APPLICATIONS OF MACHINE LEARNING IN OMICS AND COMPUTER VISION

by

MICHAEL FRANCIS SKARO

(Under the Direction of JONATHAN ARNOLD)

ABSTRACT

Bioinformatics is the computational arm of life sciences research. It is comprised of computer scientists studying imaging, omics, mathematics, and statistics. In this dissertation, I have compiled small pieces from each of these scientific disciplines to develop and deploy two machine learning architectures that address open questions in cancer metastasis and systems biology. In the first experimental chapter, we developed a first-in-class tree-based classifier capable of predicting site-specific metastases arising from primary tumors in 16 cancer types. Our model extracts and analyzes the biological determinants of cancer metastases from transcriptomic profiling data to model the biological phenomena of metastatic organotropism. We expanded the core feature selection algorithm into an end-to-end omics preprocessing and feature selection software. We validated our design on the tumor methylation array data by selecting cancer type-specific methylation array probes as input features for an artificial neural network. The MetNet architecture was trained to classify cancer types and identify organ of origin in cancers of unknown primary (CUPs). Finally, we developed a high throughput object detection suite for microscopy images of fungal structures. We established and deployed a first-in-class instance segmentation tool for the identification of arbuscular mycorrhizal fungi colonizing terrestrial plant roots. Our

model is available for public use on amazon webservices to support the greater scientific efforts of mycologists.

INDEX WORDS: Bioinformatics, cancer, metastasis, metastatic organotropism, arbuscular mycorrhizal fungi, microscopy, images, image segmentation, instance segmentation.

APPLICATIONS OF MACHINE LEARNING IN OMICS AND COMPUTER VISION

by

MICHAEL FRANCIS SKARO

Bachelor of Science, Stevenson University, 2015

Master of Science, Johns Hopkins University, 2018

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2022

APPLICATIONS OF MACHINE LEARNING IN OMICS AND COMPUTER VISION

by

MICHAEL FRANCIS SKARO

Major Professor:	Jonathan Arnold
Committee:	Mandi Murph
	Liang Liu
	Shaying Zhao
	Jefferey Bennetzen

Electronic Version Approved:

Ron Walcott
Vice Provost for Graduate Education and Dean of the Graduate School
The University of Georgia
May 2022

DEDICATION

I dedicate my doctoral dissertation to my wife Kelsey, my mother Karen, my father John, and brother Jonathan. All of you have collectively created a support system for me. I know I could not have completed my education without the sustained support from my family. I would like to thank each of you for your love and care.

A SINCERE THANKS

I would like to thank Jonathan Arnold and my committee. Jonathan, you have stood by me for four years, driven me to success, and supported me in my pursuit of completing my doctoral dissertation. No words can describe my debt to you. I am honored to have been a graduate student in your lab. I am proud to join the ranks of your previous mentees and am lucky to have had such an excellent scientist to model myself after. To my committee members; Mandi Murph, Liang Liu, Shaying Zhao, and Jeff Bennetzen. Thank you all for your time. Your thought-provoking suggestions, constructive criticism, and your scientific expertise offered to me during my committee meetings and in one-on-one sessions were invaluable. Mandi, you challenged me, you never accepted mediocrity from me and helped me strive for excellence in my day-to-day research. I am a better scientist because of you. Liang, you required that I ask hard questions and think deeply about my research. Thank you for your thoughts and statistical expertise. Your suggestions in my very first committee meeting drove my methods development work. You and Jonathan together led me towards machine learning in genomics, the field I now call home. Casey, you

pushed me to better my writing and scientific communication. You dared me to broaden my expertise in bioinformatics and taught me that the only science, is reproducible science. Jeff, your humble and pragmatic approaches to research are elegant. Reading your published literature and listening to your project design ideas in our monthly meetings set the standard for my work. I found myself asking “How would Jeff do this?” when I was struggling with my research. To each of you, thank you again. I am lucky to have worked with each of you and wish all of you the best.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

LITERATURE REVIEW AND INTRODUCTION

In the past 60 years the development of machine learning (ML) algorithms has exponentially expanded and their applications have been integrated into every scientific discipline[1]. The expansion of this field is directly connected to the exponential growth in data generation[2]. It is estimated that 2.5 quintillion bytes of data are generated everyday[3]. Scientists studying life sciences, have seen an explosion of sequencing data and microscope images as two of the major forms of digital information being experimentally generated[4]. The applications of ML in the biological sciences have unsurprisingly been focused on developing new models and tools to analyze these data[5]. While all ML algorithms fit models to data, the specific methods and applications are very diverse. Each application requires the design of a fit for purpose approach. Designing and executing a fit for purpose ML algorithm occurs in multiple phases. These phases include data mining, data preprocessing, feature selection, dimensionality reduction, algorithm training, validation and testing, and architecture deployment. We will visit these phases and review the emerging algorithms developed in these phases for applications in bioinformatics. Applications of modeling methods, computer vision, clustering, as well as genetic, deterministic, and stochastic heuristics for feature selection will be presented. Finally, we will introduce the knowledge gaps we aim to fill by developing new algorithms for feature selection and by modifying existing methods in machine learning and bioinformatics.

1.1 Machine Learning in Genomics begins with data mining:

Genomics research makes up an extremely large proportion of bioinformatics. High throughput sequencing methods have increased the volume and veracity of assembled genome sequences at an exponential rate[4]. By design, it is our field standard to deposit assembled genomes, transcriptomes, epigenomes, and sample metadata into central repositories to benefit the research efforts of the greater scientific community. The size and diversity of the publicly available genomics data have opened the door for new genomics research questions to be asked that can only be answered by data mining these extremely large resources[6-9].

Data mining public resources requires a significant interdisciplinary effort. Designing the software to deposit and access public data in an application program interface (API) is accomplished by software engineers. APIs manipulate the structure of data to optimize for the flow of information. Bioinformaticians download, assemble, and reconfigure genomics data in its model-ready format. As ML algorithms require many samples to properly represent and learn from the diversity of instances in public databases, they often require mining samples from tens, if not hundreds, of different experimental workflows. Further, it is common for an investigator to produce his or her own in-house sample sets that they intend to use the ML tools to analyze during their experiments. Combining all the public and in-house samples creates a heterogenous set of information that requires further manipulation for an ML algorithm to learn. The structure, observation ranges, and classes represented in data arising from different sources are often non-standardized across the experiments[10]. Experimental standardization and data normalization are key steps in data preprocessing[10]. Data preprocessing unifies the structure of raw or unprocessed information into a model-ready form.

In the case of omics data, sequenced sample files are represented in raw FASTQ format, containing nucleotide strings and base qualities reported from a sequencing machine. Raw FASTQ data require considerable data preprocessing before it is ready for analysis. The phases of FASTQ preprocessing are quality control, trimming, mapping, quantification, and data scaling. These steps standardize information input into an ML algorithm[11]. Optimizing these standard steps in data preprocessing are central areas of research in bioinformatics. The applications of machine learning in each of these areas are growing rapidly.

1.2 ML in Sequence Quality Control, an emerging discipline:

Controlling the quality of next generation sequencing data is complex. The most well-known quality assessment tool in genomics is the software FastQC[12]. This software conducts a suite of analyses that address position-dependent biases, sequencing adapter contamination and DNA over amplification. Large scale genomics projects have complemented this software with multi-QC, a helper tool that gives the user a global estimate of the experimental data quality[13]. The drawbacks of these approaches are that they require modification and customization for every sequencing analysis. The ideal quality control suite would ingest any raw data and automatically identify poor quality sequence in the data. These kinds of architectures are currently under development in the emerging field of omics quality control using machine learning. Published in a recent report in *Genome Biology*, seqQscorer, is a tree-based learning and deep learning software developed to conduct quality control on sequencing data[14]. The seqQscorer software is trained on raw human and mouse ENCODE project sequence data, producing two models; one trained on individual scoring methods reported from FastQC, while the second was an ensemble model trained on the conglomerate estimations using MultiQC. The outputs of these models are formatted similarly to the style bioinformaticians are accustomed to analyzing in the FastQC and MultiQC

results. The model automatically identifies regions of DNA sequence that it considers low quality or with adapter content that needs to be trimmed off, without a reference or adapter table. The low-quality and adapter sequences are automatically removed using the trained model. The advantages of this approach are the ease of use and accessibility for the experimentalists using sequence data to answer their experimental questions. The drawbacks of this model are the limited scope and variable loss. While mouse and human models are heavily studied in the disciplines of human health, its application into other organism's sequences has not been validated. Further, all models suffer from some loss, the improper removal of valuable sequence data could result in miscalculations that are difficult to resolve. This type-one error would be particularly problematic in genome assembly.

1.3 ML in genome assembly:

Genome assembly (GA) is a central research focus in bioinformatics. The field is focused on two main goals. First, the construction of useful genome assemblies in organisms that have not been thoroughly characterized. §Second, the sharpening of the completeness of previously assembled genomes. These goals are accomplished using two approaches: de novo assembly and reference-based assembly. These approaches often complement each other, benefitting from the scaffoldings of other genome assemblies but, in the extreme case where no other organism in its class has ever been fully sequenced, the assembly of an organism's genome is completely de novo[15,16].

In a computational sense, de novo GA is an incredibly hard problem, NP-hard, in fact. NP hard problems are a class of computational problems where a polynomial time solution has not been resolved or is impossible to derive with our current methods[17]. In these classes of problems, investigators have two options for reducing the computational time complexity for approaching

acceptable solutions. The first is to use a compression algorithm to reduce the data into latent representations of the complete dataset. As lossless compression is the purest form of machine intelligence, there is currently a worldwide competition hosted by Marcus Hutter to compress a 1 GB file of Wikipedia text[18]. The field of genomics has inherited from the break through work of Burrows and Wheeler describing their first in class compression algorithm, the Burrows-wheeler transform[19]. This compression algorithm uses a set of permutations to compress and organize an organism genomes into an easily traversable tree[19]. While this operation is computationally costly, this operation needs only to be computed once and reduces the mapping of newly sequenced reads time complexity to a linear time search of the indexed genome[19].

The second approach to solving NP hard problems is to make a heuristic to reduce the computational complexity of the problems. Heuristic based GA approaches dominate the field but AI and machine learning based approaches have begun to emerge with promising results that match and, in some cases, exceed the performance of the heuristic-based approaches[20-22]. Side-by-side methods utilizing multi-omics and machine learning are the current best approaches for GA[21]. The combination of long-read sequencing and Hi-C data have opened the door for large scale side-by-side methods to redefine the completeness of the most central model organisms in genomics[23-29]. These efforts are at the forefront of developing reference genomes in staple crops that will require significant genomics engineering as we begin to deal with climate change and topsoil depletion[30-34]. Assembling the highest quality reference genomes offers the highest confidence in the annotation of the variants in genotypes of the organisms[32].

1.4 ML in de novo genome assembly:

De novo GA represents the recovery and accurate assembly of a genome without a reliable reference to scaffold the approach. In the early days of de novo GA, the focus of the research was

to resolve a single taxon such as the human genome project[16,35]. Software to assemble these genomes were primarily derived from using sequence overlaps to configure assembly graphs that would reconstruct continuous stretches of unbroken sequence information[36,37]. New approaches are attempting to attack this problem using recurrent neural networks for de novo assembly of single taxa[38]. However, as sequencing technology has been rapidly advancing, the field of metagenomics is exploding. One new challenge for de novo assembly is no longer resolving a single taxon from DNA of a single organism. The new challenge for de novo genome assembly lies in the assignment of reads to the correct metagenomic taxa and parallel reconstruction of thousands of taxa's individual genomes in a soup of DNA[39]. Direct approaches at solving this challenge were taken on by software teams developing tools such as SPAdes, SOAPdenovo2, ALLPATHS, and MaSuRCA [37,40-42]. These software were designed to separate and assemble each individual genome. This technique proved to be ineffective resulting in low quality assemblies when assessed by software like QUAST, and GAGE[43,44]. This sparked the development of a new generation of short read metagenomic specific tools including MetaSPAdes, MetaVelvet, IDBA-UD, and Ray Meta[45-48]. These approaches were fit-forpurpose ML algorithms designed to handle the inherent imbalance of DNA that would be present from each of the taxa in the soup of samples. While these algorithms have shown a marked improvement in contig resolution and confidence, they suffer from short read inaccuracy and assembled genomes needing several rounds of polishing with HI-C and, long read sequencing. Further, these processes are tedious and require many man hours to resolve highly repetitive regions[39].

The next stage for this research area is clear, an end-to-end automated assembly using deep learning. The evolution of recurrent convolutional neural networks has increased the speed and resolution of GA. The process is simple in design but has been shown to achieve equal or better

than field-standard genome assemblies. Conceptually, a model will train on real and simulated read sets. Seed reads will be randomly drawn from the mixed set. For each read set drawn, an RNN capable of modeling the corresponding sequence is built. The training continues until each RNN can classify the organism and location of reads in the genome above a pre-defined threshold. Finally, each of the remaining read sets are concatenated. Each concatenation introduces a cascade of new RNN training loops until the misclassification rate is reduced to a stable level[38]. The quality of the assemblies are judged against high confidence genomes assembled with multi-omics technology using a local similarity algorithm called SIM[49]. RNNs have been used side-by-side with overlap layout consensus approaches[50]. This approach lays the foundation to pair a neural network system implemented together with assemblers in order to investigate the effects a read grouping approach has on assembly performance[50].

