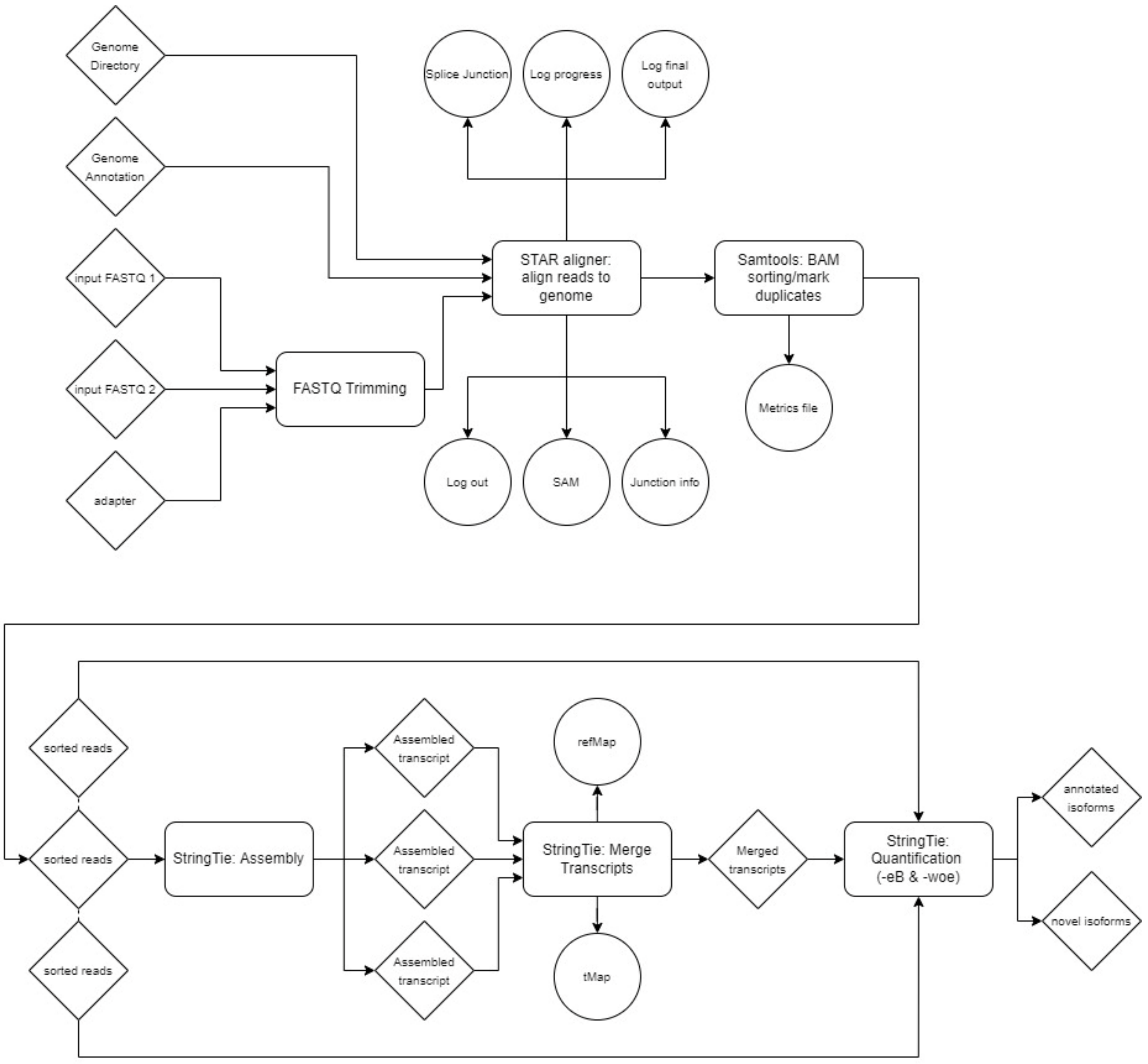


Figure 11: Workflow for the Novel Hybrid *de novo* Pipeline

The novel Hybrid de novo pipeline involved multi-step workflows of well-known computational tools, such as the STAR aligner and StringTie. (Techachakrit, 2022)

Section I: Design and rationale of the novel Hybrid *de novo* transcript reconstruction Pipeline for novel isoform discovery

Attempts to computationally predict/discover TSAs via non-synonymous mutations faced the exceedingly high false discovery and low discovery rates, which resulted in the TSAs discovery being dependent on system-level molecular MHC peptide repertoire via the high-throughput MS/MS studies (Laumont et al., 2018). Even so, the current tandem MS (MS/MS) studies are still reliant on a user-defined protein database, most often generated from proteogenomics strategies in order to match the acquired MS/MS spectrum to peptide sequences.

Ideally, for the discovery of TSAs, the transcriptome-based proteome database generated from RNA sequences that matched to these MS/MS spectra should contain all potential peptides, both annotated (canonical) and unannotated ones (non-canonical). Only by considering novel transcript isoforms that exist in a tumor or a cluster of tumors, which later would be translated into peptides, can we discover the extent of TSAs.

Due to limitations with tandem MS software tools, current tools used in MS/MS analyses are unable to compute large search spaces created by translating all RNA sequencing reads in all reading frames. Thus, we devised the novel hybrid *de novo* pipeline (Figure 11) as a strategy for generating database for proteogenomic analysis, in order to re-investigate the non-canonical transcriptional frames to infer transcript structures from the mapped reads with and without the absence of previously annotated transcript structures.

By using a *de novo* transcript reconstruction strategy in tandem with the canonical transcriptome reconstruction, we can identify transcript clusters that are from canonical and non-canonical sources, and thus, comprehensively characterize the TSAs landscape coded from all genomic regions.

In the first computational round of the pipeline, a canonical transcriptome is generated based on the alignment of normal and tumor RNA sequencing to the human genome and genome annotations provided by Ensembl (GRCh38.p13), which are then assembled into a transcriptome. Reads which are contained within exons are aligned with the reference genome, while reads that are spanned between splice junctions are aligned with gaps to the reference human genome. This first round of assembly results in multiple transcriptomes containing canonical (pre-annotated) transcripts that are specific to each input RNA sequencing sample.

The second computational round of the pipeline involved using the merged results of the canonical transcriptomes (merged transcriptome) as an assembly reference for the *de novo* transcriptome reconstruction. The unbiased approach allows for comprehensive identification of all transcripts within the sample by excluding those that had been previously identified via canonical annotations (redundant transcript structures). While the common gene/transcript databases may contain some novel isoforms, this second step ensures a complete transcriptome identification from any genomic regions within the experimental samples.

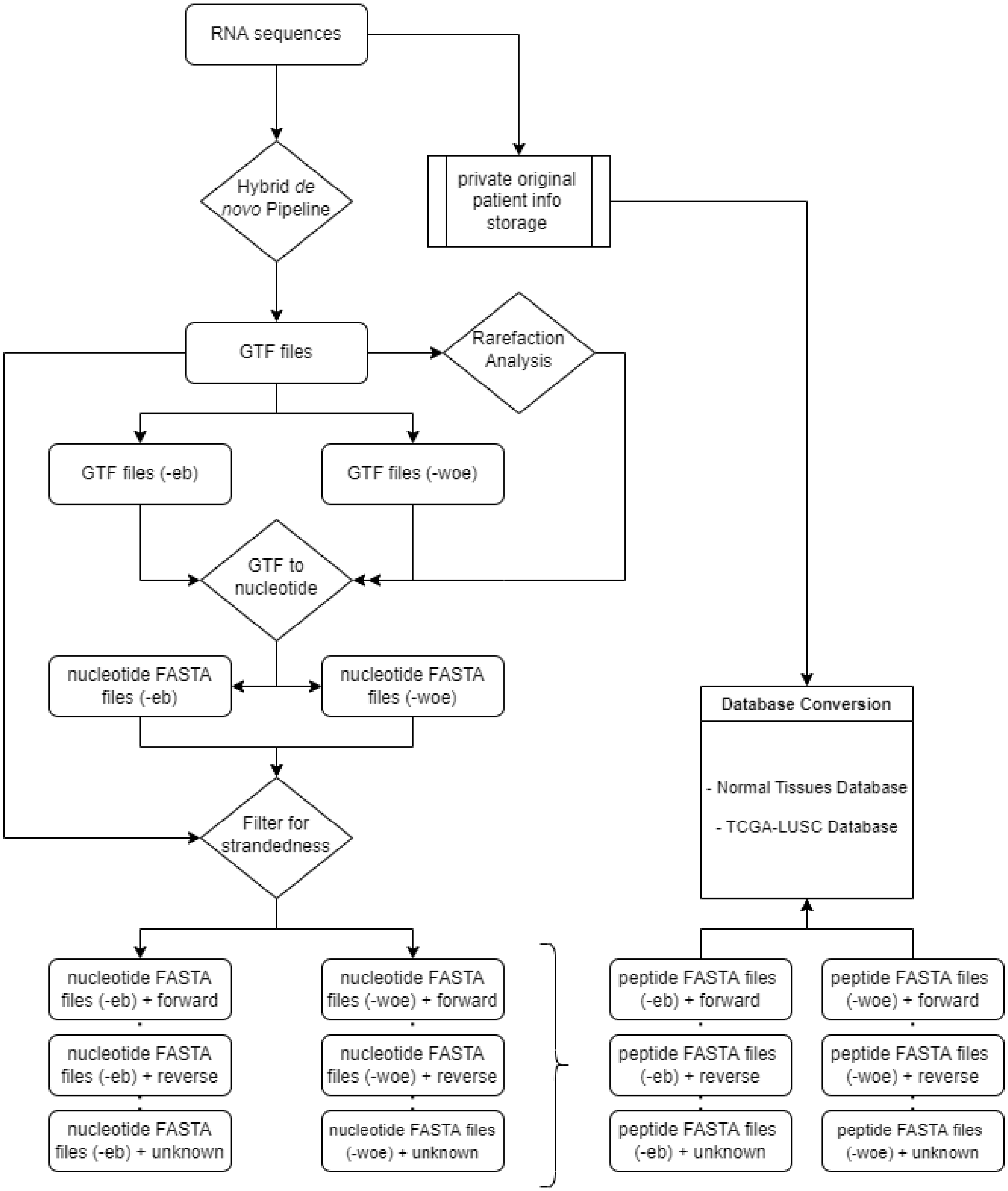

RNA sequences
Hybrid de novo Pipeline
private original patient info storage
GTF files
Rarefaction Analysis
GTF files (-eb)
GTF files (-woe)
GTF to nucleotide
nucleotide FASTA files (-eb)
nucleotide FASTA files (-woe)
Database Conversion
- Normal Tissues Database
- TCGA-LUSC Database
Filter for strandedness
nucleotide FASTA files (-eb) + forward
nucleotide FASTA files (-eb) + reverse
nucleotide FASTA files (-eb) + unknown
nucleotide FASTA files (-woe) + forward
nucleotide FASTA files (-woe) + reverse
nucleotide FASTA files (-woe) + unknown
peptide FASTA files (-eb) + forward
peptide FASTA files (-eb) + reverse
peptide FASTA files (-eb) + unknown
peptide FASTA files (-woe) + forward
peptide FASTA files (-woe) + reverse
peptide FASTA files (-woe) + unknown

Figure 12: Post-processing steps of the discovered data via novel Hybrid *de novo* Pipeline

The discovery of databases involved post-processing of the obtained data in order to clean up and extract the necessary transcripts, and subsequently, in silico translated peptide. The workflow shows the steps involved with our in-house python script and Salmon quantification resulting in the final databases. (Techachakrit, 2022)

Section II: Post-processing analysis for the generation of Transcriptome-based Proteome Databases

To generate interpretable data from the novel hybrid *de novo* analyses, we have generated a Python-based sub-workflow which convert the resulting GTF files into cross-searchable peptide dictionaries on Jupyter Notebook. The post-processing pipeline has streamlined the procedure in generating the peptide dictionaries in both healthy tissues and cancer. The generated sub-workflow successfully provided easy-access to our translated ORF data across both canonical and non-canonical platforms.

The GTF file format or the general transfer file format are files resulting from the StringTie output, and are one of the final outputs of the novel hybrid *de novo* pipeline. The GTF file contains nine total fields, which are tab-separated in a Dataframe-like structure. These fields are (1) seqname – contains the name of chromosome of scaffold, assigned based on the reference genome during the computational processing, (2) source – the database or project name, (3) feature – whether the particular line is a gene, transcript, or exon, (4) start – the starting position of the particular feature, (5) end – the stopping position of the particular feature, (6) score – a point value assigned to the particular feature, (7) strand – forward, reverse, or unknown, (8) frame – denote the starting codon position, (9) attribute – a semicolon-separated list containing additional information of each feature.

For the purpose of generating the databases, 'the attributes', which contained valuable information such as the gene ID, transcript ID, TPM, coverages, and FPKM must be extracted. However, the attribute length within each sample set can vary thus it cannot be simply expanded to a finite number of cells as would the other fields of the GTF dataframe.

Thus, we have elected to generate a Python function that would "parse" out the individual attributes, in their various lengths, and recombine them with the original Dataframe for downstream processing (see Appendix 1).

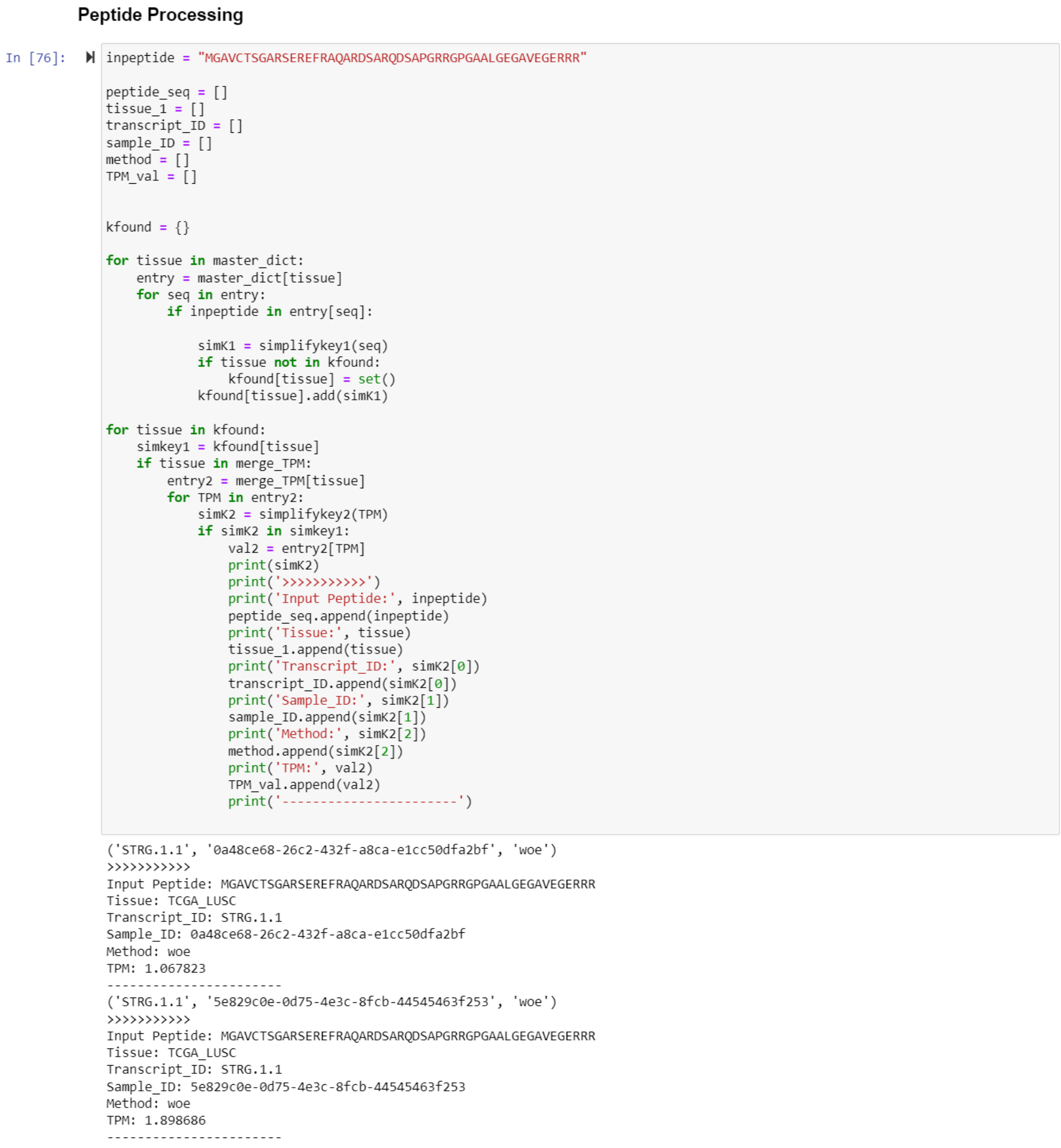
The transcript ID generated in the GTFs were then counted and processed for uniqueness within each sample to generate subsampling data, which were used to generate the rarefaction curves. The rarefaction curves are used to identify the number of samples necessary to generate a relatively homogeneous and reliable database. For the rarefication procedures, we generated an in-house Python script ‘Unique_counter.py’ (see Appendix 1) that will subsample N numbers of the dataset, which were then parsed individually and counted.

Following the identification of a sufficient sample size, we constructed a multi-level GTF conversion tool that extracted the "transcript_ID" of the attribute field, as well as the "seqname," "start," and "end" fields. This information is stored in BED files (browser extensible data), a text file format that is used to store the genomic regions as coordinates and associated annotations. The data are presented in the form of columns separated by tabs. The BED files containing these coordinates were then fed into Bedtools2, in combination with an in-house Python script ‘ConvertGTF_nt_sample_wise.py’ (see Appendix 2) which allowed for coordinate-based extraction of nucleotides based on genomic regions, resulting in nucleotide FASTA files. The ‘ConvertGTF_nt_sample_wise.py’ script contains a 'holder' that considers the exonic coordinate and sequence that has been fed into it, and acts as a stepwise find-and-stitch secondary tool to assure novel isoforms are properly generated without the intronic regions.

In order to decrease the number of false discoveries, we have elected to use a filtration system to segregate the raw nucleotide FASTA files generated via the Bedtools2 and ‘ConvertGTF_nt_sample_wise.py’. We generated another in-house Python script ‘GTF2FilteredList-eb.py’ and ‘GTF2FilteredList-woe.py’ (see Appendix 3 and 4, respectively), which generated a filtered list of each of the GTF lines based on their strandedness. The list of each strandedness (forward, reverse, and unknown) was fed as input into seqkit.py, which utilizes the tool Seqkit with a python automatic execution script ‘seqkit.py’ (see Appendix 5) in order to generate three different nucleotide FASTA files based on each individual input (Figure 12).

The nucleotide FASTA files, now clustered based on their strandedness were then translated accordingly (forward, reverse, and six-frames, respectively) using an in-house Python script ‘translate_allORF.py’ (Appendix 6), which detects the flag extensions on the files and translates them accordingly based on found starting codons combination, resulting in datasets of peptide FASTA files.

The peptide FASTA files were then piped into a Jupyter Notebook-based database Python nested dictionary. In Python, a dictionary is an unordered collection of items, which allows us to add key-to-value pairs of data. A nested dictionary contains another outer layer of key to a key-to-value pair of data. This storage format allows for us to create compact, transferrable, and searchable sets of databases that can search input peptides as values (Figure 13)



```
In [76]:  inpeptide = "MGAVCTSGARSEREFRAQARDSARQDSAPGRRGPGAALGEGAVEGERRR"

          peptide_seq = []
          tissue_1 = []
          transcript_ID = []
          sample_ID = []
          method = []
          TPM_val = []


          kfound = {}

          for tissue in master_dict:
              entry = master_dict[tissue]
              for seq in entry:
                  if inpeptide in entry[seq]:

                      simK1 = simplifykey1(seq)
                      if tissue not in kfound:
                          kfound[tissue] = set()
                      kfound[tissue].add(simK1)

          for tissue in kfound:
              simkey1 = kfound[tissue]
              if tissue in merge_TPM:
                  entry2 = merge_TPM[tissue]
                  for TPM in entry2:
                      simK2 = simplifykey2(TPM)
                      if simK2 in simkey1:
                          val2 = entry2[TPM]
                          print(simK2)
                          print('>>>>>>>>>>>')
                          print('Input Peptide:', inpeptide)
                          peptide_seq.append(inpeptide)
                          print('Tissue:', tissue)
                          tissue_1.append(tissue)
                          print('Transcript_ID:', simK2[0])
                          transcript_ID.append(simK2[0])
                          print('Sample_ID:', simK2[1])
                          sample_ID.append(simK2[1])
                          print('Method:', simK2[2])
                          method.append(simK2[2])
                          print('TPM:', val2)
                          TPM_val.append(val2)
                          print('-----------------------')
```

```
('STRG.1.1', '0a48ce68-26c2-432f-a8ca-e1cc50dfa2bf', 'woe')
>>>>>>>>>>>
Input Peptide: MGAVCTSGARSEREFRAQARDSARQDSAPGRRGPGAALGEGAVEGERRR
Tissue: TCGA_LUSC
Transcript_ID: STRG.1.1
Sample_ID: 0a48ce68-26c2-432f-a8ca-e1cc50dfa2bf
Method: woe
TPM: 1.067823
-----------------------
('STRG.1.1', '5e829c0e-0d75-4e3c-8fcb-44545463f253', 'woe')
>>>>>>>>>>>
Input Peptide: MGAVCTSGARSEREFRAQARDSARQDSAPGRRGPGAALGEGAVEGERRR
Tissue: TCGA_LUSC
Transcript_ID: STRG.1.1
Sample_ID: 5e829c0e-0d75-4e3c-8fcb-44545463f253
Method: woe
TPM: 1.898686
-----------------------
```

Figure 13: Jupyter Notebook view of the Peptide Search Algorithms over TCGA-LUSC Python dictionary (Techachakrit, 2022)

The output peptides from the input "MGAVCTSGARSEREFRAQARDSARQDSAPGRRGPGAALGEGAVEGERRR" has resulted in the search over all entries of translational events of all 100 patients, returning two shared peptides within the TCGA-LUSC database, including its method and TPM

Section III: Sample-level transcript variability is greater in tumor samples than healthy control samples

We have elected to compare a sample-level transcript saturation between TCGA-LUSC (The Cancer Genome Atlas: Lung Squamous Cell Carcinoma Project) and the ONCOBOX healthy lung tissues. We have identified samples from 100 individuals from the TCGA-LUSC project, and determined that the number is sufficient to act as a baseline for sample-level transcript homogeneity (Figure 14). The method for this sampling follows a simple subsampling equation (see Method II.I) following gradual rounds of input increase and resampling.

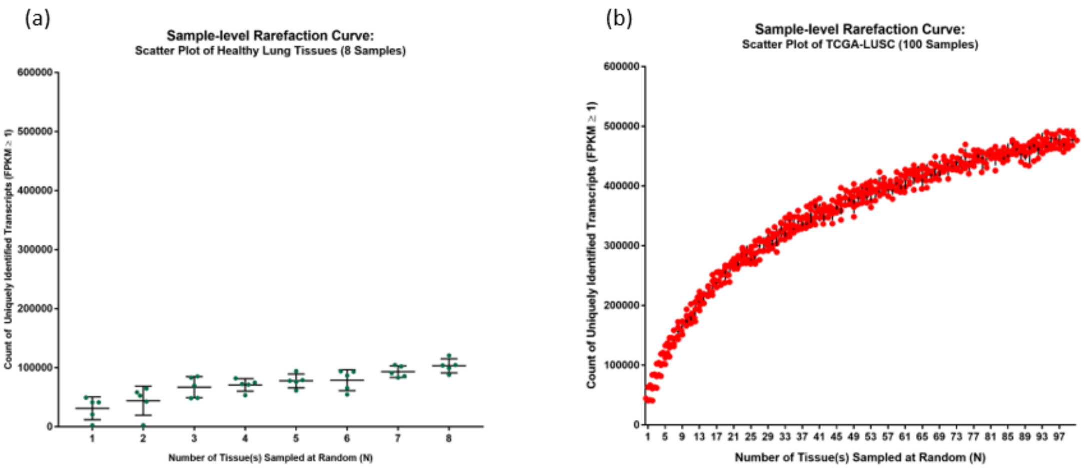


Figure 14: Sample-level transcript Rarefaction Curves

(a) Healthy Lung Tissues (N = 8), and (b) TCGA-LUSC (N = 100). (Techachakrit, 2022)

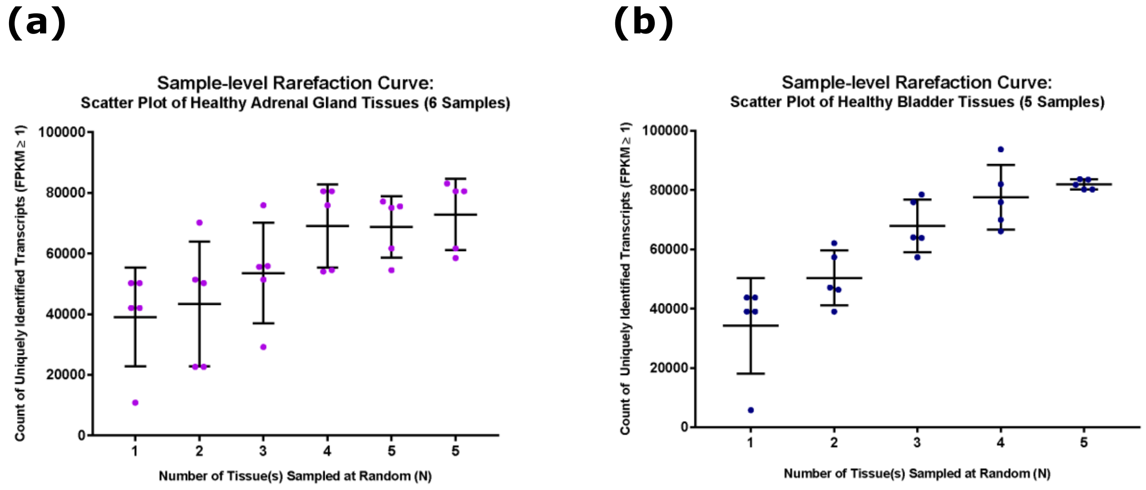
Expectedly, there is an observable difference in the transcript saturation rates between the healthy lung tissues and the cancerous lung tissues. In the rarefaction curve generated from subsampling healthy lung tissues, the highest number of unique transcript counts is 120,982, with the mean maximum unique transcript counts of approximately 103,388. The standard deviation amongst the maximum lung tissue samples (with sampling rounds of 5) is 11,845, with a margin of error of 5,297 unique transcripts.

On the other hand, a much higher number of unique transcripts were identified in the cancerous lung tissues. At 100 samples (the studies' technical limitation), the highest number of unique transcript counts is 485,790, with the mean maximum unique transcript counts of approximately 478,186. The standard deviation amongst the maximum cancerous lung tissue samples (with sampling rounds of 5) is 6,834, with a margin of error of 3,056 unique transcripts.