1.5 ML in reference-based genome assembly:

In the case of reference-based alignment, dynamic programming is the primary tool used to assemble the genome. Well known aligners such as Smith-Waterman, a variant of Needleman-Wunsch alignment, uses a distance scoring method[51]. The Burrows-Wheeler aligner and Bowtie systematically permute string arrays within the genome. This process creates streaks of redundant base pairs as a pre-process to optimize the string-matching searches of a similarly permuted reference genome[51-54]. Classically, substrings of genomes were organized as a suffix tree instantiated as a directed graph that was easily traversed to resolve global alignment of sub-strings in a large text[55]. Without indexing, one can use dynamic programming to find all the local alignments between a text T and a pattern P in O(|T||P|) time, but this would be too slow when the text is of genomic scale[4]. The most revolutionary discovery in computational genomics came in the form of the Burrows-Wheeler transform[19]. The BWT is a lossless permutation of substrings

such that the resulting matrix is easily organized into an alphabetically sorted array of strings with a computational complexity of O(log n). The genius of the BWT lies in the tendency for the algorithm to produce repeated sequences of characters. The indexed strings form a suffix array which searchable in O(n) time complexity and can be easily alphabetically matched. The compression algorithm using the FM-index formed by the BWT has been extended into the E-BWT and has been shown to be the most successful compression of organism genomes. This work opened the door for a number of new methods exploring compression intelligence in GA[56].

The first index for read sets method based on the EBWT was BEETL, followed by RopeBWT. The BEETL and RopeBWT algorithms both use a heuristic to reduce the number of runs in the EBWT. The software put the separator characters at the ends of reads into lexicographic order. These methods have been extended recently by Gaige et al. in which they build a directed edge-labeled tree of the human genome[57]. Their algorithm works by taking a partially assembled genome as a trunk and grafting it on the reads that map to the starting positions of their alignments. Their work is being applied to solve open questions in pan-genomics read assembly[58-63].

Mechanic algorithms are developed around GA to correct and smooth errors in assembly. Short read assembly suffers from biological challenges and inherent constraints due to the length of the reads they are trying to align or assemble. Biological challenges in genome assembly generally stem from regions in a chromosome that are repetitive or are under heavy evolutionary pressure. Specifically in plants, the pervasiveness and persistence of transposable elements plagues short- read GA. A sequencing machine will make systematic errors when it estimates the quality score of each base call. The most common score assessment software is Base Quality Score Recalibration algorithm (BQSR)[64]. The BQSR algorithm applies a two-phase optimization-maximization technique to model these errors empirically and adjust the quality scores recursively.

First, the BQSR builds a model of covariation based on the input data and a set of known variants from the organism of investigation. The BQSR optimization step adjusts the base quality scores in the data, producing a new BAM file. This cycle repeats until the known variation from the organism can be accurately used to mask out bases at sites of expected variation. Once an optimized criterion is met, any remaining mismatches that occurred outside masked locations pass filter and are considered true variants.

1.6 ML in variant calling and annotation:

A genomic variant is a nucleotide alteration from an organism's reference genome. The most common variations are the smallest, base pair substitutions. In genetic regions, the effects of these changes range from silent to pathological variations that drive disease[65-67]. Outside of genetic regions, the effects of single nucleotide variants are difficult to validate. Therefore, it is paramount that the software that calls these variants and the algorithms that assess their quality and molecular impact are both sensitive and specific. ML applications in variant calling are applied post alignment.

The first step in genomic variant calling is to preprocess the data to ensure data quality during alignment and coverage quantification. Pre-alignment quality scoring is directly measured on the machine by Phred score. The Phred Score reported for a base pair call is reported as the Q score. The Q score is defined as $-10 \log_{10}(P)$, where P is the probability, the machine made an erroneous base call. While this system is strong and reproducible, the sequencing by synthesis employed by Illumina machine bias errors towards the ends of the reads and thus, the reported Phred score is usually unreliable for end-of-read data. The poor end-of-read quality is usually attributed to Illumina adapter sequences and reagent exhaustion. The removal of adapter sequences relies on unsupervised learning from string similarity matrices against a known background. Short-

read sequence data can be preprocessed with a series of steps in the GTAK best practices[68]. Raw reads are converted to an unmapped BAM file. The uBAM file is passed into the mark Illumina Adapter Sequences software. The software is built on top of the Levenshtein distance model using fuzzy string matching for adapter sequence identification. As the expected position in reads, adapter length and sequence identity are stored, the algorithm can estimate the probability that a substring belongs to harvested organism DNA or is Illumina adapter content. A position-specific similarity matrix is constructed against the known Illumina adapter sequences and high confidence matches are marked in each of the read groups. Adapters are removed with trimming software and passed to read alignment[69]. Variants passing filter are called and annotated by ML software.

Variant calling with ML on the alignment file is an established application of ML in Bioinformatics. Early variant calling frameworks were optimized for short read alignments and were based directly from the alignment file: DinDel, PolyBayes, samtools/BCFTools, GATK-Haplotype Caller, GATK-Unified Genotyper. However, other ML models call variants based on the literal sequences of reads aligned to the target instead of the precise alignment[52,68,70-75]. A popular Bayesian framework, freebayes, is a generalized haplotype-based variant detection algorithm with two main applications. Freebayes uses three inputs, the alignment files, the read Phred 33 scores and the reference genome to generate point positions it finds to be putatively polymorphic[70]. Freebayes is best applied when priors may be supplied. Data-mined VCF files or copy number maps in a BED file allow freebayes to determine non-uniform ploidy variation across the samples in its analyzes[70]. Statistical ML frameworks works are being replaced by larger deep learning architectures in other areas of research and, unsurprisingly, in the field of genomics, deep learning is similarly the current gold standard.

Google's Deep Variant has paved the way for deep learning models to emerge in the variant caller space[76]. The Deep Variant (DV) architecture far outperforms previous variant calling methods in recognizing substitutions, tandem repeats, insertions and deletions. The DV framework workflow begins with an aligned reference genome. DV uses a series of candidate variant selection steps followed by the collection of images drawn from read pileups. The images of read pileups are used as the input layer into a convolutional neural network optimized with stochastic gradient descent. The trained CNN reports the genotype likelihoods as confidence scores to report the variant calls[76]. The major advance in the DV framework is the generalization of the CNN. The trained CNN model generalizes across genome builds and mammalian species, allowing nonhuman sequencing projects to benefit from the wealth of human ground-truth data. The architecture has been since adapted with long read data to advance calling larger segments of DNA sequence called copy number variations (CNVs).

Long-read sequencing is changing the way we approach variant calling. Statistical and deep learning frameworks have been developed in the last five years specifically for long read sequencing data and most recently, single molecule sequencing (SMS). The most prominent workflow being the software Clairvoyant, a multitask convolutional deep neural network specifically designed for variant calling with SMS reads[77]. Clairvoyant was trained on the well characterized human cell line NA12878 to call variants. The variant caller extended the work from DV in that it especially effective in characterizing data from long read sequencing that are best for capturing long range CNV variation. The software expanded the known variation in the NA12878 sample by over 3000 variants missed by variant callers built on short read sequence data. The method performed in the excellent range of F1-score on whole genome analysis at 98.65%. By outperforming similar software, the method placed itself as the gold standard for Deep learning-

based variant calling on long read sequencing data [77]. End-to-end deep learning variant callers both in short read and long read experiment are emerging as the preferred feature engineering software in genomics pipelines. The challenge of selecting which variants are the most valuable for the biological experiment is accomplished using feature selection algorithms. Feature selection algorithm development is dominated by statistical machine learning and deep learning frameworks.

1.7 ML for feature selection:

Feature selection is a dimensionality reduction technique that is commonly used in the fields of bioinformatics and computational biology. This technique aims to select a subset of relevant features from the original set of features according to some criteria. Examples of feature selection techniques to include Information Gain in decision trees, Relief, Chi Squares, Fisher Score, and Lasso. This problem arises most commonly in bioinformatics where a sequencing machine will output millions of reads that cover thousands of genes, but we only can afford to study a small number of samples. In the case of gene expression data, feature selection is necessary as the sequencing data usually contains many redundant, lowly expressed, or noisy transcripts. In practice the data may be fully labeled or partially characterized. This leads to the use of supervised methods for labeled data and unsupervised methods for unlabeled samples. The supervised feature selection is the process of reducing the feature subset to latent variables measuring the importance of the variables for discriminating between classes in the labeled data. In contrast, unlabeled data requires algorithms to draw intrinsic differences in the sample level by measuring patterns in the features such as innate information like variance, distance, or stratification.

Using machine learning to tackle feature selection is a common approach in computational biology. A heuristic was proposed by Yu and Liu that allots features into four major categories;

(I) completely irrelevant and noisy features, (II) weakly relevant and redundant features, (III) weakly relevant and non-redundant features, and (IV) strongly relevant features. They suggest the theoretically optimal set of features are a combination of sets III and IV where, you can describe your classes using the strongly relevant and non-redundant features[78]. It is important to note that the removal of a feature that is redundant or noisy is due to its possible strong correlations with possible irrelevant features that will contribute to loss in the architecture. A single gene or transcript alone may have little affect but in combination with others, the signal may be significantly stronger. In this section we will discuss unsupervised and supervised methods for gene selection with special attention to previously developed methods in transcriptomics, microarray, single cell, and genetic networks.

Minimal redundancy and maximum relevance (mRMR) analysis is an unsupervised method that is best applied to gene expression data when the objective is to discriminate between two classes of data where the gene expression sets are produced. The features are selected one by one by applying a greedy search to maximize one of two commonly used types of the objective functions; MID (Mutual Information Difference criterion) or MIQ (Mutual Information Quotient criterion). These draw one of two values, either the difference or the quotient of relevance and redundancy, respectively[79,80]. This method has been adapted an advanced since Peng et al, by Radovic et al developing the TMRMR-M and TMRMR-C feature selection algorithms[80]. This software is implemented as an easily implementable MATLAB package with side-by-side data preprocessing resources that have been subsequently added on.

Dimensionality reduction methods such as principal component analysis (PCA), t-stochastic neighbor embedding (t-SNE), and uniform manifold approximation and projection (UMAP) reduce the complexity of a gene selection problem by engineering sets of latent variables

with the objective of observing the maximum variance of the experimental features in a reduced set of orthogonally engineered dimensions. PCA for gene selection is used in both combination with supervised methods and has been modified for a singular approach for gene selection. As a singular approach, Ma et al use PCA for selection of differential gene pathways[81]. But it is currently used more prominently as a first a pass in filter and wrapper supervised methods[82,83]. T-sne dimensionality reduction is the most prominent method for gene selection in single cell transcriptomics. The method excels at revealing local structure in high-dimensional data, but naive applications often suffer from misrepresentation of global expression structures and has thus fallen out of favor for UMAP[84,85]. The UMAP algorithm is a geometric machine learning algorithm that preserves local and global data structure better than t-SNE, with a shorter run time in multiple gene selection experiments across single-cell omics technologies[86].

Factor Analysis (FA) models, specifically Bayesian FA models are used to describe a large number of observed variables by a smaller number of hidden variables called factors. The models are used for gene selection because all correlation drawn between observed variables can be explained by combinations of common factors. Classical FA models differ from Bayesian FA as Bayesian FA methods tend to infer sparse networks by enforcing sparsity through the calculated priors. In contrast, the classical FA methods enforce sparsity through reversible matrix rotations and transpositions. Purnara et al. showed in cases where a Bayesian FA model does not impose strict sparsity through the priors it can be used for accurate gene network reconstruction and investigation of gene regulatory networks[87]. They applied FA using a two-layer network to model gene expression regulation. The network was comprised of unobserved TF variables in the first layer and observed gene expression variables in the second layer where TFs are connected to regulated genes by weighted edges. Their work showed that the assumption of the activity levels

of transcription factors (TFs) being proportional to their mRNA expression is unlikely but, using FA they were able to reconstruct unobserved activity profiles of TFs from the expression profiles of target genes[87].

Linear Discriminant Analysis (LDA) has been commonly used for gene feature selection from microarray data[88]. LDA is used for feature selection in genomics because the sample sizes are generally low and the high dimensional sample features are projected into a lower dimensional space such that the classes separate more easily. LDA is most commonly used as a first pass in wrapper methods but has been used a single functional method for gene selection. Huerta et al developed a hybrid approach for LDA as a preprocessing step for analysis by a genetic algorithm for gene selection in prostate and lung cancer microarray data sets[88]. In their study they were able to feed series of one-hundred gene sets into the LDA to discriminate between tumor and normal data prior to gene selection in the genetic optimization algorithm. This preprocessing step increased the computational efficiency of the agents in the subsequent search[88].

Singular Value Decomposition (SVD) is classically used or gene selection in cancer data. The SVD algorithm is an unsupervised approach for finding the most important or central genes for the disease. The central genes are aptly named as the algorithm determines the most informative genes are the ones closest to the corresponding cluster's centers. These genes are then reduced and the algorithm constructs a pruned dataset of the same samples but with less dimensionality[89]. The algorithm may be reclusively called until only the centroids remain or in practice this algorithm may be used in combination with a heuristic. The centers may be modified posthumously to only the differentially expressed candidates and the SVD will recursively search for a stable differentially expressed cluster center[89].