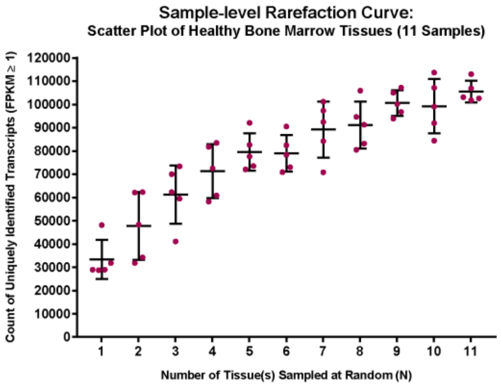
The significant increase in cancer transcript variabilities post-hybrid de novo pipeline canonical and non-canonical transcript discoveries, compared with the healthy tissue transcript discoveries conform with the ideas that: (1) there is a greater variability in cancer cells, and (2) that these variabilities arise from non-mutational, non-canonical sources.

On the other hand, since there are both systemic and technical limitations for usable data, we were unable to perform a one-to-one rarefaction analysis between the healthy and cancerous lung tissues. Ideally, 100 samples from healthy lung tissues should be used to compare for transcript variabilities, however, we were unable to obtain more total RNA sequencing data of the healthy lung tissues at this time.

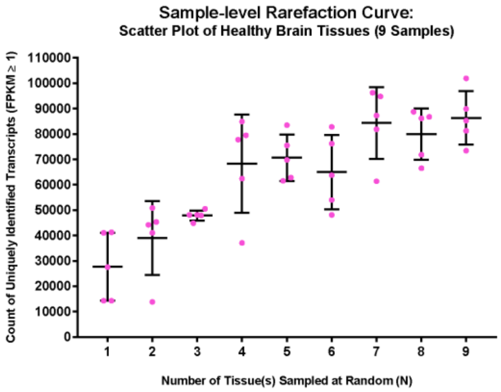
In order to determine the general trend that greatly differentiated the sample-level transcript homogeneity between tumor and normal tissues, we further investigated 22 other normal healthy tissues (a total of 168 tissue samples) from the ONCOBOX repertoire to identify the average trend for normal tissue saturation. Two additional rounds of rarefaction analyses were performed on TCGA-KIRC and TCGA-KIRP, which originated from kidney renal clear cell carcinoma and cervical kidney renal papillary cell carcinoma, respectively, in order to determine the same concept from the perspective of tumor diversity. We have found 8-12 samples to be sufficient for transcript homogeneity in non-tumorigenic, healthy tissues, while at least 100 samples are required for tumorigenic tissues (Figure 15 and 16, respectively).



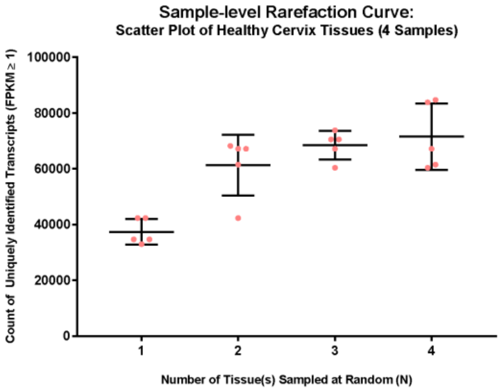
(c)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Bone Marrow Tissues (11 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
Number of Tissue(s) Sampled at Random (N)

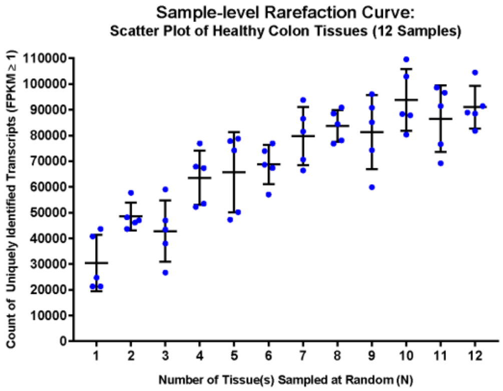
(d)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Brain Tissues (9 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
Number of Tissue(s) Sampled at Random (N)

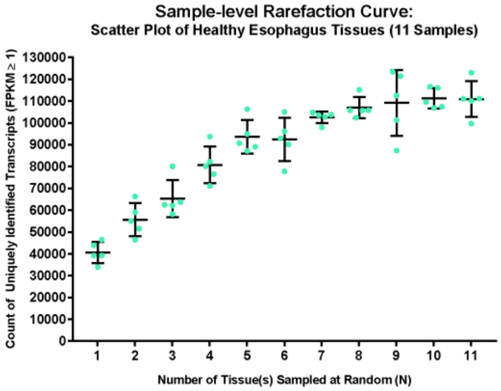
(e)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Cervix Tissues (4 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
100000
80000
60000
40000
20000
0
1
2
3
4
Number of Tissue(s) Sampled at Random (N)

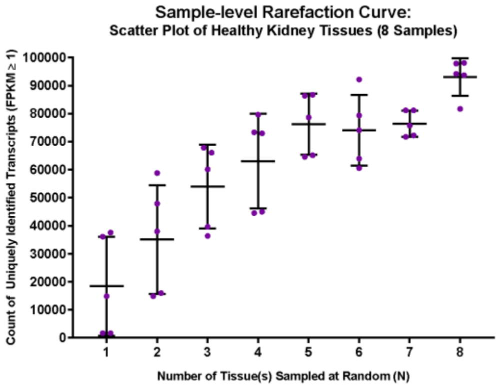
(f)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Colon Tissues (12 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
12
Number of Tissue(s) Sampled at Random (N)

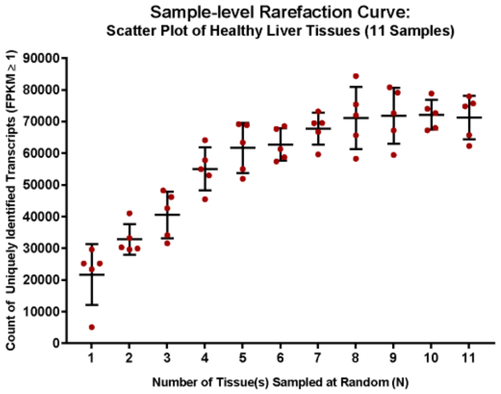
(g)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Esophagus Tissues (11 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
130000
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
Number of Tissue(s) Sampled at Random (N)

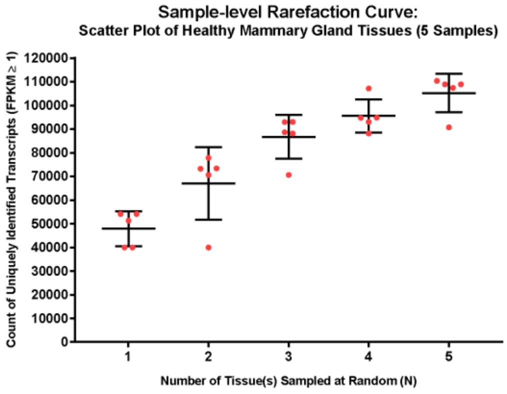
(h)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Kidney Tissues (8 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
Number of Tissue(s) Sampled at Random (N)

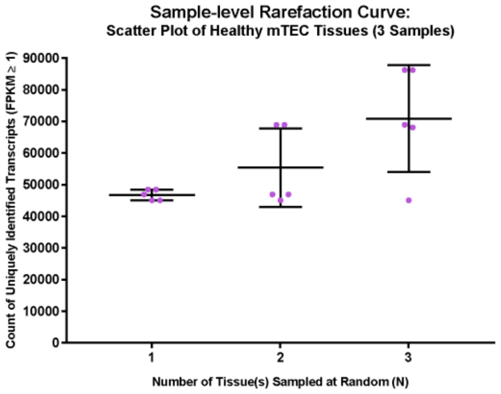
(i)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Liver Tissues (11 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
Number of Tissue(s) Sampled at Random (N)

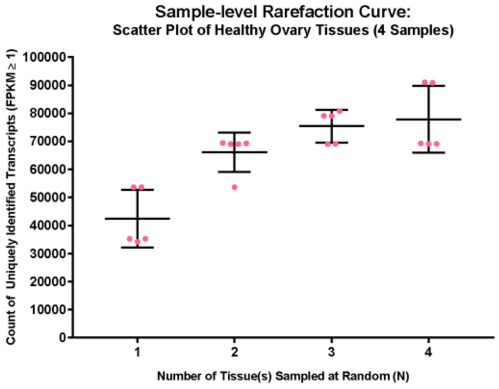
(j)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Mammary Gland Tissues (5 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
Number of Tissue(s) Sampled at Random (N)

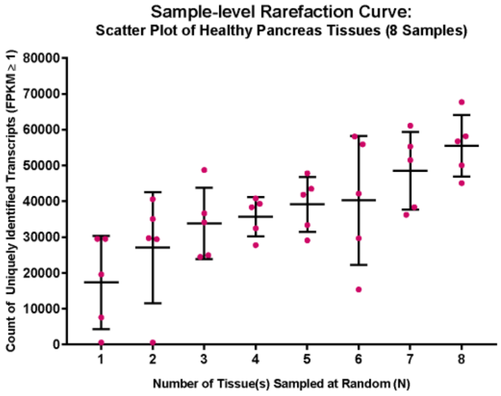
(k)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy mTEC Tissues (3 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
Number of Tissue(s) Sampled at Random (N)

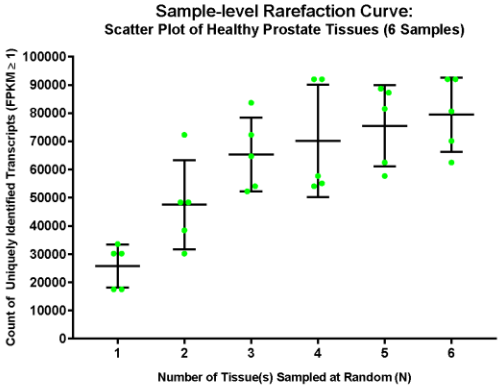
(l)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Ovary Tissues (4 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
Number of Tissue(s) Sampled at Random (N)

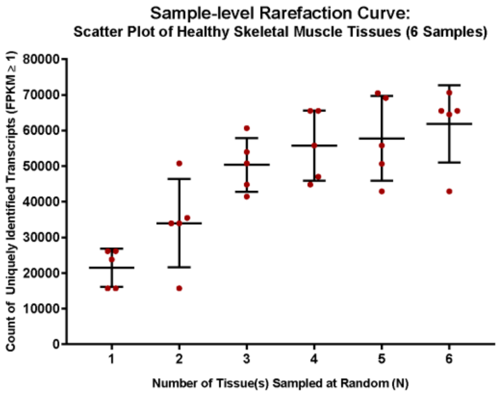
(m)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Pancreas Tissues (8 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
Number of Tissue(s) Sampled at Random (N)

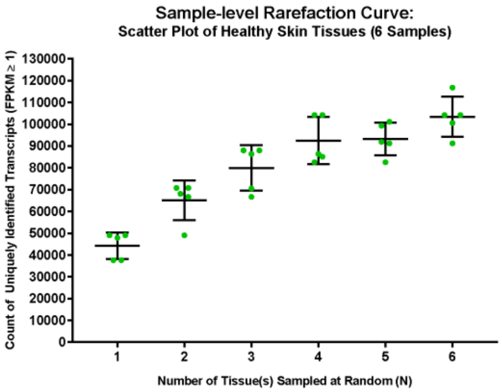
(n)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Prostate Tissues (6 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
Number of Tissue(s) Sampled at Random (N)

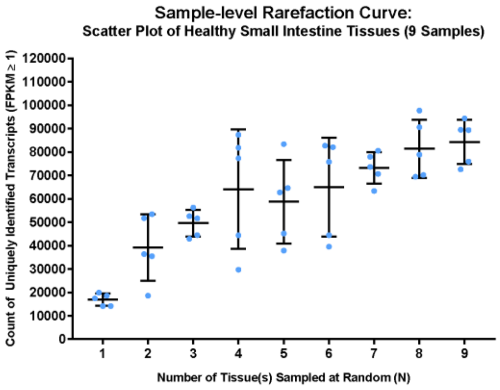
(o)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Skeletal Muscle Tissues (6 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
Number of Tissue(s) Sampled at Random (N)

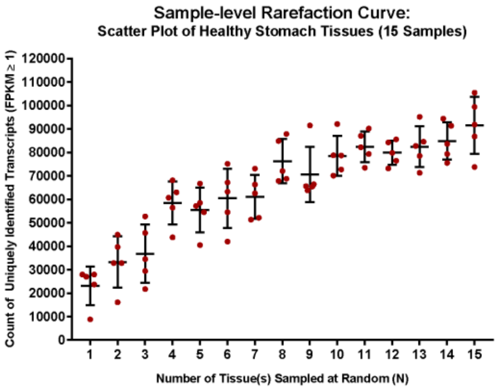
(p)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Skin Tissues (6 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
130000
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
Number of Tissue(s) Sampled at Random (N)

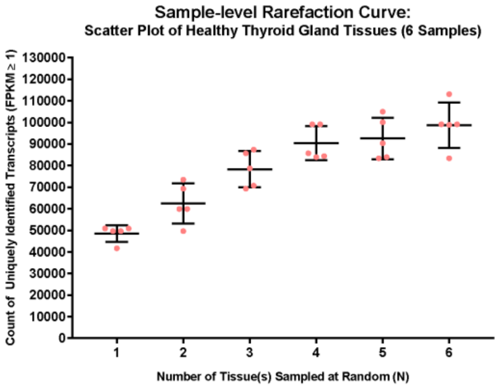
(q)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Small Intestine Tissues (9 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
Number of Tissue(s) Sampled at Random (N)

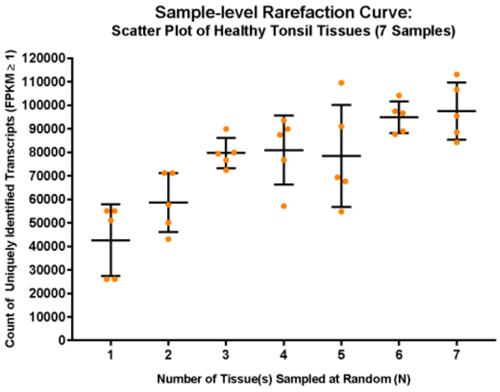
(r)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Stomach Tissues (15 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Number of Tissue(s) Sampled at Random (N)

(s)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Thyroid Gland Tissues (6 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
130000
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
Number of Tissue(s) Sampled at Random (N)

(t)


Sample-level Rarefaction Curve:
Scatter Plot of Healthy Tonsil Tissues (7 Samples)
Count of Uniquely Identified Transcripts (FPKM ≥ 1)
120000
110000
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
1
2
3
4
5
6
7
Number of Tissue(s) Sampled at Random (N)

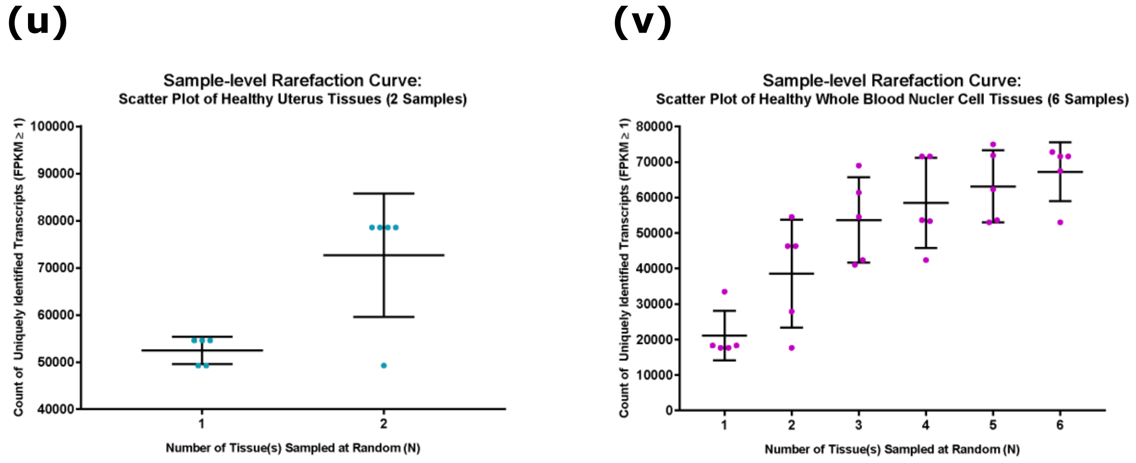


Figure 15: Sample-level Rarefaction Analysis on Various Tissues

(a) Adrenal Gland, (b) Bladder, (c) Bone Marrow, (d) Brain, (e) Cervix, (f) Colon, (g) Esophagus, (h) Kidney, (i) Liver, (j) Lung, (k) Mammary Gland, (l) mTECs, (m) Ovary, (n) Pancreas, (o) Prostate, (p) Skeletal Muscle, (q) Skin, (r) Small Intestine, (s) Stomach, (t) Thyroid Gland, (u) Tonsil, (v) Uterus, (w) WBNCs. (Techachakrit, 2022)

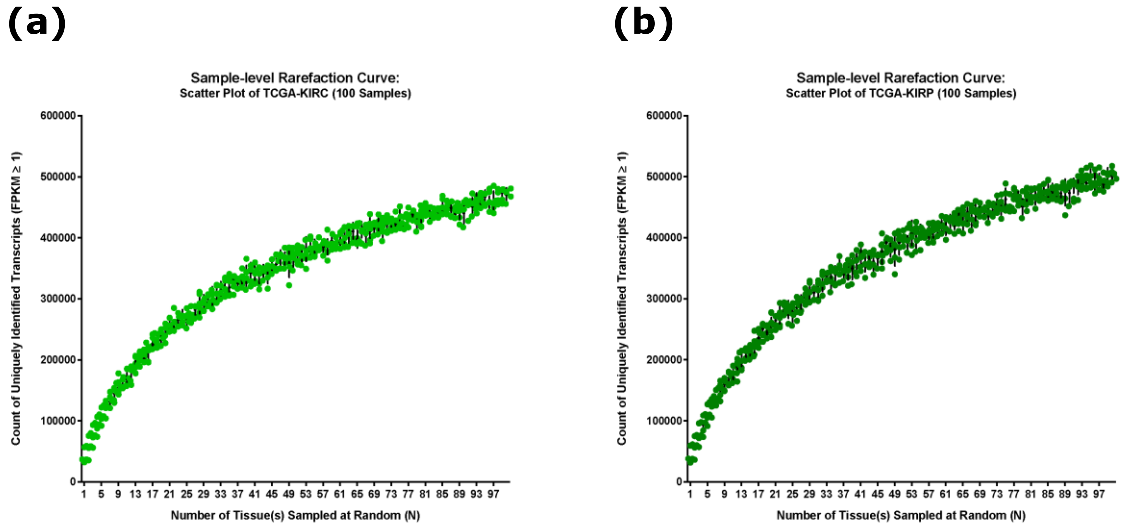


Figure 16: Sample-level Rarefaction Analysis on other TCGA projects

(a) TCGA-KIRP and (b) TCGA-KIRC. (Techachakrit, 2022)

Section III: Non-canonical transcription sites are sources of shared TSAs amongst TCGA-LUSC clusters

Through the Jupyter Notebook dictionary sub-workflows, we have identified 31 ORFs that are shared within the TCGA-LUSC samples, ranging from 8 to 60 amino acids in length. Tracing back, these ORFs originated from 7 distinct transcripts: (1) ENST00000000233.8, (2) ENST00000000412.6, (3) MSTRG.1004.1, (4) MSTRG.10052.1, (5) STRG.1.1, (6) STRG.10.1, and (7) STRG.100.1. Three of the seven transcripts were derived from the "woe" computational procedure (STRG.1.1, STRG.10.1., and STRG.100.1), while two of the seven were derived from the "eb" computational procedure (MSTRG.1004.1 and MSTRG.10052.1), and two of the seven were from the canonical frames (ENST00000000233.8 and ENST00000000412.6). These ORFs are given a unique identification 'ptx' followed by a number ranging from 1 to 31 (Figure 17).

Amongst the 31 ORFs which were found to be shared amongst patients in the TCGA-LUSC sampling set, 20 are considered TSA candidates. The TSA candidates (ptx2, ptx3, ptx4, ptx6, ptx7, ptx9, ptx12, ptx13, ptx16, ptx17, ptx18, ptx20, ptx21, ptx23, ptx24, ptx25, ptx26, ptx27, ptx30, and ptx31) are ORFs that are completely absent from the healthy tissues dictionary (with TPM = 0), and have passed the expression level parsing in the lung cancer dictionary (with TPM > 1).

The TSA candidates have been identified as ORFs originating from the non-canonical transcriptional events (as annotated via StringTie), except for one (ptx2) which originated from canonical sources as a novel isoform.

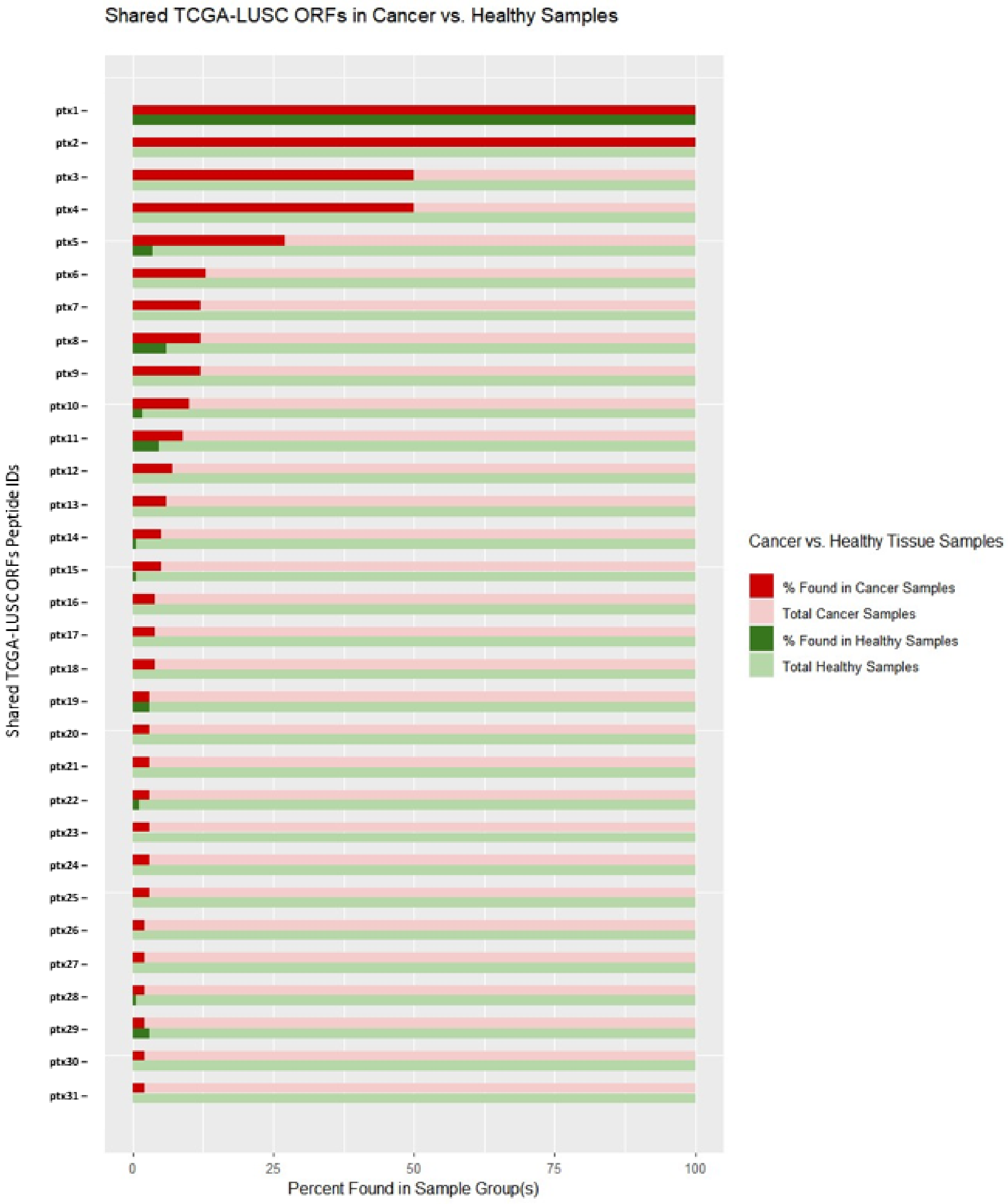
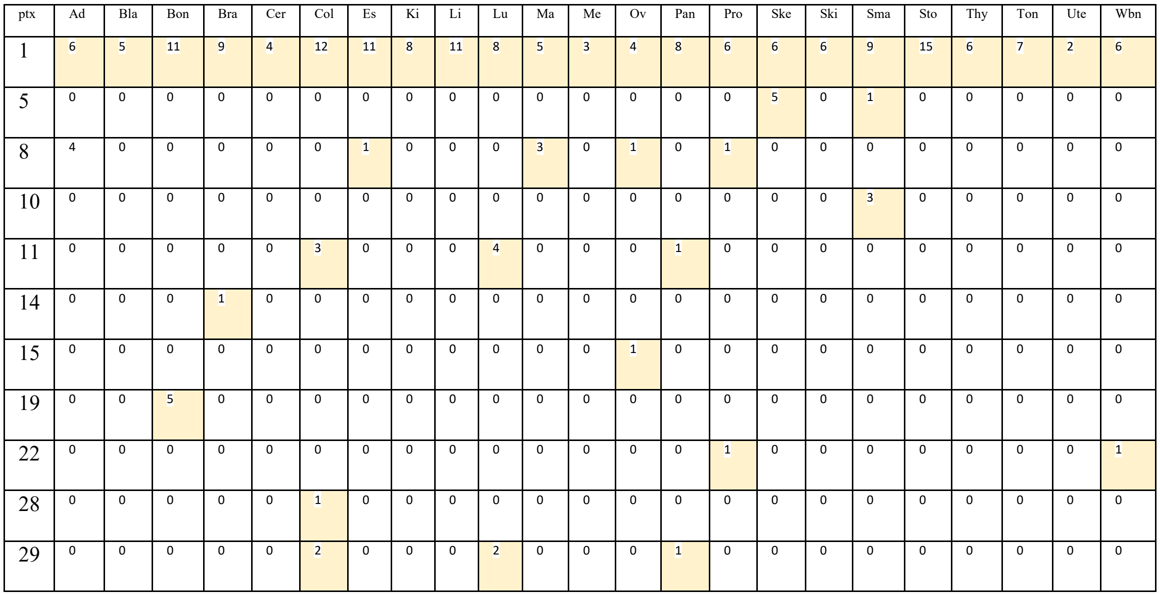


Figure 17: Shared TCGA-LUSC ORFs in Cancer vs. Healthy Samples

(Techachakrit, 2022)

Amongst the 31 ORFs which were found to be shared amongst patients in the TCGA-LUSC sampling set, 11 are considered TAA candidates. The TAA candidates (ptx1, ptx5, ptx8, ptx10, ptx11, ptx14, ptx15, ptx19, ptx22, ptx28, and ptx29) are ORFs that are present in the healthy tissues dictionary (with TPM ≠ 0), and have passed the expression level parsing in the lung cancer dictionary (with TPM > 1). The TAA candidates are present in various tissues (Figure 18).



ptx	Ad	Bla	Bon	Bra	Cer	Col	Es	Ki	Li	Lu	Ma	Me	Ov	Pan	Pro	Ske	Ski	Sma	Sto	Thy	Ton	Ute	Wbn
1	6	5	11	9	4	12	11	8	11	8	5	3	4	8	6	6	6	9	15	6	7	2	6
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	1	0	0	0	0	0
8	4	0	0	0	0	0	1	0	0	0	3	0	1	0	1	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0
11	0	0	0	0	0	3	0	0	0	4	0	0	0	1	0	0	0	0	0	0	0	0	0
14	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
19	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
28	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	2	0	0	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0

Figure 18: Count of patients with TAA candidates expression in various healthy tissues

Tissues listed from left to right, respectively: adrenal gland, bladder, bone marrow, brain, cervix, colon, esophagus, kidney, liver, lung, mammary gland, medullary thymic epithelial cells (mTECs), ovary, pancreas, prostate, skeletal muscle, skin, small intestine, thyroid gland, tonsil, uterus, whole blood nuclear cells (WBNC). (Techachakrit, 2022)

Interestingly, there were two sets of ORFs originating from canonical transcriptional events (ptx1 and ptx2) that are present in all TCGA-LUSC sampling sets. The first, ptx1 is a TAA candidate with presence in all healthy and cancer tissues. Since ptx1 is presented in all healthy tissues at relatively high TPMs, ranging from 5.65 to 124.28 (Figure 19), its use as cancer biomarker and therapeutics may be null. The ORFs representing ptx1 were identified as an isoform of the nuclear distribution protein nudE-like 1 (NDEL1) via UniProt search. The NDEL1 gene in the general coding regions encodes for a coiled-coil protein that plays key roles in cytoskeletal organization, cellular signaling, and neuron migration. It is required for mitosis in some cell types but appears to be dispensable for mitosis in cortical neuronal progenitors. Multiple isoforms of the genes have been previously observed, with a ubiquitous expression in all 22 listed tissues.