Supervised methods of feature selection have three sub-categories of approaches. Supervised methods may be divided into wrapper, filter, or intrinsic. While wrapper methods are extremely costly these algorithms benefit from exploring different combinations of features to fit a model. The combinations may be explored by adding and or removing predictors to find the optimal combination that maximizes model performance. The implicit nature of wrapper methods lends them to gene selection problems. Previous literature suggests wrapper methods should be preceded by a dimensionality reduction step to reduce the co-linear features[82]. Riverol et al demonstrated the viability of preceding wrapper methods with dimensionality reduction steps for gene selection by ranking genes with Pearson correlation matrix. They identified clusters with PCA as a filtering step prior to feeding selected orthogonal sets of features into classifiers trained to differentiate between tumor and normal data. They expanded the applicability of their method for classification on protein IHC and microarray transcriptomics data[82].

Forward selection for gene selection is especially popular for defining genetic networks, gene regulation and understanding metabolic reprogramming in cancer[90,91]. In these models the forward selection algorithm starts with one predictor and adds more iteratively. At each subsequent iteration, the best of the remaining original predictors are added based on performance criteria. This method is specifically useful in graph traversal. Genetic networks may have extremely high dimensionality, high interconnectivity and many nodes may have extremely high betweenness centrality. Traversing through these nodes may cause the minimal spanning tree from one gene to another to become extremely large. Forward selection of gene expression data using knowledge graph traversal has been demonstrated in disease state prediction[92].

In contrast to forward selection, backward elimination begins with a filtering step in which we only consider variables that meet a criterion, build successive models by eliminating the lowest

ranking or lowest scoring variables until our model does not improve. A common criterion for gene selection in backward elimination models are pvalue or fold change in a differential expression experiment[93]. The foundation of canonical transcriptomics based models use a filtering step for differentially expressed genes and built backward elimination models from the DEG. Interestingly, in some cases a simple t-score is strong enough to separate classes in data using a backward elimination approach[94].

Step-wise selection methods are bi-directional and are a combination of forward selection and backward elimination. This approach is also known as a hybrid approach in which more than one kind of feature selection architecture is employed for gene selection. Step-wise eliminations suffer from the computational cost of deriving a new model for edge cases but are advantageous to forward and back selection as they never add an eliminated feature back into consideration. Step wise selection is uncommon in transcriptomics analyses however, they are extremely useful in GWAS analyses across multiple domains[95-97].

1.8 Evaluation of methods for feature selection:

Evaluating the performance of feature selection methods follows standard formats. The loss and cost functions of an algorithm are optimized using the training and validation sets. The parameters and hyperparameters that modify the functions in an algorithm are serially tuned, measures of performance are optimized, and inductive bias and bias variance are minimized.

The outputs of machine learning model will deviate from the ground truth values or target ranges. The statistical functions that measure these deviations between the theoretically perfect outputs and the ones observed in practice are called loss functions[98]. In supervised classifications a loss functions measure the deviation directly against the encoded class values as cross entropy.

With respect to regression problems estimating a measure, the loss is commonly measured as mean square error.

Optimizing the parameters of a mathematical function is is synonymous to feature selection. However, while training an ML model, there are tunable values can be adjusted in between training trials to achieve the best performance. These values are known as hyperparameters, they are adjustable values that unlike canonical features in the model, they remain constant during each training trial[99]. Hyperparameters still have an impact on the training of the model and its performance and in a fit-for-purpose model design, the optimal hyperparameters are searched for using a gird search, serial adjustment, or expectation maximization subroutine[99]. A common example of a hyperparameter is the learning rate[99]. The learning rate refers to the rate at which function parameters are adjusted in the training process[99].

Training is the process of adjusting parameters such that the model recognizes the features vector patterns that characterize observation classes. The training process isolates a large enough proportion of the data set such that the model can recognize the underlying diversity in feature vectors. The parameter adjustments are fit to training set and are recursively tested on the validation data set. The validation data set is used to monitor but not influence the training process to detect potential overfitting. The validation data set allows the model to make serial modifications without tainting the test set and ruining the integrity of the test results. The test set is the proof of generalization for the model. The reported performance of a machine learning model is the performance on the test set. The measures of performance depend on the type of dataset but generally describe how often a model identified a sample into the correct class versus how often it misidentified a sample.

Measuring the performance of an ML model on the test data set demonstrates to the potential user what their expectations should be when using the model on their dataset. It allows the user to decide whether the model is fit for their purpose and if the level of loss is acceptable for their problem. There are five common measures of performance; accuracy, precision, recall, F-1 measure and Mathew's correlation co-efficient. The true positive rate (TPRate) represents the true positives called by the models amongst the test set. The false positive rate (FPRate) represents the false positives called by the model amongst the test set. Precision is the fraction of true positives amongst all of the instances which were predicted to belong in a certain class:

$$Precision = \frac{TP}{(TP + FP)}$$

Recall is the fraction of instances which were predicted to belong to a class divided by all of the instances that truly belonged in the class.

$$Recall = \frac{TP}{(TP + FN)}$$

The F-Measure is the combination of precision and recall and is optimally weighted for model assessment:

$$F\beta = (1 + \beta^2)\frac{Precision \cdot Recall}{(\beta^2 \cdot Precision) + Recall}$$

The β parameter is tunable. It allows for a trade-off between precision and recall. β<1 weights the assessment of precision higher, while β>1 allocates more weight to recall. The Mathew correlation coefficient (MCC) assessment is used in machine learning as a measure of the quality of binary classifications[101]. MCC is a balanced measure and it is often used if the class sizes are highly imbalanced. The MCC returns a value between −1 and +1. A coefficient close to 1 represents a strong correlation between prediction and observation label, a coefficient of 0 represents a random

prediction, and a coefficient of −1 indicates a failure to find correlation between the predicted class and observation label.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{((TP + FP)(TP + FN)(TN + FP)(TN + FN))}}$$

A receiver operating characteristic area is a measure showing the performance of a classification model at all classification thresholds. This curve plots two parameters, TPRate vs FPRate. The area under the curve measures the entire two-dimensional area underneath the entire ROC curve. Finally, the PRCArea, a precision-recall curve shows the relationship between precision (positive predictive value) and recall (model sensitivity). The PRCArea is the entire two-dimensional area underneath the curve.

The cost functions of unsupervised methods are optimized however the goodness of fit in approaches such as clustering is not adjusted to a ground truth value. Instead of fitting a model to a ground truth, unsupervised algorithms measure the goodness of fit based on homogeneity of the groupings or clusters they generate. It is often the case that the number of groupings has the highest influence on the performance of an unsupervised algorithm. A common example of this measure is used in the K-means algorithm.[102] The k-means algorithm estimates the classes of the data by randomly generating k number of cluster centroids where k is a hyperparameter pre-defined prior to clustering. The algorithm assigns the samples closest to these centroids into the associated clusters based on Euclidean or Manhattan distance in two dimensional or N dimensional space, respectively. The cost function is adjusted such that the centroid is updated to represent the newest "center" of each cluster. The cluster number is optimized by serially adjusting the hyperparameter and calculating the average within-cluster sum of square distances from the centroids (WCSS)[102].

In all ML modeling, assumptions must be made to constrain the problems you are applying an algorithm to solve. The inductive bias of a model refers to the set of assumptions made by the model to learn the solutions to the problems. The assumptions will make certain outcomes more likely than others, and this is known as the bias of the model. A common bias occurs in recurrent models[103]. Recurrent models suffer from bias toward the order or sequence in which data is loaded into the model. The model will, with all other parameters equal, bias towards a solution that favors data fed into the model at the beginning of the sequence[103,104]. We resolve this bias by adjusting the parameter weights and slowing the learning rates in recurrent models.

Bias-variance is a balance stricken between the number of constraints on a model to reproduce very precise outputs versus fewer constraints that will produce a larger variance in the prediction outputs. If a model has a high bias, the model may be very constrained into a highly specific problem. While a model with a high variance may be broader but suffers from the possibility of catastrophic unlearning. Catastrophic unlearning occurs when a trained model may lose performance when a new class is added to the data. For example, if a model trained to identify dogs in an image is subsequently trained to identify ducks, it may suffer a catastrophic loss in performance in identifying dogs in the next applications. Catastrophic unlearning is extremely common in convolutional neural networks[105]. This has led to the adoption of super data sets. These are extremely large, labeled datasets with a diverse set of classes[106]. The model is trained to identify all the classes at once. Training these models can be extremely costly however, it uncovered the most powerful applications of deep learning: transfer learning and knowledge inheritance[105]. Transfer learning allows the user to take advantage of the pretrained networks and harness learned underlying residual information from previous models. The knowledge inheritance opens the

opportunity to tune previously trained models and deploy them on small datasets that may not be suited for deep learning on their own[107].

1.9 Deep Learning and current methods for segmentation:

Deep Learning (DL) has been an extremely successful branch of artificial intelligence in the field of computer vision (CV). In the past 2 years over 14,000 papers with a diverse set of deep neural network architectures have been published in computer vision in the CVPR conference. The dominating architecture for computer vision for many years has been the convolutional neural network (CNN). CNNs can become extremely complex in their design but have a very simple core network structure of neurons. CNNs have two kinds of neuron layers, first the convolutional layer and second a pooling layer[108]. A filter is passed across and down the pixels of an image performing small matrix multiplication operations called convolutions[108]. The filter is comprised of four core properties; the kernel dimensions, padding, stride, bias, initial weights and number of output channels[108]. In successive strides, the filter is slid across image patches and conducts the matrix multiplications. A pooling layer is a subsampling of the patches that have been operated upon by the convolutional filter[108]. The matrix is combined with the underlying image values and the maximum value from each convolution is obtained as a pooled value. In a CNN, the convolutional layers are not fully connected. The only fully connected layer is the final layer responsible for the classification task[108]. The CNN framework has been shown to be extremely successful in problems such as image classification, disease state determination, pattern recognition and self-driving[109]. The CNN framework has limitations. These include training with limited data, difficult for real-applications and very powerful tools are required to deploy the models in practice[109].

Recurrent Neural Networks have emerged as a powerful tool for on the fly updating and are powerful for sequential operations such as projecting flight path and in natural language. The

RNN is a special kind of a neural network that permits continuing information related to past knowledge by utilizing a special kind of looped architecture. To understand an RNN we can divide the network by the flow of information. In each round a new instance of our model is created. Each instance is divided into discrete time steps, that contain recurrent layers. An element of a sequence is fed into the model in each step[110]. A feedforward connection links information flow from one neuron to another. The data contained in this link is the calculated neuronal activation which, is passed forward to the next step[110]. In contrast, a recurrent connection constitutes neuronal activation data flowing from the preceding time step[110]. In both cases a neuron activation in a recurrent network is not unidirectional and only reflects the network instance's current state. The optimal values for the initial parameter are defined similarly to the optimal values for every connection's weight during the training process[110]. The recurrent layers are modified and cast as a feed forward layer, a new gradient dependent only on the modified recurrent layer is derived and all the methods of backpropagation used for a feedforward network are then applied for RNN training. Error is modified such that the adjustment of the weights is not a direct change but is dependent on the error derivatives computed after each batch of training examples[110]. The sum of all the derivatives is averaged over all the links that belong to the same set. The average error derivative assumes that all the dynamics that act on a connection's weight will be captured by averaging the entire until the network converges[110]. Applications of RNNs in bioinformatics have been primarily focused in the space of proteomics[111-114]. This is likely because of the sequence of peptides fits well into the RNN learning.

Interestingly, RNNs and CNNs are now converging in structure with recurrent convolutional neural networks and their applications are now dominating the computer vision space[108]. A specific task in computer vision is segmentation, which is the process of teaching a

computer to recognize targets pictured within an image and discriminating them from the background[108]. Segmentation of an image is made up of with two steps: classification of the image subject and locating the target within an image. There are two types of segmentation: semantic segmentation or instance segmentation. Semantic segmentation attempts to classify every pixel in the image[115]. Semantic segmentation is dependent on careful labeling of target value boundaries and the accompanying network architecture[115]. Strong success in semantic segmentation has been archived with the U-NET architecture[115].

The U-Net architecture is made of two halves for image processing. The front half is the encoder which is used to capture the context in the image. The encoder is a traditional stack of convolutional and max pooling layers. The second half is symmetric to the first expanding as the decoder for object localization using a series of transposed convolutions. End-to-end the U-like structure is a fully convolutional network (FCN), only containing convolutional layers with no dense layers, as such it returns an image and not a single value classification from the final layer. The returned image is a two channel or multi-channel representation of each of the pixels. The classification of the pixels is inherent in that the colors produced are mapped to the indexed class[115].

Instance segmentation is the current cutting edge of computer vision. The advancement of instance segmentation beyond semantic segmentation in addition to pixel level classification, the models identify where each instance in an image is located for each class separately[116,117]. Thus, there are three steps in the process that lend themselves to hybrid recurrent convolutional neural network[116,117]. The series of information contained in the image is updated using the back propagation[116,117]. New models are generated for each sequence generating strong simultaneous classification of objects in an image[116,117].

A recent adaptation of a recurrent convolutional neural network was the mask-rcnn[116]. The software was built on top of three separate backbones; ResNet-101-C4, ResNet-101-FPN, ResNeXt-101-FPN and outperformed MNC, FCIS, and FCIS$^+$ on all three backgrounds[106,118,119]. The model achieved a level best of AP_{50} score of 54.9,49.5 and 60 in the respective background beating the aforementioned architectures on all backgrounds and in all benchmarks on the COCO dataset[118].

1.10 Thesis outline

The overall goal of this dissertation is to design, create and deploy machine learning architectures in the fields of bioinformatics and life sciences. In this section I gave a brief overview of how we designed a hybrid feature selection algorithm to reduce the computational complexity of deploying machine learning models in the fields of transcriptomics and cancer biology. We extended out feature selection algorithm to preprocess and select features from epigenomics, microarray and genomics data as a general-purpose omics feature selection software. Finally, we explored areas of deep learning in computer vision. We designed and implemented a high throughput instance segmentation algorithm aimed at segmenting fungal structures in microscope images for sorghum roots. Finally, this architecture was deployed on aws as a free service to benefit fungal biology community.

In Chapter 2, we data mined over ten thousand tumor transcriptomes from the TCGA database and programmatically curated the clinicopathologic annotations to characterize the metastatic patterns of primary tumors arising in thirty-two anatomic locations. The database revealed 125 distinct metastatic loci with clear organotropic metastases in 16 anatomic locations in 12 cancers. We built a tree-based classifier to predict tumor type and site-specific metastatic loci from individual primary tumors. Our feature selection algorithm produced latent feature sets

that showed significant enrichment for hallmark cancer metastasis pathways. This phenomenon was consistently reproducible and was demonstrated to be persistent across tumor types arising in separate locations that metastasized to concordant distant organs.

In Chapter 3, we demonstrate our feature selection algorithm goes beyond selection of features in transcriptomic profiling data and outfitted the software to analyze data from Illumina legacy 450k array data, trained a neural network wrapper model and successfully classified the tumor type of over ten thousand primary tumors demonstrating our feature selection architecture successfully removes redundant features in problems with an extreme Big-P to Little-N ratio. We extended the functionality beyond continuous variables and to accommodate discrete variables in genomics problems such as variants. The software is outfitted with over thirty functions for data preprocessing, genomic window manipulation, variant selection. It is now completely multi-threaded. The software is published as publicly available on the python package index.

In Chapter 4, we collaborated with fungal biologists to implement a first in class instance segmentation suite for fungal structures. The algorithm segments five classes of fungal structures and two classes of root structures. The trained model was integrated into the Amazon webservices platform, and a web-based interface allows users to upload microscope images onto aws for on-the-fly instance segmentation. Finally, we have implemented the model on cloud front as with access at the command line (cli) interface for high throughput usage of our internal teams. We plant to integrate the cli with the high throughput microscopy team to produce an immediate end-to-end sample to analysis workflow.

1.11 Bibliography:

1 Moor, J. The Dartmouth College Artificial Intelligence Conference: The next fifty years. *Ai Magazine* **27**, 87-91 (2006).

2 Moore, G. E. Cramming more components onto integrated circuits (Reprinted from Electronics, pg 114-117, April 19, 1965). *Proceedings of the Ieee* **86**, 82-85, doi:Doi 10.1109/Jproc.1998.658762 (1998).

3 Marr, B. *How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read*, <https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/?sh=63b2a6a860ba> (May 21, 2018).

4 Stephens, Z. D. *et al.* Big Data: Astronomical or Genomical? *PLoS Biol* **13**, e1002195, doi:10.1371/journal.pbio.1002195 (2015).

5 Goecks, J., Jalili, V., Heiser, L. M. & Gray, J. W. How Machine Learning Will Transform Biomedicine. *Cell* **181**, 92-101, doi:10.1016/j.cell.2020.03.022 (2020).

6 Cancer Genome Atlas Research, N. *et al.* The Cancer Genome Atlas Pan-Cancer analysis project. *Nat Genet* **45**, 1113-1120, doi:10.1038/ng.2764 (2013).

7 Consortium, G. T. The GTEx Consortium atlas of genetic regulatory effects across human tissues. *Science* **369**, 1318-1330, doi:10.1126/science.aaz1776 (2020).

8 Consortium, E. P. The ENCODE (ENCyclopedia Of DNA Elements) Project. *Science* **306**, 636-640, doi:10.1126/science.1105136 (2004).

9 Navarro Gonzalez, J. *et al.* The UCSC Genome Browser database: 2021 update. *Nucleic Acids Res* **49**, D1046-D1057, doi:10.1093/nar/gkaa1070 (2021).

10 Hastie, T., Tibshirani, R. & Friedman, J. H. *The elements of statistical learning : data mining, inference, and prediction*. 2nd edn, (Springer, 2009).

11 Zhao, Y., Wong, L. & Goh, W. W. B. How to do quantile normalization correctly for gene expression data analyses. *Sci Rep* **10**, 15534, doi:10.1038/s41598-020-72664-6 (2020).

12 (2015).

13 Ewels, P., Magnusson, M., Lundin, S. & Kaller, M. MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32**, 3047-3048, doi:10.1093/bioinformatics/btw354 (2016).

14 Albrecht, S., Sprang, M., Andrade-Navarro, M. A. & Fontaine, J. F. seqQscorer: automated quality control of next-generation sequencing data using machine learning. *Genome Biol* **22**, 75, doi:10.1186/s13059-021-02294-2 (2021).

15 Institute, N. H. G. R. *Data Tools and Resources*, <https://www.genome.gov/research-at-nhgri/Data-Tools-and-Resources> (2022).

16 Waterston, R. H., Lander, E. S. & Sulston, J. On the sequencing of the human genome. *Proceedings of the National Academy of Sciences of the United States of America* **99**, 3712 - 3716 (2002).

17 Propositiones, J. H., David Singmaster.

18 Hutter, M. *The hutter prize*, <http://prize.hutter1.net> (2022).

19 M. Burrows , D. J. W. A block-sorting lossless data compression algorithm. *SRC Research Report* (1994).

20 M. Bocicor, G. C. a. I. C. A Reinforcement Learning Approach for Solving the Fragment Assembly Problem. *13th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, doi:10.1109/SYNASC.2011.9. (2011).

21 Luo, J. *et al.* A comprehensive review of scaffolding methods in genome assembly. *Brief Bioinform* **22**, doi:10.1093/bib/bbab033 (2021).

22 Wang, H., Cimen, E., Singh, N. & Buckler, E. Deep learning for plant genomics and crop improvement. *Curr Opin Plant Biol* **54**, 34-41, doi:10.1016/j.pbi.2019.12.010 (2020).

23 Baudry, L. *et al.* instaGRAAL: chromosome-level quality scaffolding of genomes using a proximity ligation-based scaffolder. *Genome Biol* **21**, 148, doi:10.1186/s13059-020-02041-z (2020).

24 Ghurye, J. *et al.* Integrating Hi-C links with assembly graphs for chromosome-scale assembly. *PLoS Comput Biol* **15**, e1007273, doi:10.1371/journal.pcbi.1007273 (2019).

25 Dudchenko, O. *et al.* De novo assembly of the Aedes aegypti genome using Hi-C yields chromosome-length scaffolds. *Science* **356**, 92-95, doi:10.1126/science.aal3327 (2017).

26 Marie-Nelly, H. *et al.* High-quality genome (re)assembly using chromosomal contact data. *Nat Commun* **5**, 5695, doi:10.1038/ncomms6695 (2014).

27 Burton, J. N. *et al.* Chromosome-scale scaffolding of de novo genome assemblies based on chromatin interactions. *Nat Biotechnol* **31**, 1119-1125, doi:10.1038/nbt.2727 (2013).

28 Kaplan, N. & Dekker, J. High-throughput genome scaffolding from in vivo DNA interaction frequency. *Nat Biotechnol* **31**, 1143-1147, doi:10.1038/nbt.2768 (2013).

29 Leinonen, M. & Salmela, L. Optical map guided genome assembly. *BMC Bioinformatics* **21**, 285, doi:10.1186/s12859-020-03623-1 (2020).

30 Jarvis, D. E. *et al.* The genome of Chenopodium quinoa. *Nature* **542**, 307-312, doi:10.1038/nature21370 (2017).

31 Gautam, S., Mishra, U., Scown, C. D. & Zhang, Y. Sorghum biomass production in the continental United States and its potential impacts on soil organic carbon and nitrous oxide emissions. *Global Change Biology Bioenergy* **12**, 878-890, doi:10.1111/gcbb.12736 (2020).

32 Hu, Z., Olatoye, M. O., Marla, S. & Morris, G. P. An Integrated Genotyping-by-Sequencing Polymorphism Map for Over 10,000 Sorghum Genotypes. *Plant Genome* **12**, doi:10.3835/plantgenome2018.06.0044 (2019).

33 Cooper, E. A. *et al.* A new reference genome for Sorghum bicolor reveals high levels of sequence similarity between sweet and grain genotypes: implications for the genetics of sugar metabolism. *BMC Genomics* **20**, 420, doi:10.1186/s12864-019-5734-x (2019).

34 Paterson, A. H. *et al.* The Sorghum bicolor genome and the diversification of grasses. *Nature* **457**, 551-556, doi:10.1038/nature07723 (2009).

35 NCBI. (2017).

36 Prjibelski, A., Antipov, D., Meleshko, D., Lapidus, A. & Korobeynikov, A. Using SPAdes De Novo Assembler. *Curr Protoc Bioinformatics* **70**, e102, doi:10.1002/cpbi.102 (2020).

37 Bankevich, A. *et al.* SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J Comput Biol* **19**, 455-477, doi:10.1089/cmb.2012.0021 (2012).

38 Kleber Padovani, R. X., Andre Carvalho, Anna Reali, Annie Chateau, Ronnie Alves. A step towards a reinforcement learning de novo genome assembler. *Quantitative Biology* (2021).

39 Padovani de Souza, K. *et al.* Machine learning meets genome assembly. *Brief Bioinform* **20**, 2116-2129, doi:10.1093/bib/bby072 (2019).

40 Zimin, A. V. *et al.* The MaSuRCA genome assembler. *Bioinformatics* **29**, 2669-2677, doi:10.1093/bioinformatics/btt476 (2013).

41 Maccallum, I. *et al.* ALLPATHS 2: small genomes assembled accurately and with high continuity from short paired reads. *Genome Biol* **10**, R103, doi:10.1186/gb-2009-10-10-r103 (2009).

42 Butler, J. *et al.* ALLPATHS: de novo assembly of whole-genome shotgun microreads. *Genome Res* **18**, 810-820, doi:10.1101/gr.7337908 (2008).

43 Gurevich, A., Saveliev, V., Vyahhi, N. & Tesler, G. QUAST: quality assessment tool for genome assemblies. *Bioinformatics* **29**, 1072-1075, doi:10.1093/bioinformatics/btt086 (2013).

44 Salzberg, S. L. *et al.* GAGE: A critical evaluation of genome assemblies and assembly algorithms. *Genome Res* **22**, 557-567, doi:10.1101/gr.131383.111 (2012).

45 Boisvert, S., Raymond, F., Godzaridis, E., Laviolette, F. & Corbeil, J. Ray Meta: scalable de novo metagenome assembly and profiling. *Genome Biol* **13**, R122, doi:10.1186/gb-2012-13-12-r122 (2012).

46 Peng, Y., Leung, H. C., Yiu, S. M. & Chin, F. Y. IDBA-UD: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics* **28**, 1420-1428, doi:10.1093/bioinformatics/bts174 (2012).

47 Nurk, S., Meleshko, D., Korobeynikov, A. & Pevzner, P. A. metaSPAdes: a new versatile metagenomic assembler. *Genome Res* **27**, 824-834, doi:10.1101/gr.213959.116 (2017).

48 Namiki, T., Hachiya, T., Tanaka, H. & Sakakibara, Y. MetaVelvet: an extension of Velvet assembler to de novo metagenome assembly from short sequence reads. *Nucleic Acids Res* **40**, e155, doi:10.1093/nar/gks678 (2012).

49 Miller, X. H. a. W. A Time-Efficient, Linear-Space Local Similarity Algorithm. *Advances in Applied Mathematics* **12** (1991).

50 Constantinescu, R. A machine learning approach to DNA shotgun sequence assembly. *Journal of Computer Science* (2016).

51 Smith, T. F. & Waterman, M. S. Identification of common molecular subsequences. *J Mol Biol* **147**, 195-197, doi:10.1016/0022-2836(81)90087-5 (1981).

52 Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**, 1754-1760, doi:10.1093/bioinformatics/btp324 (2009).

53 Needleman, S. B. & Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol* **48**, 443-453, doi:10.1016/0022-2836(70)90057-4 (1970).

54 Langmead, B., Trapnell, C., Pop, M. & Salzberg, S. L. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* **10**, R25, doi:10.1186/gb-2009-10-3-r25 (2009).

55 McCreight, E. M. A Space-Economical Suffix Tree Construction Algorithm. *Journal of the ACM* **23**, 262–272, doi:10.1145/321941.321946 (1976).

56 Mantaci, S., Restivo, A., Rosone, G. & Sciortino, M. An extension of the Burrows-Wheeler transform. *Theoretical Computer Science* **387**, 298-312, doi:10.1016/j.tcs.2007.07.014 (2007).

57 Gagie, T., Manzini, G. & Siren, J. Wheeler graphs: A framework for BWT-based data structures. *Theor Comput Sci* **698**, 67-78, doi:10.1016/j.tcs.2017.06.016 (2017).

58 Rossi, M. *et al.* Finding Maximal Exact Matches Using the r-Index. *J Comput Biol*, doi:10.1089/cmb.2021.0445 (2022).

59 Rossi, M., Oliva, M., Langmead, B., Gagie, T. & Boucher, C. MONI: A Pangenomic Index for Finding Maximal Exact Matches. *J Comput Biol*, doi:10.1089/cmb.2021.0290 (2022).

60 Alanko, J., Alipanahi, B., Settle, J., Boucher, C. & Gagie, T. Buffering updates enables efficient dynamic de Bruijn graphs. *Comput Struct Biotechnol J* **19**, 4067-4078, doi:10.1016/j.csbj.2021.06.047 (2021).