On the other hand, the ptx2 is highly specific to the TCGA-LUSC sampling set. It is a TSA candidate, as it lacks any expression (TPM = 0) in the healthy tissues dictionary. All 100-lung cancer patients express ptx2 at varying degrees of TPM, ranging from 23.2476 to 187.7537 (Figure 19).

The transcript associated with ptx2, ENST00000000412.6, is a novel alternative isoform associated with the cation-dependent, mannose-6-phosphate receptor (CD-MPR) gene. CD-MPR plays a key role in lysosomal function through the specific transport of mannose-6-phosphate-containing acid hydrolases from the Golgi complex to lysosomes. Multiple isoforms of the genes have been previously observed. Additionally, the expression of CD-MPR protein has been noted to be influenced by estradiol in MCF-7 breast cancer cells (Bannoud et al., 2018).

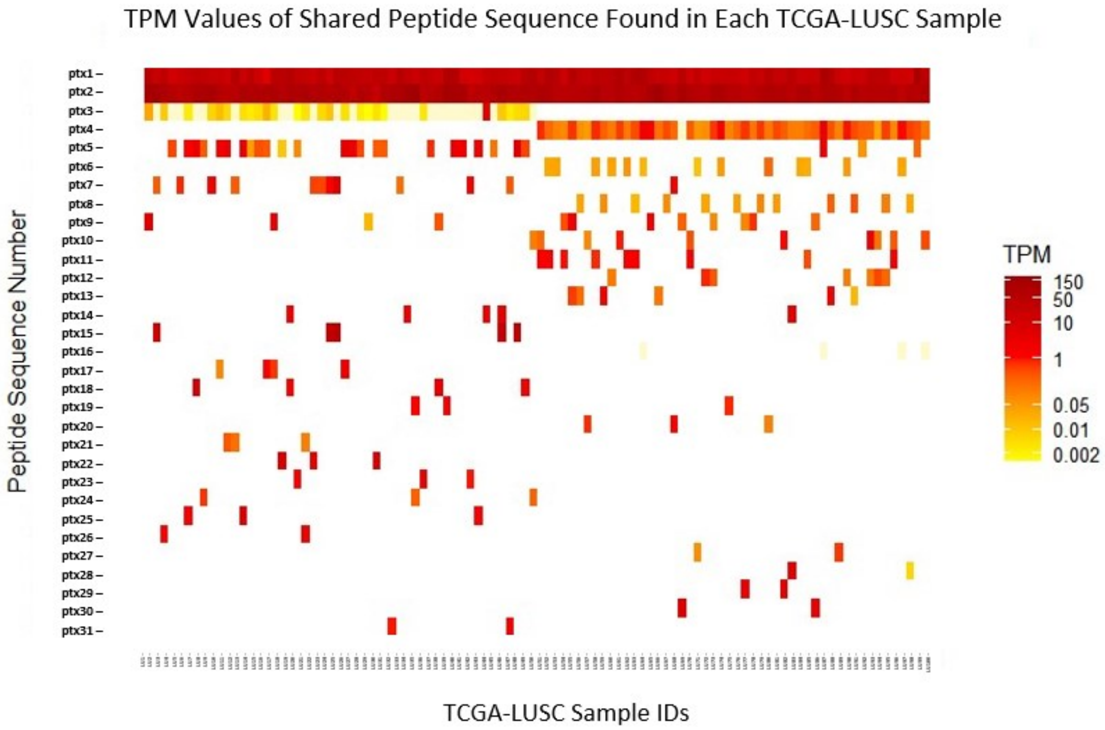


Figure 19: TPM values of shared peptide sequence amongst TCGA-LUSC samples

The distribution of TPM values amongst the peptide ID (ptx1-ptx31) in the y-axis as correlated to the house-assigned TCGA-LUSC patient sample ID (LU1-LU100) in the x-axis. (Techachakrit, 2022)

Furthermore, we have identified an even split in expressions between ptx3 and ptx4 in amongst the 100 patients. Upon investigation, we have found that the first group of patients (50 people), belonging to those diagnosed with lung squamous cell carcinomas with 'classical' subtype, present the expression of ptx3 peptides and not ptx4 peptides (Figure 19). On the other hand, the other half of our sample size (50 people), diagnosed with squamous cell carcinomas with 'basal' subtype presents the expression of ptx4 peptides and not ptx3 peptides.

According to the original study by the Cancer Genome Atlas Research Network, the classical subtype is characterized by chromosome instability, hypermethylation, and alterations in KEAP1, NFE2L2, and PTEN genes, while the basal subtype expresses alterations in the NF1 gene (Figure 20).

We believe this to be a sampling bias on our end. Due to limited computational space, RNA data (in BAM file format) from 50 samples were first downloaded from the TCGA-LUSC project and subsampled for homogeneity. Due to insufficiency in the sample size, the data director had bulk downloaded another 50 samples and subsampled again, resulting in highly segregated results between ptx3 and ptx4.

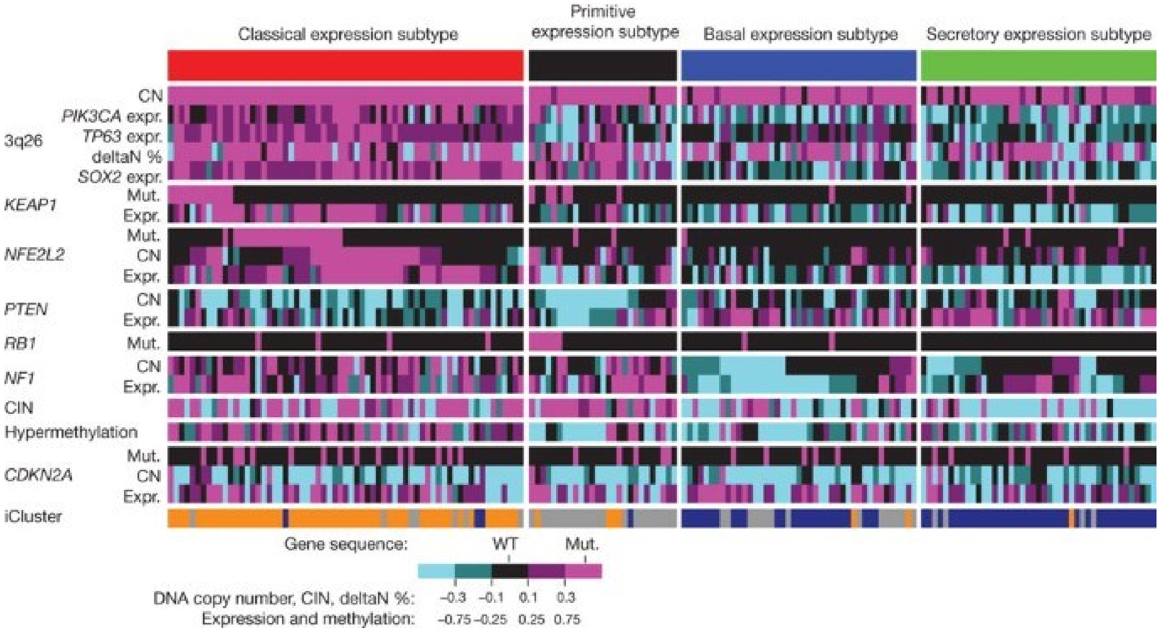


Figure 20: Gene expression subtypes integrated with genomic alterations

(The Cancer Genome Atlas Research Network, 2012)

Section IV: Cross-tissues distribution analysis of ptx2

We also performed a distribution analysis for the TPM values of the 31 ORFs that were shared (Figure 21). The overall distributions are trimodal, with the second mode roughly normal. With a special interest in the TSA candidate, ptx2, we overlayed the TPM distribution against the overall distribution, resulting in an approximately normal curve within the fourth quartile of the overall TPM values.

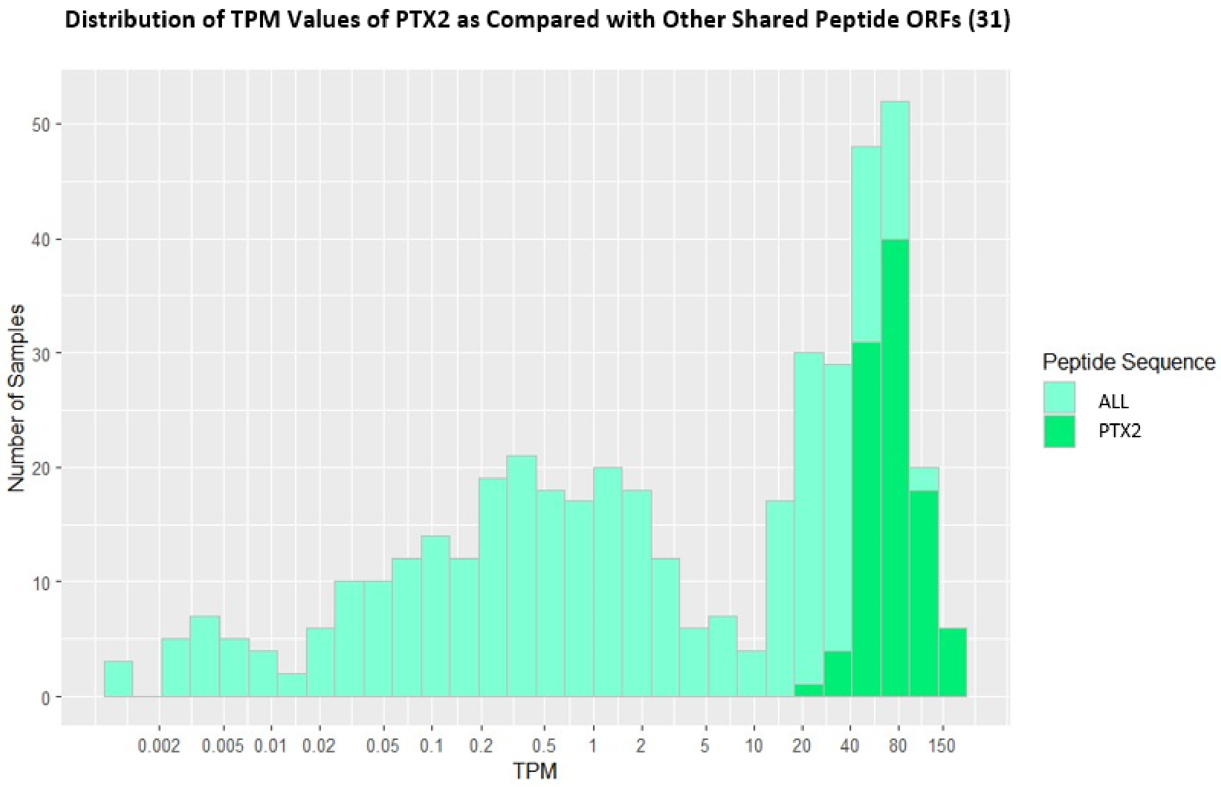

Distribution of TPM Values of PTX2 as Compared with Other Shared Peptide ORFs (31)

Figure 21: TPM values of ptx2 as compared to the distribution of other shared peptides

The distribution histogram comparing the results of ptx2, a peptide presented specifically to all TCGA-LUSC patient samples. (Techachakrit, 2022)

Section V: Identification of relative environmental factors on ptx2

We further studied the potential correlations between this highly specific, high TPM, shared ptx2 peptide with a known factor of exposure, smoking. We have analyzed three separate situations in which smoking was involved: looking at the TPM value distributions of (1) ptx2 and the length of years the patient has been smoking (Figure 22.a), (2) ptx2 and the number of cigarettes smoked per day (Figure 22.b), and (3) ptx2 and packs of cigarette smoked per year (Figure 22.c).

However, we were unable to find a statistically significant correlation in all three situations. This is due to the lack of homogeneity in the obtained data, where in the given samples of 100 patients, roughly 85% are clear smokers. However, the other 15% are of unknown origins, with which has partial to no information regarding the state of exposure to cigarettes. 30% of the sample also lacks the length of time that patients have been smoking prior to cancer diagnosis, even though they were denoted clinically as smokers.

In Figure 22.b and 22.c, despite not showing a correlation with high TPM density, the randomness may suggest that smoking does not act as a factor that is relevant to the existence of ptx2.