61 Kuhnle, A. *et al.* Efficient Construction of a Complete Index for Pan-Genomics Read Alignment. *J Comput Biol* **27**, 500-513, doi:10.1089/cmb.2019.0309 (2020).

62 Mun, T. *et al.* Matching Reads to Many Genomes with the r-Index. *J Comput Biol* **27**, 514-518, doi:10.1089/cmb.2019.0316 (2020).

63 Boucher, C. *et al.* Prefix-free parsing for building big BWTs. *Algorithms Mol Biol* **14**, 13, doi:10.1186/s13015-019-0148-5 (2019).

64 DePristo, M. A. *et al.* A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat Genet* **43**, 491-498, doi:10.1038/ng.806 (2011).

65 Hanahan, D. & Weinberg, R. A. Hallmarks of cancer: the next generation. *Cell* **144**, 646-674, doi:10.1016/j.cell.2011.02.013 (2011).

66 Vogelstein, B. *et al.* Cancer genome landscapes. *Science* **339**, 1546-1558, doi:10.1126/science.1235122 (2013).

67 Genomes Project, C. *et al.* An integrated map of genetic variation from 1,092 human genomes. *Nature* **491**, 56-65, doi:10.1038/nature11632 (2012).

68 McKenna, A. *et al.* The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res* **20**, 1297-1303, doi:10.1101/gr.107524.110 (2010).

69 Bolger, A. M., Lohse, M. & Usadel, B. Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics* **30**, 2114-2120, doi:10.1093/bioinformatics/btu170 (2014).

70 Garrison, E. a. M., Gabor. Haplotype-based variant detection from short-read sequencing. *Quantitative Biology - Quantitative Methods* (2012).

71 Albers, C. A. *et al.* Dindel: accurate indel calls from short-read data. *Genome Res* **21**, 961-973, doi:10.1101/gr.112326.110 (2011).

72 Marth, G. T. *et al.* A general approach to single-nucleotide polymorphism discovery. *Nat Genet* **23**, 452-456, doi:10.1038/70570 (1999).

73 Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**, 2987-2993, doi:10.1093/bioinformatics/btr509 (2011).

74 Narasimhan, V. *et al.* BCFtools/RoH: a hidden Markov model approach for detecting autozygosity from next-generation sequencing data. *Bioinformatics* **32**, 1749-1751, doi:10.1093/bioinformatics/btw044 (2016).

75 Danecek, P. & McCarthy, S. A. BCFtools/csq: haplotype-aware variant consequences. *Bioinformatics* **33**, 2037-2039, doi:10.1093/bioinformatics/btx100 (2017).

76 Poplin, R. *et al.* A universal SNP and small-indel variant caller using deep neural networks. *Nat Biotechnol* **36**, 983-987, doi:10.1038/nbt.4235 (2018).

77 Luo, R., Sedlazeck, F. J., Lam, T. W. & Schatz, M. C. A multi-task convolutional deep neural network for variant calling in single molecule sequencing. *Nat Commun* **10**, 998, doi:10.1038/s41467-019-09025-z (2019).

78 Yu, L. & Liu, H. Efficient Feature Selection via Analysis of Relevance and Redundancy. *J. Mach. Learn. Res.* **5**, 1205-1224 (2004).

79 Hua, J., Liu, H., Zhang, B. & Jin, S. LAK: Lasso and K-Means Based Single-Cell RNA-Seq Data Clustering Analysis. *IEEE Access* **8**, 129679-129688, doi:10.1109/access.2020.3008681 (2020).

80 Ding, C. & Peng, H. Minimum redundancy feature selection from microarray gene expression data. *J Bioinform Comput Biol* **3**, 185-205, doi:10.1142/s0219720005001004 (2005).

81 Radovic, M., Ghalwash, M., Filipovic, N. & Obradovic, Z. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinformatics* **18**, 9, doi:10.1186/s12859-016-1423-9 (2017).

82 Ma, S. & Kosorok, M. R. Identification of differential gene pathways with principal component analysis. *Bioinformatics* **25**, 882-889, doi:10.1093/bioinformatics/btp085 (2009).

83 Perez-Riverol, Y., Kuhn, M., Vizcaino, J. A., Hitz, M. P. & Audain, E. Accurate and fast feature selection workflow for high-dimensional omics data. *PLoS One* **12**, e0189875, doi:10.1371/journal.pone.0189875 (2017).

84 Liu, J. X., Xu, Y., Zheng, C. H., Wang, Y. & Yang, J. Y. Characteristic gene selection via weighting principal components by singular values. *PLoS One* **7**, e38873, doi:10.1371/journal.pone.0038873 (2012).

85 Kobak, D. & Berens, P. The art of using t-SNE for single-cell transcriptomics. *Nat Commun* **10**, 5416, doi:10.1038/s41467-019-13056-x (2019).

86 McInnes, L. & Healy, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv* **abs/1802.03426** (2018).

87 Becht, E. *et al.* Dimensionality reduction for visualizing single-cell data using UMAP. *Nat Biotechnol*, doi:10.1038/nbt.4314 (2018).

88 Pournara, I. & Wernisch, L. Factor analysis for gene regulatory networks and transcription factor activity profiles. *BMC Bioinformatics* **8**, 61, doi:10.1186/1471-2105-8-61 (2007).

89 Bonilla Huerta, E., Duval, B. & Hao, J.-K. 250-261 (Springer Berlin Heidelberg).

90 Mirzal, A. 223-230 (Springer Singapore).

91 Liu, S. *et al.* Feature selection of gene expression data for Cancer classification using double RBF-kernels. *BMC Bioinformatics* **19**, 396, doi:10.1186/s12859-018-2400-2 (2018).

92 Mohapatra, P., Chakravarty, S. & Dash, P. K. Microarray medical data classification using kernel ridge regression and modified cat swarm optimization based gene selection system. *Swarm and Evolutionary Computation* **28**, 144-160, doi:10.1016/j.swevo.2016.02.002 (2016).

93 Hu, J. *et al.* DGLinker: flexible knowledge-graph prediction of disease-gene associations. *Nucleic Acids Res* **49**, W153-W161, doi:10.1093/nar/gkab449 (2021).

94 Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol* **15**, 550, doi:10.1186/s13059-014-0550-8 (2014).

95 Mundra, P. A. & Rajapakse, J. C. Gene and sample selection using T-score with sample selection. *Journal of Biomedical Informatics* **59**, 31-41, doi:10.1016/j.jbi.2015.11.003 (2016).

96 Lu, S. *et al.* Stepwise selection on homeologous PRR genes controlling flowering and maturity during soybean domestication. *Nat Genet* **52**, 428-436, doi:10.1038/s41588-020-0604-7 (2020).

97 Knuppel, S. *et al.* Multi-locus stepwise regression: a haplotype-based algorithm for finding genetic associations applied to atopic dermatitis. *BMC Med Genet* **13**, 8, doi:10.1186/1471-2350-13-8 (2012).

98 Huh, I., Kwon, M. S. & Park, T. An Efficient Stepwise Statistical Test to Identify Multiple Linked Human Genetic Variants Associated with Specific Phenotypic Traits. *PLoS One* **10**, e0138700, doi:10.1371/journal.pone.0138700 (2015).

99 Barron, J. T. A General and Adaptive Robust Loss Function. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4326-4334 (2019).

100 Klein, A. in *Encyclopedia of Machine Learning and Data Mining.*

101 Matthews, B. W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim Biophys Acta* **405**, 442-451, doi:10.1016/0005-2795(75)90109-9 (1975).

102 MacQueen, J.

103 Doucette, A. 35-40 (Association for Computational Linguistics).

104 Libbrecht, M. W. & Noble, W. S. Machine learning applications in genetics and genomics. *Nat Rev Genet* **16**, 321-332, doi:10.1038/nrg3920 (2015).

105 Ozawa, S. & Abe, S.

106 He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. arXiv:1512.03385 (2015). <https://ui.adsabs.harvard.edu/abs/2015arXiv151203385H>.

107 Pan, S. J. & Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* **22**, 1345-1359 (2010).

108 Dumoulin, V. & Visin, F. A guide to convolution arithmetic for deep learning. *ArXiv* **abs/1603.07285** (2016).

109 Chai, J., Zeng, H., Li, A. & Ngai, E. W. T.

110 Buduma, N. & Locascio, N.

111 Khanh Le, N. Q., Nguyen, Q. H., Chen, X., Rahardja, S. & Nguyen, B. P. Classification of adaptor proteins using recurrent neural networks and PSSM profiles. *BMC Genomics* **20** (2019).

112 Gori, M., Hammer, B., Hitzler, P. & Palm, G. n. Perspectives and challenges for recurrent neural network training. *Log. J. IGPL* **18**, 617-619 (2010).

113 Ma, C. *et al.* DeepRT: deep learning for peptide retention time prediction in proteomics. *arXiv: Quantitative Methods* (2017).

114 Zohora, F. T. *et al.* DeepIso: A Deep Learning Model for Peptide Feature Detection from LC-MS map. *Scientific Reports* **9** (2019).

115 Ronneberger, O., Fischer, P. & Brox, T. in *MICCAI.*

116 He, K., Gkioxari, G., Doll·r, P. & Girshick, R. B. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **42**, 386-397 (2020).

117 Ren, S., He, K., Girshick, R. B. & Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **39**, 1137-1149 (2015).

118 Dai, J., He, K. & Sun, J. Instance-Aware Semantic Segmentation via Multi-task Network Cascades. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3150-3158 (2016).

119 Li, Y., Qi, H., Dai, J., Ji, X. & Wei, Y. Fully Convolutional Instance-Aware Semantic Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4438-4446 (2017).

CHAPTER 2

ARE WE THERE YET? A MACHINE LEARNING ARCHITECTURE TO PREDICT ORGANOTROPIC METASTASES.

[1] Michael Skaro, Marcus Hill, Shannon Quinn, Melissa B. Davis, Andrea Sboner, Mandi Murph, Jonathan Arnold PhD. Accepted. November 24, 2021. BMC Medical Genomics. Reprinted here with the permission of the publisher.

2.1 Abstract:

Background & Aims: Cancer metastasis into distant organs is an evolutionarily selective process. A better understanding of the driving forces endowing proliferative plasticity of tumor seeds in distant soils is required to develop and adapt better treatment systems for this lethal stage of the disease. To this end, we aimed to utilize transcript expression profiling features to predict the site-specific metastases of primary tumors and second, to identify the determinants of tissue specific progression. Methods: We used statistical machine learning for transcript feature selection to optimize classification and built tree-based classifiers to predict tissue specific sites of metastatic progression. Results: We developed a novel machine learning architecture that analyzes 33 types of RNA transcriptome profiles from The Cancer Genome Atlas (TCGA) database. Our classifier identifies the tumor type, derives synthetic instances of primary tumors metastasizing to distant organs and classifies the site-specific metastases in 16 types of cancers metastasizing to 12 locations. Conclusions: We have demonstrated that site specific metastatic progression is predictable using transcriptomic profiling data from primary tumors and that the overrepresented biological processes in tumors metastasizing to congruent distant loci are highly overlapping. These results indicate site-specific progression was organotropic and core features of biological signaling pathways are identifiable that may describe proliferative plasticity in distant soils.

2.2 Introduction:

Metastasis accounts for 90% of cancer associated mortality[1]. While disease spread is a definitive turning point in patient pathology, metastasis is a long, arduous, and inefficient process for a primary tumor[1,2]. To establish an overt colonization in a distant organ, metastasis proceeds through multiple restrictive bottlenecks. Tumor sheds must first retain membrane integrity during a violent intravasation and successfully navigate the circulatory vasculature. Arriving in the new settlement, cells must elude immune response, retain activation of growth signals, and survive radiotherapies or putative ablation via chemotherapeutics[3-5]. The possible organs sites of metastasis are tumor type specific; and in part determined by primary lesion anatomic location, intratumor metabolic reprogramming, augmented protein functions and disrupted biological pathways driving tumor cell fitness in the distant organs[6-10]. The dissemination of successful metastases is an organized process known as metastatic organotropism.

Metastatic organotropism is a long-standing problem in cancer research and characterizing the metastatic patterns of primary tumors is a critical step towards treating patients with advanced disease[11,12]. Experimentally driven investigations have focused on characterizing the biological underpinnings of organotropic metastasis while computational approaches have developed tools attempting to predict the sites of metastases. Previous research has described the patterns of bone, liver, and lung tropisms. Bone tropisms arise primarily from breast and prostate cancers[13]. In prostate cancers, three major clusters of pathologies have evolved, one of which show high androgen receptor signaling and high bone-tropism compared to the other clusters[14,15]. Liver tropisms primarily arise from breast, lung, and gastrointestinal cancers[13]. A 17-gene signature has been shown to indicate adverse outcomes for breast cancer patients and has some correlative evidence suggesting liver progression from breast tumors[16]. Lung tropisms are observed most

commonly in breast, melanoma and thyroid cancers[13,17]. Similar to liver tumors, a 54 gene panel expression signature has been developed for showing correlation for organotropic metastasis from breast tumors progressing to the lung[18].

Studies using molecular information for retrospective analyses of tumor metastatic sites have been xenograft selection studies that extrapolated organotropic features from metastasis microarray data. Studies leveraging RNA transcript profiling data have been designed for single tumor type progressing to a single site. We have found no significant study has been developed on classifying site-specific metastasis from human primary tumor transcriptomic profiling data[5,19-28]. The most recent work investigating organotropic progression used no molecular data and instead used deep data mining of patient clinical data to model temporal patterns of tumor type site-specific progression and established a powerful co-occurrence based network but did not extract any biological determinants of tumor plasticity in distant organs[24].

Despite the significant progress made from previous modeling methods, a unified approach to predict site specific metastasis in multiple cancer types that learns the biological determinants of dissemination has not been resolved. We have leveraged the publicly available omics data and clinical annotations in the TCGA database to investigate metastatic organotropisms of multiple cancers. In this study, we build off the previous work and establish a machine learning architecture that models organotropic metastases by distinguishing the tumor type and in multiple cancer types predicts the loci of distant tumor metastases. We detail a migration from the canonical pipelines using differential expression for feature assessment and use statistical machine learning for feature selection to optimize classification. Our model systematically predicts site-specific metastases of primary tumors and our methods captured conserved core biological processes overrepresented in tumors of varying origin that seeded in concordant anatomic locations.

2.3 Methods:

Review of data download of TCGA transcriptomic and clinical annotation data:

The TCGA data portal has the clinical data commons that are publicly available for data mining in the clinical databank[29]. These data are accessible in multiple ways including Bulk/Batch API access, TCGA Biolinks software via Bioconductor, and Cart-Building on the portal website in a patient-by-patient search[29]. Currently, no unified patient disease progression information is directly available for bulk data mining on the portal website. Our progression annotation was built by text mining clinical files of progression annotations project by project using the batch query function in the TCGA Biolinks package. Each patient has multiple unique identifiers. In a project-by-project manner, each Case ID was cataloged. Each case ID query produced a case UUID that was used across the data types including the gene expression counts, VCF files, FASTQ files, images from slides, and clinical annotation for each experiment for each patient. Each UUID produces a patient summary. Each summary was broken down into: Data category, Experimental strategy, clinical annotations, and clinical supplemental files. The transcriptome counts files for each project were downloaded, normalized and analyzed. Each project has between 53 and 261 clinical annotation columns. The stringr and dplyr software packages were used for clinical annotation, data cleaning, and anatomical annotation[30]. Metastatic tumors identified in the clinical annotation file were drawn from the "metastatic tissue", "sites of metastases" or "metastatic tissue site" column(s). Tumor progression labeled as "synchronous" were not included in the metastatic data as the clinical timeline of diagnosis was ambiguous. The diagnosis allows for tumors to be classified as synchronous ranging between the time of diagnosis up to 6 months following the diagnosis in varying tumor types.

Review of synthetic sample generation:

Synthetic samples were generated to balance positive and negative classes using the SMOTE algorithm; where positive classes were tumors that developed a metastasis in the tested location and negative classes were tumors that did not develop a metastasis in the tested locaiton[31]. Briefly, the Synthetic Minority Oversampling Technique (SMOTE) is an algorithm to increase the representation of a minority class in machine learning classification problems. The objective function for this approach sits on top of a distance based KNN algorithm. The synthetic oversampling technique begins by selecting a minority class instance. Then finds the instance's k nearest neighbors. One of the minority class neighbors is chosen at random. A line is drawn between these two instances and a synthetic sample is generated along the line as a convex combination of the two real instances. This process repeats until it has created the desired number of synthetic samples. The number of synthetic samples generated was specific for each binary comparison. The authors suggest that the SMOTE algorithm can be used to generate a large sum of representative synthetic samples, however how large that sum is without over fitting the model is unknown. We employed an overfit prevention method during sample balancing. We measured 80% of the majority class and increased the representation of the minority to the match approximately 80% of the majority class rounded to the closest integer.

Review of Feature Selection:

Feature selection is a method in model building to reduce the dimensionality of a dataset. Overfitting can occur when the number of columns (features) outnumber the rows (instances) we can use for the model. To reduce the dimensionality of the problem we have employed three kinds of feature selection methods: Filter based, Wrapper-based and Embedded feature selection. Chi-square filtering calculates the chi-square metric between the target and the numerical variable and only reduces the features for the variables with the maximum chi-squared values. The SelectKBest,

Chi2 and MinMaxScaler Libraries from Sklearn and feature_selection module were used[32]. A Recursive feature selection estimator iteratively reduced the dimensionality of the data set by recursively considering smaller and smaller subsets of each feature block. The RFE was trained on each initial block of features and the importance of each feature was obtained through the feature_importances_ attribute. The RFE and LogisticRegression libraries from Sklearn and feature_selection module were used[32]. For embedded methods, Random forest classifier, random forest regression and lasso regression with a logistic regression estimator and L1 penalty were employed. These algorithms have an embedded feature selection method to stratify and rank features. The SelectFromModel, RfC and RfR libraries were imported from Sklearn[32,33]. We cross validated these approaches by extracting support values in each using the get_support() methods, summing the true feature support Booleans for each feature in each block across all five methods and sorting features by selection support.

Iterative Feature selection was conducted by splitting the 60,483 transcript features into 100 blocks of approximately 600 features to be assessed by the above algorithms. We extracted support values for each feature from each selection method. Each block was assessed independent of all other blocks in each classification. Transcripts were filtered for features that showed the highest cross-validated support in multiple or all algorithms. Dimensionality was finally reduced by filtering out co-linear features. The top 10% of highest scoring features were kept from each block for a total number of approximately 5000 candidate transcripts (50 transcripts x 100 blocks). The remaining transcripts were used as the input features in each binary classification. Tree-based models were selected as the best fit for the classification to account for the variability in number selected features in each classification and to allow model attributions to be extracted post-hoc.

Review of Model Building:

Random Forest classification and Gradient boosted tree classifiers were built to classify site specific progression from primary tumors. The selected features in each binary classification were used as input attributes into model classification. The model is set to report rounded value for classification but is capable of posterior probability for class likelihood. The code and the pretrained models are available through the documented Github. Model building and usage is documented on the Github wiki page.

Review of feature recapture:

Feature recapture was the final phase of model building and analysis. Testing the statistical significance of feature recapture in independently generated lists following bioinformatic analysis is an indirect however well documented technique to determine non-random enrichment[34]. Two sets of feature recapture were analyzed and displayed in Table S7. The tests were conducted; within cancer class seeding loci and the between cancer classes metastasizing in matching locations. The Fisher's exact was used to evaluate the significance of recapture between lists, as the significance of deviation from the null hypothesis can be directly calculated. Our null hypothesis was that the feature recapture when analyzing matched seeding locations across cancer types was by chance; therefore, no biological meaning can be drawn from the phenomena. Our alternative hypothesis was that recapture of features within class and between matching seeding locations indicates similar distant metastatic potential and offers candidate biomarkers for organotropic metastasis, respectively. The contingency table was set as; the background of the search space for the information gain algorithm. The starting feature selection space for each classification was the entire human transcriptome. As all of the binary compassions initially began considering all 60,483 transcripts, and each set of selected features were independently generated, the total transcriptome remained the background for all tests. In list A of each contingency table, we place the top 1000

features for each classification of primary tumor seeding location. In list B, we assess a second primary tumor type and/or metastatic location feature list. We test the significance of the intersection of the two lists considering the list sizes, background and overlap in contingency table. The GeneOverlap package on Bioconductor was used to conduct the Fisher's exact tests[35].

Gene set overrepresentation and Semantic analysis:

The clusterprofiler package was used to conduct an overrepresentation test in the GO database[36]. The selected features for each metastatic location in each cancer type were translated into their associated GO biological process IDs using the bitr function in the clusterProfiler package[36]. The overrepresented GO biological pathways were passed to into the GoSemSim package and simplify enrichment package[37]. A similarity matrix of biological functions was made using the simplfyEnrichment package in R[38]. A heatmap was produced by clustering the similarity scores of the biological functions using the package default binary cut function. A Fisher's exact test was conducted using the base GeneOverlap in R[35]. The background was changed from the human transcriptome to the GO database to account for the change in the search space[39]. The UpsetR package in R was used to display the bar graph of overlapping biological processes in the tumors seeding in matched locations[40]. All overlaps were tested between cancers metastasizing in similar organs.

Data availability and code:

We used public data sets drawn from the TCGA database using the GDC data commons for this project and its analyses[41]. We have provided all the custom computer code to produce these models.

Our code is currently available for view and use in a public Github repository: https://github.com/michaelSkaro/Classification_of_organotropic_metastases. The docker image containing all relevant environment variables, dependencies and a demo test data set is also made publicly available on docker hub and integrated into the Github actions. We have a documented wiki page that is available, demonstrating the installations, displays visualization and describes script usage within the pipeline. We have provided a general usage script that runs the entire metastatic classification pipeline. At the command line it can be ran using the metastasis_pipeline.py script within the built docker container. We have provided a general usage feature selection pipeline Feature_selection.py. We have provided the organotropic features sets for all cancer types selected in this study in the supplementary data tables. We have provided all enrichment and recapture code in the source code.

2.4 Results:

Classification of Tumor type:

Each tumor type is unique and potential metastatic sites of progression are limited based on the tumor gene expression profile, anatomic location, and blood circulation[24]. We hypothesized that each tumor type has subsets of features associated with tissue specific progression. Therefore, classifying tumor type was considered a critical step towards extracting patterns of organotropic metastasis. Thirty-three tumor types were considered by the model and are annotated by their four-letter code in the tumor type column in all figures and tables. Figure 1. displays the confusion matrix of the model as a heatmap and displays the model precision, recall and f1-score with normalized performance for population size classifying 33 cancer types in the TCGA database. Our model performs in the excellent range on thirty of the cancer classes, Cholangiocarcinoma (CHOL) showed the worst performance as the population of 45 was too small to develop a strong model for cancer type classification. Esophageal carcinoma and stomach adenocarcinoma showed some misclassification in between the types, given these tumors have been shown to be pathologically very similar in previous research this was unsurprising[43]. Colorectal adenocarcinoma (COAD) showed considerable misclassification specifically misidentifying COAD for Renal adenocarcinoma (READ) and vice-versa. The COAD and READ classes are combined in the UCSC genome browser database, and combined COAD and READ in further analyses as the metastatic progressions showed a considerable overlap.

Overall, the cancer type classification model performed in the excellent range with a macro average precision of 94.2, macro average recall of 91.98 and macro average F1 score of 92.77. The classified results were used to carry forward for site specific metastases prediction. The classification of the primary tumor type significantly decreased the complexity of predicting

possible sites of metastatic progression for each primary tumor. We annotated 125 metastatic locations in the ten thousand patient samples separated in twenty-three TCGA projects containing transcriptomic and clinical data (Figure 2). The most observed sites of metastasis were Bone, Liver, Lung and Lymph Node(Figure 2.). We filtered for metastatic sites with at least eight clinical annotations of progression for a given site and an overall total population of over fifty patients with documented non-synonymous progression of disease arising from the primary tumor. Following filtering we were able to analyze 35 tumor metastatic site pairs.

Classification of organotropic progression:

Thirty-three cancer types in TCGA were analyzed in this study, based on the availability of annotated metastatic progression in the TCGA clinical data. For sixteen cancer types, we predicted site specific organotropic metastases. The classification of the organotropic metastases in the sixteen cancer types occurred in three phases. First, synthetic sample generation, followed by feature selection, and finally classification of progression. Synthetic sample generation was used to increase the representation of tumors that metastasized to each of the tested locations. Feature selection was used to reduce the dimensionality of the data and to find transcripts that best separated the tumors that metastasized to a tested locations from negative cases. We combined five feature selection algorithms to assess feature value discriminating between positive and negative classes in each classification independent of all other comparisons[44].

In Figure 3. we show the performance of classification in sixteen cancer types. We report four metrics for the classification of site-specific progression in each cancer; precision, recall, F1 Measure and Model Accuracy. We observed an overall average precision of 0.82, average recall of 0.82, average F1 Measure of 0.82 and average accuracy of 0.82 considering all sites and all predictions. We performed in the excellent range on twenty six of 35 classification pairs. The

projects with the fewest errors were the larger projects; Bladder cancer, Breast cancer, Colorectal cancers, and lung cancers. Sites with the strongest model support for prediction were Bone, Liver, Lung and Lymph Node. Cancer type specific performance is detailed in Table 1. Considering all progressions for each cancer type.

After the classification of the organotropic metastases, we predicted tumors metastasizing to congruent loci may exhibit similar biological changes in the primary tumor endowing proliferative plasticity in the distant organ locations. To this end, we used the top 1000 selected features from each feature selection to conduct pathway enrichment. In Figure 4A. We simulated the number of expected biological processes to overlap if 1000 randomly selected transcripts were enriched in the GO database. It is known that Ensemble transcript IDs map to multiple GO biological process IDs and therefore there is a high probability of false discovery due to random chance. To establish that our observed overlap between lists of GO BP IDs were significant, we modified previously published gene overlap protocols and conducted a weighted simulation of our feature selection methods where IDs with the least amount of mapping match GO IDs are given priori over IDs with many matches[31]. The weighted simulation was conducted by randomly selecting two sets of 1000 transcript features, conducting a GO over representation test within each list, filtering for significantly overrepresented processes in the feature sets followed by testing the simulated overlap of the two independently generated GO:ID lists. We conducted this simulation a total of 750,000 times using 50,000 simulations for each possible intersection combination. We tested all pairwise combinations of 5 possible lengths of GO:ID lists ranging from 100 GO:IDs to 500 GO:IDs. The simulated results are stratified by the colored lines in Figure 4A. Our simulation shows that the feature selection method consistently produced significantly higher overlap than in random simulation. In Figures 4B-4E we show the number of overrepresented biological processes

in the tumors metastasizing to bone, liver, lung, and Lymph Node, respectively. We reported the list overlaps, odds ratio and adjusted p.value after Bonferroni adjustment in the supplementary data tables (S7).