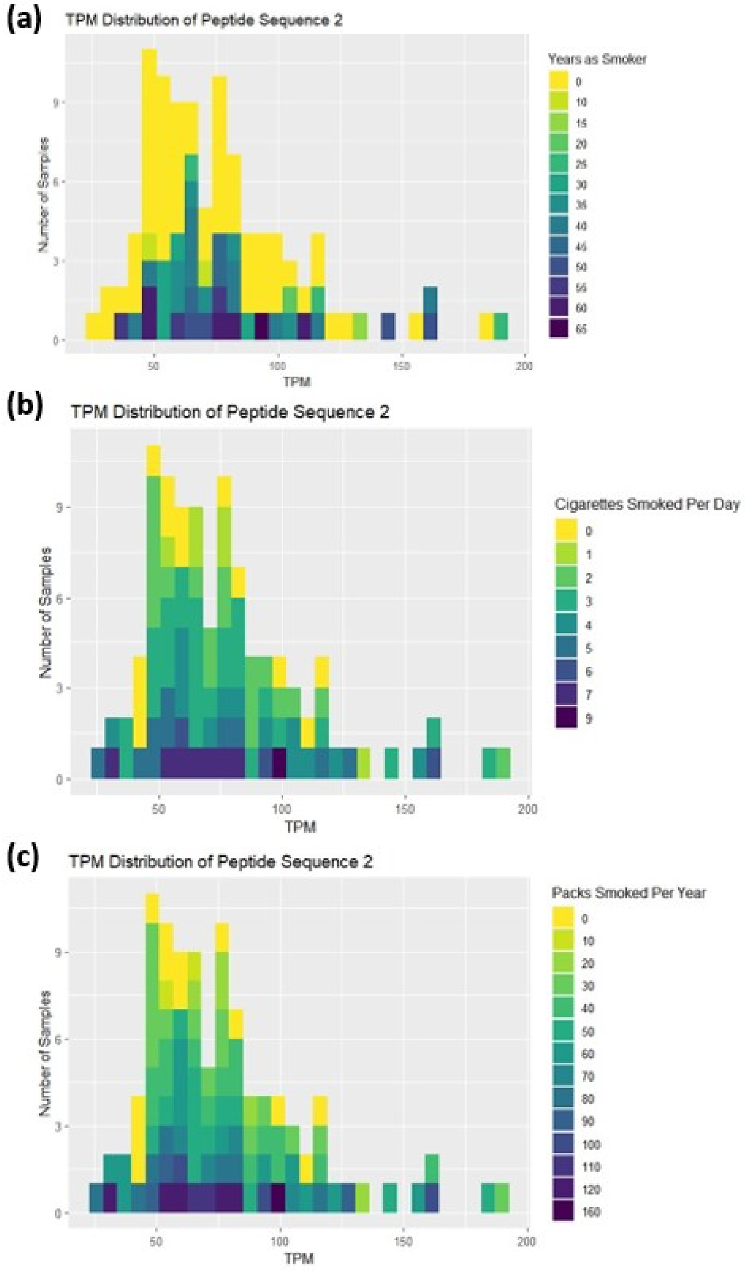


Figure 22: TPM distributions of ptx2 amongst various smoking scenarios

The TPM distributions of ptx2 amongst (a) years of smoking prior to cancer diagnosis, (2) number of cigarettes smoked per day, and (c) number of packs of cigarette smoked per year. (Techachakrit, 2022)

Section VI: *in silico* peptide cleavage from TSA- and TAA-candidates

The shared TSAs and TAAs candidates identified were analyzed for their potential as neoantigenic peptides. All 31 ORFs were processed for proteasome cleavage prediction via NetChop version 3.1. using the default C term 3.0 settings and a threshold of 0.5. The output peptides were sorted and counted length-wise.

In order to identify peptides that are usable as neoantigens, they must fit into the core binding motifs of the MHC molecules. In humans, the core binding motifs of both MHC class I and II are comprised of peptides length of approximately 9 amino acids (Garcia-Garijo et al., 2019). However, amino acids of lengths between 8 to 11 are considered for neoantigen developments.

Out of the 31 ORFs, 16 were identified to have viable lengths post-cleavage of between 8-11 amino acids. Amongst these 16 ORFs, 12 are TSA candidates (ptx2, ptx4, ptx9, ptx12, ptx16, ptx18, ptx21, ptx23, ptx26, ptx227, ptx30, and ptx31), while the other 4 are TAA candidates (ptx8, ptx10, ptx22, and ptx28). For the purpose of this Thesis, we were able to identify both shared TSAs and TAAs candidates from the novel hybrid *de novo* pipeline.

Chapter IV.

Discussion

In order to explore the global landscape of TSAs, we have developed the novel Hybrid *de novo* pipeline, utilizing proteogenomic approaches to incorporate both canonical RNA sequence processing and a *de novo* one to increase the range of discoveries of multiple novel isoforms. Even as the focal point of cancer vaccine immunotherapies was within the scope of non-synonymous mutations, we have expanded TSA discoveries to the non-canonical transcriptional frames. The study shed light on where previously unmet needs in oncological research and medicine reside with its unanticipatedly large TSA landscape.

In the 20 TSAs and 31 TAAs shared within the TCGA-LUSC sample group, only 2 (ptx1 and ptx2) were discovered based on atypical canonical transcriptional frames, while the rest resulted from non-canonical origins. The lack of fully annotated peptide amongst the list of TSAs and TAAs suggest an alarmingly high number of antigens that were missed by traditional RNA sequence processing approaches, which focused on exonic mutations. In addition to capturing of canonical transcriptional frames from the RNA sequencing data, our approach efficiently captures ORFs generated from other sources of non-mutational, non-canonical translational frames such as complex structural variants.

Since most TSAs and TAAs discovered via the novel Hybrid *de novo* pipeline which derives from non-coding regions do not overlap their mutational counterparts, they present major advantages over traditional neoantigens arising from mutational-based tumor specificity. Whereas traditional mutational-based TSAs are a private, yet changing repertoire for a singular patient, the shared TSAs and TAAs promise greater potential for therapeutics and diagnosis biomarkers, as they are shared amongst multiple patients. As biomarkers, shared TSAs and TAAs can yield early diagnosis of cancers, leading to better prognosis overall. The process of developing the TSAs vaccine should not be limited to neoantigens arising from mutations, but also include the ones shared amongst tumors. Provided that the search for neoantigens, in the future, encompasses these shared TSAs, the potential for a more 'generic' targets for T-cell-based cancer immunotherapy may be possible. If so, it will decrease the production time, cost and delivery to further improve the patients' quality of life.

Limitations

Despite potential access to international databases, such as the TCGA, NCBI-ANTE, and ONCOBOX, we acknowledge that the availability of normal human thymus and normal human tissues is limited; thus, we rely on computational optimization for creating a well-rounded global database that would potentially capture most of the self-expressed peptides. However, it could not be guaranteed that the global database would fully capture the true extent of all normal human self-expressed peptides and thus required reupdating of the database for greater computational precision. Further, since neoantigenic non-canonical transcripts are still an underexplored field of study, there were issues of validating novel transcripts, prioritizing reads from increasingly large databases, identifying likely reading frames and maintaining a compact mini-database for downstream MS searches.

In the identification of sufficient sample size, the concerns of sensitivity and false discovery rate come into play. Even though the number of samples required for complete sample coverages can be calculated statistically, we cannot guarantee the number of the tissue samples available. If the number of available samples is lower than those required by the rarified results, we flagged the specific tissue within the database as incomplete. The results of an incomplete tissue-specific database cause a lower accuracy (higher false discovery rate) when computing for neoantigens within tumor samples. However, it would not decrease the rate of discovery within the computational pipeline.

For Quantitative Analysis of Individual Tumor Samples, the peptides identified through this pipeline are not representative of all possible neoantigen peptides, as it does not account for specific mutation-based peptides (which are the current common pipeline). However, for usages in neoantigen identification, the current neoantigen variant calling pipeline for single-nucleotide variations (SNVs) and insertion-deletion (INDELS) mutations can be combined with this pipeline for an improved rate of neoantigen discovery.

Finally, in this computational study, we hypothesized the existence of a shared, possibly tumor-driven antigens when searching in a larger pool generated from the computational pipeline, taking into account the non-canonical origins. However, these transcriptome-derived peptides may not be translated *in vivo*. Our work presents the possibility of the shared TSAs and TAAs. Further, the identification of shared peptides could not guarantee immunogenicity *in vivo*. Regardless, the resulting meta-analysis should reveal the intrinsic biology of different cancer from a broader perspective.

Appendix 1

Unique_counter.py

```
import pandas as pd
import os
import random
import glob
import numpy as np
import argparse

def parse_files(fname):
    '''
    Function: Parse GTF files with 6 or 8 attributes
    Returns: Filtered list with two columns 'transcript_id', 'FPKM_val'
    V.3
    '''
    #import files, remove 2 rows
    df = pd.read_csv(fname, delimiter = '\t', header = None, skiprows = 2)

    #rename columns
    df.columns = ['seqname', 'source',
'feature','start','end','score','strand','frame','attributes']

    #split attributes
    if fname.endswith("_eb.gtf"):
        cols = ['gene_id', 'transcript_id_raw',
'ref_gene_name','cov','FPKM_raw','TPM','blank']
        df[cols] = df.attributes.str.split(';', expand = True)
        df[['toss1', 'transcript_id_mark', 'transcript_id']] =
df.transcript_id_raw.str.split(" ", expand=True)
        df[['toss2', 'FPKM_mark', 'FPKM_val']] = df.FPKM_raw.str.split(" ",
expand=True)
    else:
        cols = ['gene_id', 'transcript_id_raw', 'reference_id',
'ref_gene_id','ref_gene_name','cov','FPKM_raw','TPM','blank']
        df[cols] = df.attributes.str.split(';', expand = True)
        df[['toss1', 'transcript_id_mark', 'transcript_id']] =
df.transcript_id_raw.str.split(" ", expand=True)
        df[['toss2', 'FPKM_mark', 'FPKM_val']] = df.FPKM_raw.str.split(" ",
expand=True)

    #get rid of quotations around values
    df['transcript_id'] = df.transcript_id.str.replace('"', '')
    df['FPKM_val'] = df.FPKM_val.str.replace('"', '')

    #Boolean filter for only FPKM (ignore cov)
    df = df[df.FPKM_mark=='FPKM']
    #convert string value to float
    df.FPKM_val = df.FPKM_val.astype(float)

    #final dataframe
    df_filtered = df[['transcript_id', 'FPKM_val']]

    return df_filtered

def key2files(files_lst):
    """
    Function: Convert input files to Dict
    Returns: Dict with common Key name to multiple values
    """
    samples_dict = {}
    for f in files_lst:
        s = f.split('_')[0]
        if s not in samples_dict:
            samples_dict[s] = []
        samples_dict[s].append(f)
    return samples_dict

def merge_df(lst_fname):
    """
    Function: Merge & parsed input files
    Input: List of files
    Returns: Merged file with two columns "transcript_id" "FPKM_val"
    """
    merged_df = None
    for fname in lst_fname:
        df1 = parse_files(fname)
        if merged_df is None:
            merged_df = df1
        else:
            merged_df = pd.concat([merged_df, df1])
    return merged_df

def calc_unique_tx(idx, samples_dict) -> int:
    """
    Function: Calculate Unique transcripts based on merged df
    Returns: unique counts of the file
    """
    unique_tx = [] # running list of unique transcripts
    for i in idx:
        df_filtered = merge_df(samples_dict[i])
        df_filtered = df_filtered[df_filtered.FPKM_val >= 1]
        unique_tx_tmp = df_filtered.transcript_id.unique().tolist() # unique tx
list for given file
        unique_tx = unique_tx + unique_tx_tmp

    return len(np.unique(unique_tx))

def retrieve_unique_counts(n_increment, samples_dict):
    """
    Function: Retrieve Unique Counts
    Input: n_increment is the number of increasing increments
    """

    n_samples = 1 #number of samples for sampling
    n_samples_total = len(samples_dict) #number of total samples files

    dfs = []

    while True:
        for i in range(5):
            n_idx = [random.randint(0, n_samples_total-1) for n in
range(n_samples)]
            idx_key = [list(samples_dict.keys())[i] for i in n_idx]
            n_unique_tx = calc_unique_tx(idx_key, samples_dict)
            df = pd.DataFrame([{'n_sample': n_samples, 'n_unique_tx':
n_unique_tx}])
            dfs.append(df)
        n_samples += 1 #increments
        if n_samples > n_samples_total:
            break
    df_merged = pd.concat(dfs, ignore_index=True)

    return df_merged

def main(args):
    #list of input files within the folder
    print("Starting unique counter...")
    files_lst = [f for f in glob.glob("*.gtf")]
    samples_dict = key2files(files_lst)
    random.seed(10)
    df = retrieve_unique_counts(args.n_increment, samples_dict)
    print("Writing output...")
    df.to_csv('output.csv')


if __name__ == '__main__':
    parser = argparse.ArgumentParser(description="Unique counter")
    parser.add_argument('n_increment', type=int)
    args = parser.parse_args()

    main(args)
```

Appendix 2

ConvertGTF_nt_sample_wise.py

```
import glob
import os
import sys
import pybedtools
from pybedtools import BedTool
from os import path
from gtfparse import read_gtf

def GTF2BED(files_lst):
    '''
    Function: Parse GTF files with 6 or 8 attributes
    Returns: BED files (CSV) with 'chromosome', 'start', 'end', 'exon_id',
'score', 'strand'
    V.1
    '''
    #import files via gtfparse
    for file in files_lst:
        df = read_gtf(file)
        df = df.rename(columns = {'seqname':"chromosome"})
        df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
        df['length'] = df['end'] - df['start']

        df = df[df['feature'] != 'transcript']
        df = df[df['length'] != 0]

        #return df_filtered
        df_filtered = df[['chromosome', 'start', 'end', 'exon_id', 'score',
'strand']]


        df_filtered.to_csv(file[:-4] + ".bed", sep = '\t', header = False,
index = False)

def BED_toNT(bed_lst):
    '''
    Function: take in BED file from list and extract nt sequences based on
GRCh38.102 fasta file, then stitch together multiple exons to form transcript &
isoforms
    Returns: fasta file in multiple-exons as novel isoforms
    '''
    in_fasta = BedTool('/shareqnap/TCGA_JT/GRCh38.d1.vd1/GRCh38.d1.vd1.fa')

    for b in bed_lst:
        in_bed = BedTool(b)
        bedExt = in_bed.sequence(fi = in_fasta, nameOnly = True)
        #print(open(bedExt.seqfn).read())

        file = open(b[:-4] + ".fa", "w")
        s = open(bedExt.seqfn).read()

        key2lines = {}
        key = ''

        for line in s.split('\n'):
            line = line.strip()
            if '>' in line:
                key = line.split('_')[0]
                continue
            key2lines[key] = key2lines.get(key, '') + line
        for e in sorted(key2lines):
            #print(e)
            #print(key2lines[e])
            file.write(e + '\n' + key2lines[e] + '\n')

        file.close()

pybedtools.helpers.set_bedtools_path(path='/data/users/techajir/.py3/lib64/pyth
on3.6/site-packages/pybedtools/')

files_lst = [f for f in glob.glob("*.gtf")]
bed_lst = [b for b in glob.glob(".bed")]

GTF2BED(files_lst)
BED_toNT(bed_lst)
```

Appendix 3

GTF2FilteredList-eb.py

```
import pandas as pd
import glob
from gtfparse import read_gtf

def parseGTF2filterpos(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == '+']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListpos(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_eb.gtf"):
            eb_merged = parseGTF2filterpos(fname)
            eb_merged = eb_merged.append(parseGTF2filterpos(fname))

            eb_BED = eb_merged[['transcript_id']]

    return eb_BED.to_csv("eb-pos_list.txt", header = False, index = False)

def parseGTF2filterneg(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with 
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == '-']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListneg(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on 
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_eb.gtf"):
            eb_merged = parseGTF2filterneg(fname)
            eb_merged = eb_merged.append(parseGTF2filterneg(fname))

            eb_BED = eb_merged[['transcript_id']]

    return eb_BED.to_csv("eb-neg_list.txt", header = False, index = False)

def parseGTF2filterunk(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with 
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == 'nan']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListunk(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_eb.gtf"):
            eb_merged = parseGTF2filterunk(fname)
            eb_merged = eb_merged.append(parseGTF2filterunk(fname))

            eb_BED = eb_merged[['transcript_id']]

    return eb_BED.to_csv("eb-unk_list.txt", header = False, index = False)

files_lst = [f for f in glob.glob("*.gtf")]

merge_toListneg(files_lst)
merge_toListpos(files_lst)
merge_toListunk(files_lst)
```

Appendix 4

GTF2FilteredList-woe.py

```
import pandas as pd
import glob
from gtfparse import read_gtf

def parseGTF2filterpos(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == '+']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListpos(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_woe.gtf"):
            woe_merged = parseGTF2filterpos(fname)
            woe_merged = woe_merged.append(parseGTF2filterpos(fname))

            woe_BED = woe_merged[['transcript_id']]

    return woe_BED.to_csv("woe-pos_list.txt", header = False, index = False)

def parseGTF2filterneg(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == '-']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListneg(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_woe.gtf"):
            woe_merged = parseGTF2filterneg(fname)
            woe_merged = woe_merged.append(parseGTF2filterneg(fname))

            woe_BED = woe_merged[['transcript_id']]

    return woe_BED.to_csv("woe-neg_list.txt", header = False, index = False)

def parseGTF2filterunk(gtf_name):
    '''
    Function: Parse GTF files with 6 or 8 attributes, filtering for one with
TPM > 0 transcript
    Returns: df with exon_id, filtered for exons only
    V.1
    '''
    #import files via gtfparse
    df = read_gtf(gtf_name)
    df = df.rename(columns = {'seqname':"chromosome"})
    df['exon_id'] = df['transcript_id'] + "_" + df['exon_number']
    df['length'] = df['end'] - df['start']
    df['TPM'] = df['TPM'].notna()
    df['TPM'] = df['TPM'].astype(float)

    df = df[df['strand'] == 'nan']
    df = df[df['length'] != 0]
    df = df[df['TPM'] > 0]
    df = df[df['feature'] == 'transcript']

    #return df_filtered
    df_filtered = df[['transcript_id']]

    return df_filtered

def merge_toListunk(files_lst):
    '''
    Function: take in list of gtf files, parsing over them, and merged based on
type, saving as csv (bed file)
    Returns: csv files containing merged eb and woe results
    V.1
    '''
    for fname in files_lst:
        if fname.endswith("_woe.gtf"):
            woe_merged = parseGTF2filterunk(fname)
            woe_merged = woe_merged.append(parseGTF2filterunk(fname))

            woe_BED = woe_merged[['transcript_id']]

    return woe_BED.to_csv("woe-unk_list.txt", header = False, index = False)

files_lst = [f for f in glob.glob("*.gtf")]

merge_toListneg(files_lst)
merge_toListpos(files_lst)
merge_toListunk(files_lst)
```

Appendix 5

seqkit.py

```
import os

dir_ = '.'
out_ = 'out_'
listtxt = os.path.join(dir_, 'neg_list.txt')
for f in os.listdir(dir_):
        if f.endswith('.fa') and not f.startswith(out_):
                infile = os.path.join(dir_, f)
                outfile = os.path.join(dir_, out_ + f)
                cmd = 'seqkit grep -n -f "%s" "%s" > "%s"' % (listtxt, infile,
outfile)
                os.system(cmd)

dir_ = '.'
out_ = 'out_'
listtxt = os.path.join(dir_, 'pos_list.txt')
for f in os.listdir(dir_):
        if f.endswith('.fa') and not f.startswith(out_):
                infile = os.path.join(dir_, f)
                outfile = os.path.join(dir_, out_ + f)
                cmd = 'seqkit grep -n -f "%s" "%s" > "%s"' % (listtxt, infile,
outfile)
                os.system(cmd)

dir_ = '.'
out_ = 'out_'
listtxt = os.path.join(dir_, 'unk_list.txt')
for f in os.listdir(dir_):
        if f.endswith('.fa') and not f.startswith(out_):
                infile = os.path.join(dir_, f)
                outfile = os.path.join(dir_, out_ + f)
                cmd = 'seqkit grep -n -f "%s" "%s" > "%s"' % (listtxt, infile,
outfile)
                os.system(cmd)
```

Appendix 6

translate_allORF.py

```
import glob
from Bio import SeqIO
from Bio import Seq
from collections import defaultdict
from Bio.SeqRecord import SeqRecord

def translate_pos(fa_lst):
    '''
    Function: translate all nucleotide fasta files that are positively stranded
three frames forward, keep all ORFs with length >= 8
    Returns: peptide fasta files
    '''
    for f in fa_lst:
        with open(f[:-3] + 'ALLorf.fa', 'w') as fout:
            for rec in SeqIO.parse(f, 'fasta'):
                for strand, seq in [(1, rec.seq)]:
                    for frame in range(3):
                        length = 3 * ((len(rec)-frame) // 3)
                        for pro in
seq[frame:frame+length].translate().split("*"):
                            splitlocal = pro.find("M")
                            seq_final = pro[splitlocal:]
                            if len(seq_final) >= 8:
                                print("%s...%s - length %i, strand %i, frame
%i" \
                                % (seq_final[:10], pro[-3:], len(seq_final),
strand, frame))
                                SeqIO.write(SeqRecord(seq = seq_final, id =
rec.id, description = str(frame)), fout, 'fasta')

def translate_neg(fa_lst):
    '''
    Function: translate all nucleotide fasta files that are negatively stranded
three frames reverse, keep all ORFs with length >= 8
    Returns: peptide fasta files
    '''
    for f in fa_lst:
        with open(f[:-3] + 'ALLorf.fa', 'w') as fout:
            for rec in SeqIO.parse(f, 'fasta'):
                for strand, seq in [(-1, rec.seq.reverse_complement())]:
                    for frame in range(3):
                        length = 3 * ((len(rec)-frame) // 3)
                        for pro in
seq[frame:frame+length].translate().split("*"):
                            splitlocal = pro.find("M")
                            seq_final = pro[splitlocal:]
                            if len(seq_final) >= 8:
                                print("%s...%s - length %i, strand %i, frame
%i" \
                                % (seq_final[:10], pro[-3:], len(seq_final),
strand, frame))
                                SeqIO.write(SeqRecord(seq = seq_final, id =
rec.id, description = str(frame)), fout, 'fasta')

def translate_unk(fa_lst):
    '''
    Function: translate all nucleotide fasta files that are unknown stranded in
all six frames, keep all ORFs with length >= 8
    Returns: peptide fasta files
    '''
    for f in fa_lst:
        with open(f[:-3] + 'ALLorf.fa', 'w') as fout:
            for rec in SeqIO.parse(f, 'fasta'):
                for strand, seq in (1, rec.seq), (-1,
rec.seq.reverse_complement()):
                    for frame in range(3):
                        length = 3 * ((len(rec)-frame) // 3)
                        for pro in
seq[frame:frame+length].translate().split("*"):
                            splitlocal = pro.find("M")
                            seq_final = pro[splitlocal:]
                            if len(seq_final) >= 8:
                                print("%s...%s - length %i, strand %i, frame
%i" \
                                % (seq_final[:10], pro[-3:], len(seq_final),
strand, frame))
                                SeqIO.write(SeqRecord(seq = seq_final, id =
rec.id, description = str(frame)), fout, 'fasta')


posfa_lst = [p for p in glob.glob("pos_*")]
translate_pos(posfa_lst)

negfa_lst = [n for n in glob.glob("neg_*")]
translate_neg(negfa_lst)

unkfa_lst = [u for u in glob.glob("unk_*")]
translate_unk(unkfa_lst)
```

Appendix 7

GTF2TPM.py

```
import pandas as pd
import glob
from gtfparse import read_gtf

def GTF2GeneExp(gtf_list):
    '''
    Function: Parse GTF files with 6 or 8 attributes
    Returns: df with transcript_id:TPM, dropping multiple exons (keep
transcripts)
    '''
    for gtf in gtf_list:
        df = read_gtf(gtf)
        df_ext = df[['transcript_id', 'TPM']]

        #remove all non-values
        nan_value = float("NaN")
        df_ext.replace("", nan_value, inplace=True)
        df_ext.dropna(subset = ["TPM"], inplace=True)

    #return df_ext:
    df_ext.to_csv(gtf[:-4] + ".csv", sep = '\t', header = False, index = False)

gtf_list = [f for f in glob.glob("*.gtf")]
GTF2GeneExp(gtf_list)
```

Appendix 8

TPM_dict-LUSC.ipynb

```
peptide_seq = []
tissue_1 = []
transcript_ID = []
patient_ID = []
method = []

tx2tissuesample = {}
for tissue in master_dict:
    entry = master_dict[tissue]
    for seq in entry:
        peptide = entry[seq]
        if peptide not in tx2tissuesample:
            tx2tissuesample[peptide] = set()
        tx2tissuesample[peptide].add((tissue, seq))
print('------------output------------')
print(len(tx2tissuesample))
for transcript in tx2tissuesample:
    t_samples = sorted(tx2tissuesample[transcript])
    print(transcript)
    print(type(transcript))
    print(len(t_samples))
    for e in t_samples:
        peptide_seq.append(transcript)
        print('\t', e)
        string_e = ''.join(e)
        print(string_e)
        print(string_e.split(sep = '>')[0])
        tissue_var = string_e.split(sep = '>')[0]
        tissue_1.append(tissue_var)
        print(string_e.split(sep = '>')[1].split(sep = ' ')[0])
        transcript_ID.append(string_e.split(sep = '>')[1].split(sep = ' ')[0])
        print(string_e.split(sep = '-')[1].split(sep = '_')[1])
        patient_ID.append(string_e.split(sep = '-')[1].split(sep = '_')[1])
        print(string_e.split(sep = '-')[1].split(sep = '_')[2][:-9])
        method.append(string_e.split(sep = '-')[1].split(sep = '_')[2][:-9])
        print('-----------------------')
    print('>>>>>>>>>>>>>>>>>>>>>>>>>>>')
```

Appendix 9

TPM_dict-ONCOBOX.ipynb

```
peptide_seq = []
tissue_1 = []
transcript_ID = []
patient_ID = []
method = []

tx2tissuesample = {}
for tissue in master_dict:
    entry = master_dict[tissue]
    for seq in entry:
        peptide = entry[seq]
        if peptide not in tx2tissuesample:
            tx2tissuesample[peptide] = set()
        tx2tissuesample[peptide].add((tissue, seq))
print('------------output------------')
print(len(tx2tissuesample))
for transcript in tx2tissuesample:
    t_samples = sorted(tx2tissuesample[transcript])
    print(transcript)
    print(type(transcript))
    print(len(t_samples))
    for e in t_samples:
        peptide_seq.append(transcript)
        print('\t', e)
        string_e = ''.join(e)
        print(string_e)
        print(string_e.split(sep = '>')[0])
        tissue_var = string_e.split(sep = '>')[0]
        tissue_1.append(tissue_var)
        print(string_e.split(sep = '>')[1].split(sep = ' ')[0])
        transcript_ID.append(string_e.split(sep = '>')[1].split(sep = ' ')[0])
        print(string_e.split(sep = '-')[1].split(sep = '_')[1])
        patient_ID.append(string_e.split(sep = '-')[1].split(sep = '_')[1])
        print(string_e.split(sep = '-')[1].split(sep = '_')[2][:-9])
        method.append(string_e.split(sep = '-')[1].split(sep = '_')[2][:-9])
        print('-----------------------')
    print('>>>>>>>>>>>>>>>>>>>>>>>>>>>')
storage_df = pd.DataFrame({'peptide_seq':peptide_seq, 'tissue':tissue_1,
'transcript_ID':transcript_ID, 'patient_ID':patient_ID, 'method':method})
storage_df.to_csv("common_healthy_tissue_peptides.csv")
```

References

Alborelli, I. et al. (2020). Tumor mutational burden assessed by targeted NGS predicts clinical benefit from immune checkpoint inhibitors in non-small cell lung cancer. *Journal of Pathology*, *250*(1), 19–29. https://doi.org/10.1002/path.5344.