In Figure 5A, 5B, 5C, and 5D we cluster the sematic similarity of the GO:ID terms that passed the selection and filtering. We display four heatmaps that describe the biological processes found to be overrepresented in primary tumors metastasizing to concordant locations. The largest cluster common among all the comparisons was regulation of morphogenesis and migration. This is a significant result as collective cell migration is a hallmark of metastatic cancer and further suggests a progressive tumors may be identified by the expression profiles [45].

2.5 Discussion:

The capacity to accurately determine the site-specific metastases of patients' primary tumors is directly applicable to clinical actions for patients. Following tumor resection; transcriptomic analysis of a patient's tumor can provide valuable insight into disease progression and can aid clinician's treatment interventions[46]. We present an accurate and precise machine learning architecture that can classify the tumor type and can identify if and where a primary tumor will metastasize. Embedded in our model we offer potential users the opportunity to report the locations of the metastases and additionally retain the posterior probabilities of metastatic progression to each location. This offers users the ability to integrate investigation specific calibration for their data and report the confidence of the classification in the clinical setting.

The model improves on previous work in two fundamental ways. The model increases the scope and performance comparison to previous work modeling either a single cancer type or single metastatic location and identifies biological feature determinants of organotropic metastasis from unified transcript profiling data. The model was shown to be broadly applicable in 16 different cancer types. Our feature selection method is uncommon amongst canonical bioinformatics or biomedical pipelines. The differentiation of the positive class feature space was only discernable from the negative class feature space following statistical machine learning centered feature selection methods. The features that are represented in the supplementary data tables were produced cross validating five feature selection method and extracting model attribution support for the best features in each comparison.

Our model is not without clear limitations. By breaking down a multi-label, multi-output experiment into NxM binary classification experiments we sacrificed detecting possible features that may be present in non-mutually exclusive progression. An example of this break down occurs

when one patient's tumor metastasized to the liver and the lung. The model will fail to find features that may be dictating the multi-organ expansion of the patient's disease. We justify this sacrifice with an opportunity cost. While we will not find these coalescent features as there are not enough coalescent cases to properly model these phenomena, we do produce a model with very high sensitivity and specificity to detect if and where both metastases will arise in a given case. Further, the model is built in a way, upon receipt of more data, we can make the necessary modifications from a binary comparisons list to an All vs. All classification. The transition to an All vs. All classification presents the clear second limitation of this model; the very costly overhead of data production. Our model relies on the largest ever unified conglomerate of tumor transcriptome data to produce the level of precision and recall we achieved on only 16 cancer types of the 33 TCGA projects we investigated. This model is reliant on the high-quality data production pipeline in TCGA. The transcript profiling data for each tumor were produced from sequencing of patient tumors of extremely high purity which is very uncommon in most studies. If this model is to be broadly incorporated into the medical community it will need a very deep and diverse set of transcriptomes to train on that is much larger than our current TCGA dataset.

Next Steps:

Our next steps will be to include more cancer types. As the publicly available data continue to grow as a super set of TCGA and the International Cancer Genome Consortium (ICGC), more projects will have clinically annotated tumor and normal transcriptomes. Further, the TCGA database documentation has become more unified and is continuously growing in its clarity. This will allow us to incorporate multiple data types into a multiomic approach that may illuminate genetic, genomic, epigenetic and transcriptomic features working to provide proliferative plasticity

in metastatic soils. Finally, if the public data grows by a significant margin, we can approach characterizing organotropic metastasis with an All vs. All model.

2.6 Conclusion:

Our machine learning architecture expands the understanding of the cancer metastasis. The leading cause of cancer associated death is metastatic progression of disease, however incorporating this tool into the clinical timelines for patients may offer clinicians opportunities for pre-metastatic therapeutic interventions. We demonstrate our model can detect if and where metastases will arise. Our methods of synthetic sample generation and feature selection produced a clear and concise biological data-based model of metastatic progression in multiple tumor types. Our recaptured features are offered as candidate biomarkers of site-specific metastatic organotropism.

Author's contributions:

MS, MM, AS and JA planned and designed study (Design). MS and YZ collected data from the GDC data commons API, annotated samples and documented clinicopathologic data in TCGA(data acquisition, data annotation and data management). MS, and MH conducted experiments and generated code/models(Analysis). JA, MM, SQ and MBD provided experimental guidance and support for model development/analysis and revised manuscript(Design and writing). Specifically: JA advised for statistical analysis, SQ advised on feature selection and machine learning support and analysis for MS and MH, MBD and MM provided experimental guidance on patterns in cancer metastasis, biological interpretation (interpretation of data). MS, MH, and JA analyzed data. MS, AS and JA wrote the manuscript.

All authors approved the final version of the manuscript.

Acknowledgements:
A major acknowledgement for the Georgia Advanced Computing Resource Center for computational support for this project.

We would like to Acknowledge the generous fellowship provided by the Grimes family.

2.7 Figures and Tables:

Figure 1. Classification of tumor type.

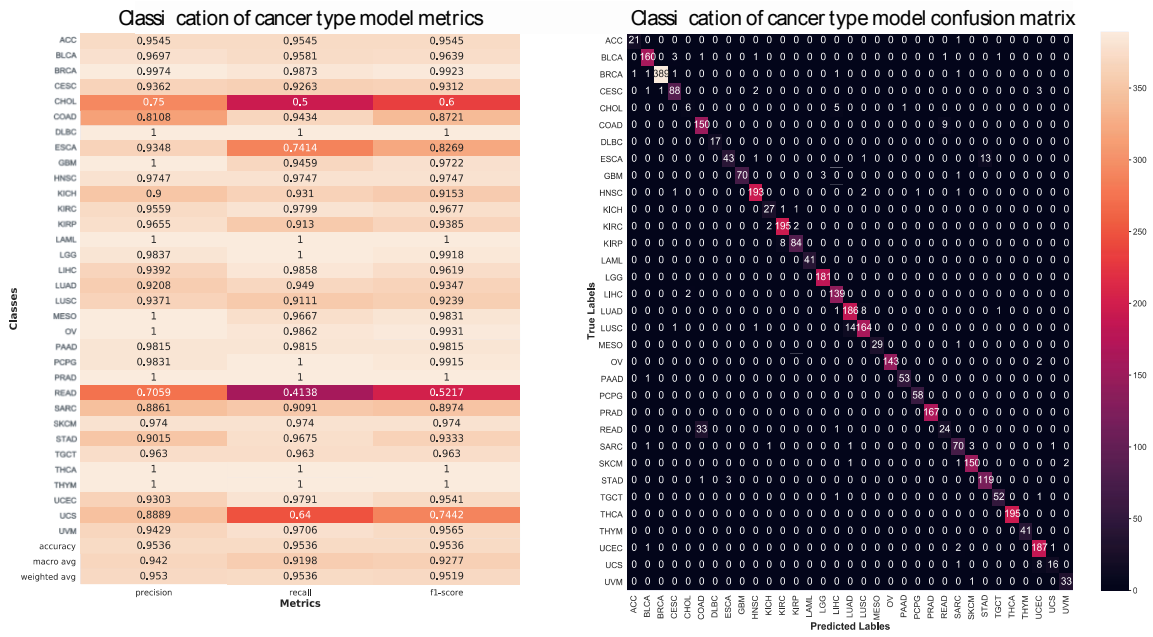

Figure 1: Classification of Cancer type. The confusion matrix detailing sample type specific performance for the GBT classification of tumor transcriptomes. 33 cancer types were considered by the model as annotated by their four letter TCGA code. The scale bar on the right-hand vertical axis denotes the density for each tile where dark tiles indicate low number of predicted values and red/white values indicate high numbers of predicted values. The major diagonal denotes the cancer type match between predicted and true labels where true labels are annotated along the left-side vertical axis and predicted labels are annotated across the horizontal axis.

Figure 2. Observed sites of metastatic progression in the TCGA database

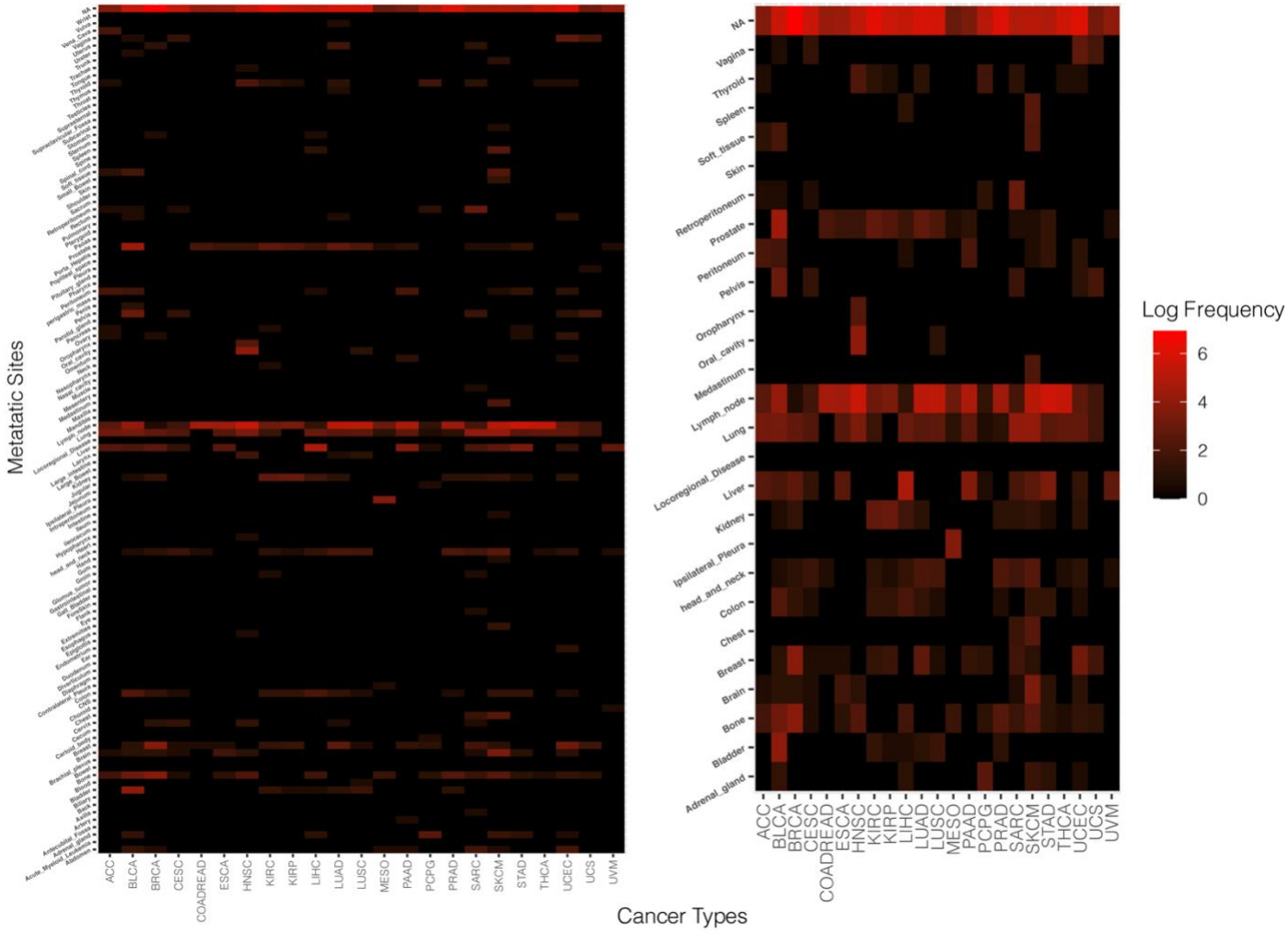

Figure 2. Thirty-three cancers in the TCGA database have recorded RNA sequencing data. Within twenty-three projects 125 anatomic locations have clinically annotated metastatic progression. Unique metastatic sites of progression found within the population are annotated on the vertical axis. The cancer type four letter codes are annotated on the horizontal axis. The heatmaps are stratified by log frequency of occurrence in the data set. The right heatmap are were locations with the greatest frequency amongst all sites. COAD and READ have been combined in this section of the analysis.