Anagnostou, V. et al. (2017). Evolution of neoantigen landscape during immune checkpoint blockade in non – small cell lung cancer. *Cancer discovery*, *7*(3), 264–277. https://doi.org/10.1158/2159-8290.CD-16-0828

Andreev, D.E. et al. (2015). Translation of 5' leaders is pervasive in genes resistant to eIF2 repression. *ELife, 2015*(4), 1-21. https://doi.org/10.7554/eLife.03971

Bannoud, N. et al. (2018). Cation-dependent mannose-6-phosphate receptor expression and distribution are influenced by estradiol in MCF-7 breast cancer cells. *PloS one, 13*(8), e0201844. https://doi.org/10.1371/journal.pone.0201844

Barvík, I. et al. (2017). Non-canonical transcription initiation: The expanding universe of transcription initiating substrates. *FEMS Microbiology Reviews*, *41*(2), 131–138. https://doi.org/10.1093/femsre/fuw041

Bassani-Sternberg, M. et al. (2016). Direct identification of clinically relevant neoepitopes presented on native human melanoma tissue by mass spectrometry. *Nature Communications*, *7*. https://doi.org/10.1038/ncomms13404.

Bjerregaard, A. M. et al. (2017). An analysis of natural T cell responses to predicted tumor neoepitopes. *Frontiers in Immunology*, *8*, 1–9. https://doi.org/10.3389/fimmu.2017.01566.

Bodey, B. et al. (2000). Failure of cancer vaccines: the significant limitations of this approach to immunotherapy. *Anticancer Research*, *20*(4), 2665–76.

Chong, C. et al. (2020). Integrated proteogenomic deep sequencing and analytics accurately identify non-canonical peptides in tumor immunopeptidomes. *Nature Communications*, *11*(1). https://doi.org/10.1038/s41467-020-14968-9

Christofi, T., & Zaravinos, A. (2019). RNA editing in the forefront of epitranscriptomics and human health. *Journal of Translational Medicin*e, *17*(1), 319. https://doi.org/10.1186/s12967-019-2071-4

Collins, J. M., Redman, J. M., & Gulley J. L., (2018). Combining vaccines and immune checkpoint inhibitors to prime, expand, and facilitate effective tumor immunotherapy. *Expert review of vaccines, 17*(8), 697-705. https://doi.org/10.1080/14760584.2018.1506332

Day, S., Ramsland, P., & Apostolopoulos, V. (2009). Non-Canonical Peptides Bound to MHC. *Current Pharmaceutical Design*, *15*(28), 3274–3282. https://doi.org/ 10.2174/138161209789105090.

Dersh, D., Hollý, J., & Yewdell, J. W. (2021). A few good peptides: MHC class I-based cancer immunosurveillance and immunoevasion. *Nature Reviews Immunology*, *21*(2), 116–128. https://doi.org/10.1038/s41577-020-0390-6.

Dobin, A. et al. (2013). STAR: Ultrafast universal RNA-seq aligner. *Bioinformatics, 29*(1), 15-21. https://doi.org/10.1093/bioinformatics/bts635

Fang, Y. et al. (2020). A pan-cancer clinical study of personalized neoantigen vaccine monotherapy in treating patients with various types of advanced solid tumors'. *Clinical cancer research: an official journal of the American Association for Cancer Research, 26*(17), 4511-4520. https://doi.org/10.1158/1078-0432.CCR-19-2881

Garcia-Garijo, A., Fajardo, C. A., & Gros, A. (2019). Determinants for Neoantigen Identification. *Frontiers in immunology, 10*, 1392. https://doi.org/10.3389/fimmu.2019.01392

Gerashchenko, M.V., Lobanov, A. V., & Gladyshev, V. N. (2012). Genome-wide ribosome profiling reveals complex translational regulation in response to oxidative stress. *Proceedings of the National Academy of Sciences of the United States of America. 109*(43), 17394-17399. https://doi.org/10.1073/pnas.1120799109

Grant, G. R. et al. (2011). Comparative analysis of RNA-seq alignment algorithms and the RNA-seq unified mapper (RUM). *Bioinformatics, 27*(18), 2518-2528. https://doi.org/10.1093/bioinformatics/btr427

Gubin, M. M. et al. (2015). Tumor neoantigens: Building a framework for personalized cancer immunotherapy. *Journal of Clinical Investigation*, *125*(9), 3413–3421. https://doi.org/10.1172/JCI80008

Guo, Y., Lei, K., & Tang, L. (2018). Neoantigen vaccine delivery for personalized anti-cancer immunotherapy. *Frontiers in Immunology*, *9*, 1–8. https://doi.org/10.3389/fimmu.2018.01499

Heng, L. et al. (2009). The sequence alignment/map format and SAMtools. *Bioinformatics, 25*(16), 2078-2079. https://doi.org/10.1093/bioinformatics/btp352

Hilf, N. et al. (2019). Actively personalized vaccination trial for newly diagnosed glioblastoma, *Nature*, *565*(7738), 240–245. https://doi.org/10.1038/s41586-018-0810-y

Hodge, K. et al. (2020). Recent developments in neoantigen-based cancer vaccines. *Asian Pacific Journal of Allergy and Immunology*, *38*(2), 91–101. https://doi.org/ 10.12932/AP-120520-0841

Istrail, S. et al. (2004). Comparative immunopeptidomics of humans and their pathogens, *Proceedings of the National Academy of Sciences of the United States of America*, *101*(36), 13268–13272. https://doi.org/10.1073/pnas.0404740101

Jayasinghe, R. G. et al. (2018). Systematic Analysis of Splice-Site-Creating Mutations in Cancer. *Cell reports, 23*(1), 270–281. https://doi.org/ 10.1016/j.celrep.2018.03.052

Kantoff, P. et al. (2010). Sipuleucel-T Immunotherapy for Castration-Resistant Prostate Cancer. *The New England Journal of Medicine,* 363(5), 411-422. https://doi.org/10.1056/NEJMoa1001294

Kahles, A. et al. (2018). Comprehensive analysis of alternative splicing across tumors from 8,705 patients. *Cancer Cell*, *34*(2), 211-224. https://doi.org/10.1016/j.ccell.2018.07.001

Keşmir C., et al. (2002). Prerdiction of proteasome cleavage motifs by neural networks. *Protein Engineering Design & Selection, 15*(4), 287-296. https://doi.org/10.1093/protein/15.4.287

Kim D., et al. (2013). TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biology, 36*(14). https://doi.org/10.1186/gb-2013-14-4-r36

Kluyver, T. et al. (2016). Jupyter Notebooks – a publishing format for reproducible computational workflows. *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, 87-90.

Lachmann, A. et al. (2018). Massive mining of publicly available RNA-seq data from human and mouse. *Nature Communications*, *9*(1). https://doi.org/10.1038/s41467-018-03751-6

Larouche, J. D. et al. (2020). Widespread and tissue-specific expression of endogenous retroelements in human somatic tissues. *Genome Medicine, 12*(1), 1–16. https://doi.org/10.1186/s13073-020-00740-7

Laumont, C. M. et al. (2018). Non-coding regions are the main source of targetable tumor-specific antigens. *Science Translational Medicine*, *10*(470). https://doi.org/ 10.1126/scitranslmed.aau5516

Nielsen, M. et al., (2005). The role of the proteasome in generating cytotoxic T cell epitopes: Insights obtained from improved prediction of proteasomal cleavage. *Immunogenetics, 57*(1-2), 33-41. https://doi.org/ 10.1007/s00251-005-0781-7

Ochsenbein, A. F. (2002). Principles of tumor immunosurveillance and implications for immunotherapy. *Cancer Gene Therapy*, *9*(12), 1043–1055. https://doi.org/10.1038/sj.cgt.7700540

Oiseth, S. J., & Aziz, M. S. (2017). Cancer immunotherapy: a brief review of the history, possibilities, and challenges ahead. *Journal of Cancer Metastasis and Treatment, 3*(10), 250. https://doi.org/10.20517/2394-4722.2017.41

Ott, P. et al. (2017). An Immunogenic Personal Neoantigen Vaccine for patients with melanoma. *Nature 547*, 217-221. https://doi.org/10.1038/nature22991

Paston, S. J., et al. (2021). Cancer Vaccines, Adjuvants, and Delivery Systems. *Frontiers in Immunology, 12.* https://doi.org/10.3389/fimmu.2021.627932

Patro, R., et al. (2017). Salmon: fast and bias-aware quantification of transcript expression using dual-phase inference. *Nature Methods, 14*(4), 417-419. https://doi.org/10.1038/nmeth.4197

Pertea M., et al. (2015). StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. *Nature Biotechnology, 33*, 290-295. https://doi.org/10.1038/nbt.3122

Pertea G. & Pertea M. (2020). GFF Utilities: GffRead and GffCompare. *F1000Research, 9,* ISCB Comm J-304. https://doi/org/10/12688/f1000research.23297.2

Prasad, S., Starck S. R., & Shastri, N. (2016). Presentation of cryptic peptides by MHC I is enhanced by inflammatory stimuli. *Journal of Immonology, 197*(8), 2981-2991. https://doi.org/10.4049/jimmunol.1502045

Prehn, R. T., & Main, J. M. (1957). Immunity to methylcholanthrene-induced sarcomas. *Journal of the National Cancer Institute*, *18*(48), 769–78

Ribatti, D. (2017). The concept of immune surveillance against tumors. *Oncotarget*, *8*(4), 7175–7180. https://doi.org/10.18632/oncotarget.12739

Rizvi, N. A. (2017). The Use of Immunotherapy in the First-Line Treatment of Lung Cancer. *Clinical Advances in Hematology & Oncology*, *15*(3), 282–284.

Sanders H. L., (1968). Marine benthic diversity: a comparative study. *The American Naturalist, 102*(925), 243-282. http://www.jstor.org/stable/2459027

Sahin, U. et al. (2017). Personalized RNA mutanome vaccines mobilize poly-specific therapeutic immunity against cancer. *Nature, 547*(7662), 222–226. https://doi.org/10.1038/nature23003

Schumacher, T. N., & Schreiber, R. D. (2015). Neoantigens in cancer immunotherapy. *Science*, *348*(6230), 69–74. https://doi.org/10.1126/science.aaa4971

Scholz, M. et al. (2017). Phase I clinical trial of sipuleucel-T combined with escalating doses of ipilimumab in progressive metastatic castrate-resistant prostate cancer. *ImmunoTargets and therapy, 6,* 11-16. https://doi.org/10.2147/ITT.S122497

Sheng, Q. et al. (2017). Multi-perspective quality control of Illumina RNA sequencing data analysis, *Briefings in Functional Genomics 16*(4), 194-204. https://doi.org/10.1093/bfgp/elw035

Smart, A. C. et al. (2018). Intron retention is a source of neoepitopes in cancer. *Nature Biotechnology*, *36*(11), 1056–1063. https://doi.org/10.1038/nbt.4239

Sriram, A., Bohlen, J., & Teleman, A. A. (2018). Translation acrobatics: how cancer cells exploit alternate modes of translational initiation. *EMBO reports*, *19*(10). https://doi.org/10.15252/embr.201845947

Starck, S. R. et al. (2012). Leucine-tRNA initiates at CUG start codons for protein synthesis and presentation by MHC class I. *Science*, *336*(6089), 1719–1723. https://doi.org/10.1126/science.1220270

Starck, S. R. et al. (2016). Translation from the 5′ untranslated region shapes the integrated stress response. *EMBO reports*, *17*(10), 1374–1395. https://doi.org/10.15252/embr.201642195

Stein, M. K., et al. (2019). Tumor mutational burden is site specific in non-small-cell lung cancer and is highest in lung adenocarcinoma brain metastases. *American Society of Clinical Oncology,* 1-13. https://doi.org/10.1200/PO.18.00376

Sun, J. et al. (2019). Neoantigen vaccine generates intratumoral T cell responses in phase Ib glioblastoma trial. *Nature, 565*(7738), 234–239. https://doi.org/10.1038/s41586-018-0792-9

Voelker, R. (2020). Immunotherapy is now first-line therapy for some colorectal cancers. *Journal of the American Medical Association*, *323*(11), 1034. https://doi.org/10.1001/jama.2020.2449

Wang, K. et al. (2010). MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Research, 38*(18), e178. https://doi.org/10.1093/nar/gkq622

Wei, Z. et al. (2019). The landscape of tumor fusion neoantigens: a pan-cancer analysis. *iScience*, *21*, 249–260. https://doi.org/10.1016/j.isci.2019.10.028

Wu, T. D. et al. (2010). Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics, 26*(7), 873-881. https://doi.org/10.1093/bioinformatics/btq057

Yates, A. D. et al. (2020). Ensembl. *Nucleic Acids Research, 48*(D1), D682-D688. https://doi.org/10.1093/nar/gkz966

Yewdell, J. W., Dersh, D., & Fåhraeus, R. (2019). Peptide channeling: The key to MHC class I immunosurveillance. *Trends in Cell Biology*, *29*(12), 929–939. https://doi.org/10.1016/j.tcb.2019.09.004

Zhou, Q. et al. (2018). RNA-QC-chain: Comprehensive and fast quality control for RNA-seq data. *BMC Genomics, 19*(1), 1-10. https://doi.org/10.1186/s12864-018-4503-6

Zugazagoitia, J. et al. (2016). Current challenges in cancer treatment. *Clinical Therapeutics*, *38*(7), 1551–1566. https://doi.org/10.1016/j.clinthera.2016.03.026