Figure 3. Prediction of Site-specific Metastases

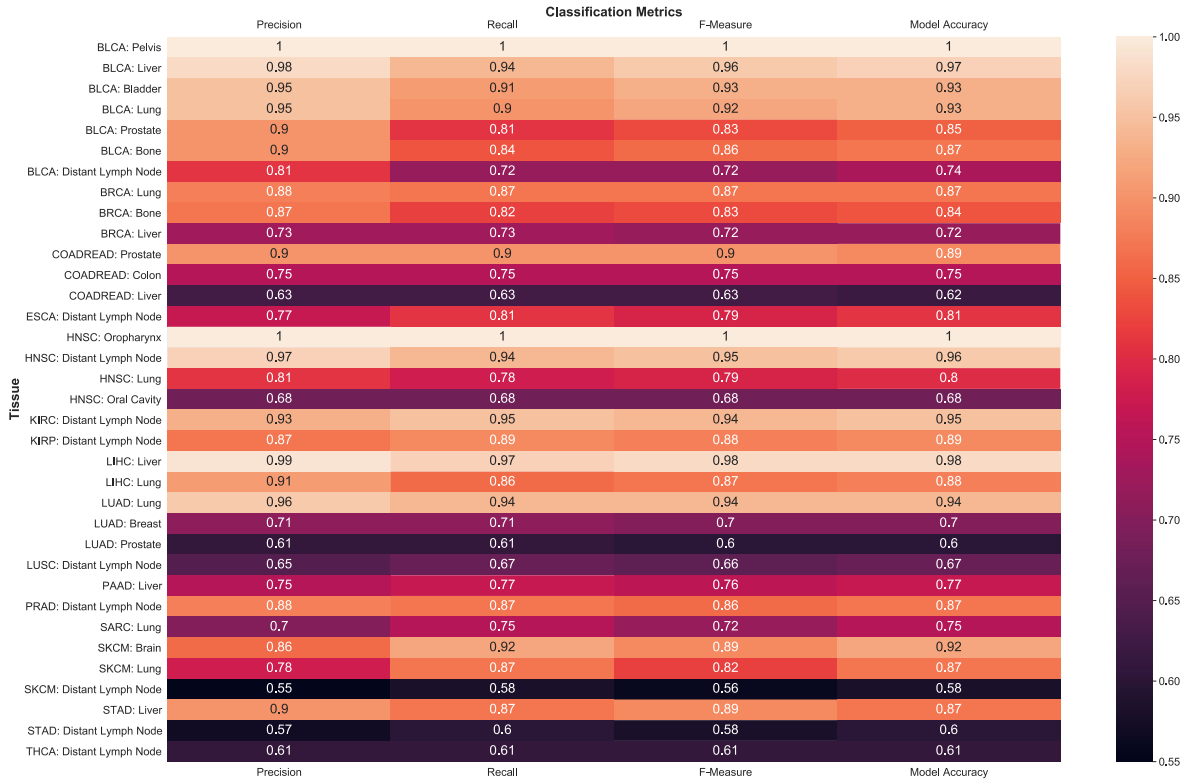

Figure 3. Displayed are the model performance metrics predicting site specific metastasis. The data was classified following a train test split where 30% of the annotated transcriptome population were held out. The performances reported are on out of bag instances that were not used as synthetic templates for training. Model performances are reported on a scale of 0 to 1. Cancer type label are in the four-letter code from the TCGA database. Total support are instances in the test set where a positive class was observed are reported in supplementary data tables.

Table 1. Average model metrics by cancer

TCGA-Project	Avg. Precision	Avg. Recall	Avg. F-Measure	Avg. Model Accuracy
BLCA	0.93	0.87	0.89	0.90
BRCA	0.82	0.80	0.81	0.81
COADREAD	0.76	0.76	0.76	0.75
ESCA	0.77	0.81	0.79	0.81
HNSC	0.86	0.85	0.85	0.86
KIRC	0.93	0.95	0.94	0.95
KIRP	0.87	0.89	0.88	0.89
LIHC	0.95	0.91	0.93	0.93
LUAD	0.76	0.75	0.75	0.75
LUSC	0.65	0.67	0.66	0.67
PAAD	0.75	0.77	0.76	0.77
PRAD	0.88	0.87	0.86	0.87
SARC	0.70	0.75	0.72	0.75
SKCM	0.73	0.79	0.76	0.79
STAD	0.73	0.74	0.74	0.74
THCA	0.61	0.61	0.61	0.61

Table 1. Displayed are the cumulative model performance metrics aggregating all locations for each cancer type. The cancers are labeled with their four letter TCGA code. Model metrics reported right to left were classification precision, classification recall, classification F-Measure and classification accuracy. Model performance variance and standard deviation are reported in the supplementary materials. Positive and Negative class specific performance reported in supplementary data tables.

Figure 4. Simulated and observed overrepresented GO biological processes

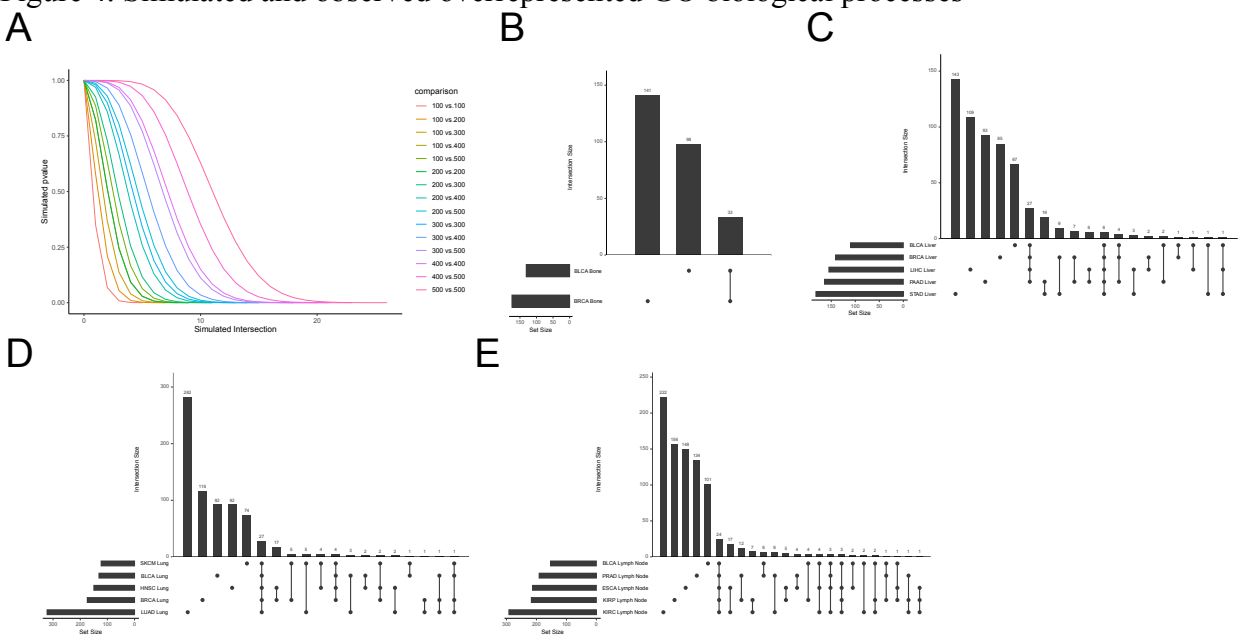

Figure 4: Gene set enrichment analysis was conducted using the clusterProfiler package in R. The Go ontology database was used to investigate feature enrichment in Biological Processes for each metastatic location in each cancer type that was classified by the model. The upsest plots were generated using the UPsetR package. The bars represent the GO IDs with an adjusted pvalue <0.05 after Bonferroni correction. A. Simulated enrichment of randomly selected transcript features overrepresented in GO. B. Enriched processes in Bone metastases. C. Enriched processes in Liver metastases. D. Enriched processes in Lung metastases. E. Enriched processes in Lymph Node metastases. Statistical significance and GO:ID enrichment results included in supplementary data tables.

Figure 5. Shared significantly overrepresented biological processes

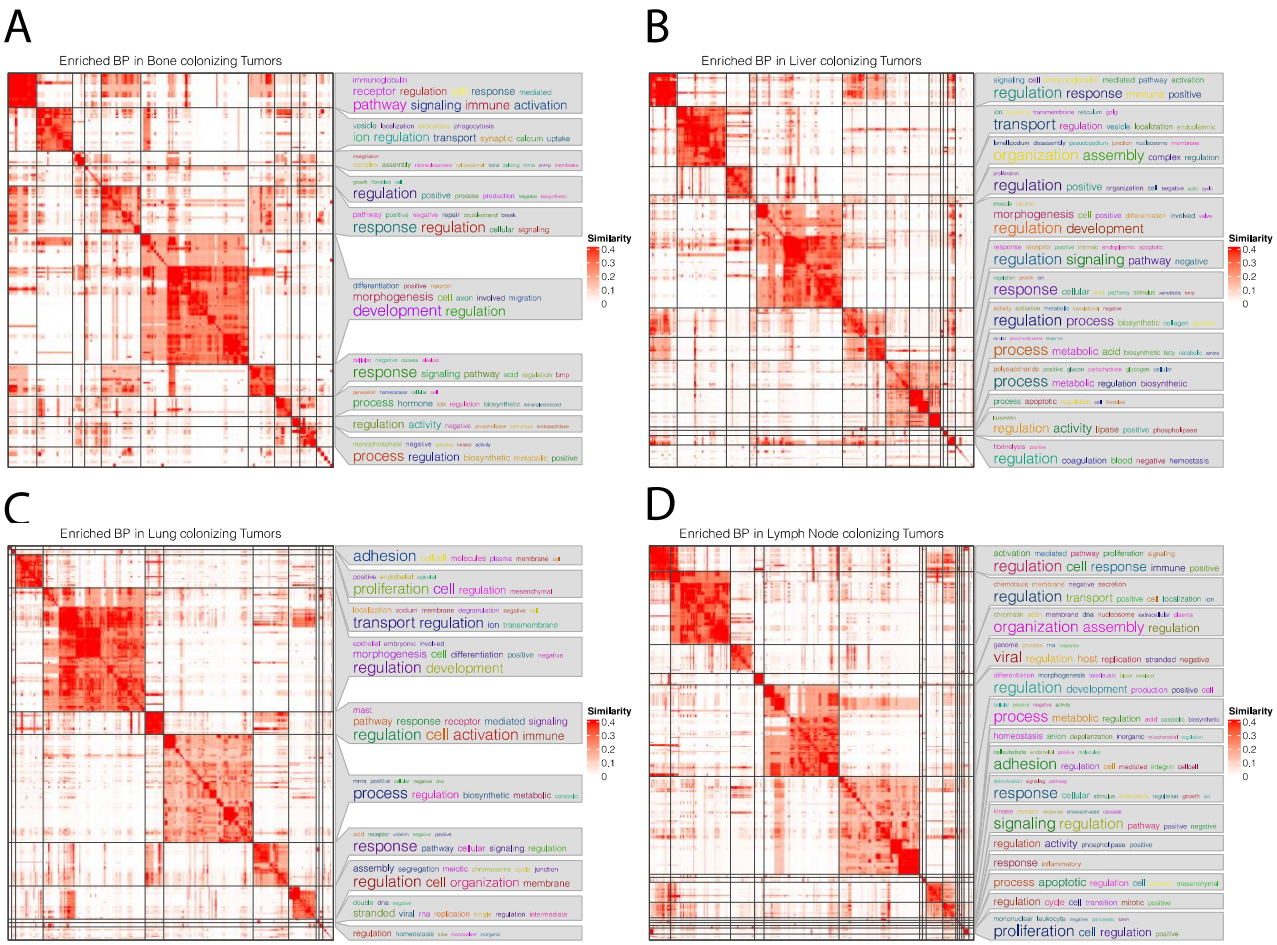

Figure 5: Gene set enrichment analysis was conducted using the clusterProfiler package in R. The Go ontology database was used to investigate feature enrichment in Biological Processes for each metastatic location in each cancer type that was classified by the model. SimplifyEnrichment package was used to cluster the semantic similarity between shared overrepresented biological processes in tumors metastasizing to concordant locations. A. Enriched processes in Bone metastases. B. Enriched processes in Liver metastases. C. Enriched processes in Lung metastases. D. Enriched processes in Lymph Node metastases. Statistical significance and GO:ID enrichment results included in supplementary data tables. Similarity scores are on a scale of 0 to 1.

2.8 Bibliography:

1 Siegel, R. L., Miller, K. D. & Jemal, A. Cancer statistics, 2020. *CA Cancer J Clin* 70, 7-30, doi:10.3322/caac.21590 (2020).

2 Massague, J. & Obenauf, A. C. Metastatic colonization by circulating tumour cells. *Nature* 529, 298-306, doi:10.1038/nature17038 (2016).

3 Lopez, M. *et al.* [Role of adjuvant chemotherapy in the choice of chemotherapeutic treatment of metastatic breast cancer]. *Clin Ter* 160, 489-497 (2009).

4 Teoh, S. T., Ogrodzinski, M. P., Ross, C., Hunter, K. W. & Lunt, S. Y. Sialic Acid Metabolism: A Key Player in Breast Cancer Metastasis Revealed by Metabolomics. *Front Oncol* 8, 174, doi:10.3389/fonc.2018.00174 (2018).

5 Ward, P. S. & Thompson, C. B. Metabolic reprogramming: a cancer hallmark even warburg did not anticipate. *Cancer Cell* 21, 297-308, doi:10.1016/j.ccr.2012.02.014 (2012).

6 Hart, I. R. & Fidler, I. J. Role of organ selectivity in the determination of metastatic patterns of B16 melanoma. *Cancer Res* 40, 2281-2287 (1980).

7 Fidler, I. J. Seed and soil revisited: contribution of the organ microenvironment to cancer metastasis. *Surg Oncol Clin N Am* 10, 257-269, vii-viiii (2001).

8 Langley, R. R. & Fidler, I. J. The seed and soil hypothesis revisited--the role of tumor-stroma interactions in metastasis to different organs. *Int J Cancer* 128, 2527-2535, doi:10.1002/ijc.26031 (2011).

9 Hoshino, A. *et al.* Tumour exosome integrins determine organotropic metastasis. *Nature* 527, 329-335, doi:10.1038/nature15756 (2015).

10 McDonald, O. G. *et al.* Epigenomic reprogramming during pancreatic cancer progression links anabolic glucose metabolism to distant metastasis. *Nat Genet* 49, 367-376, doi:10.1038/ng.3753 (2017).

11 Paget, S. The distribution of secondary growths in cancer of the breast. 1889. *Cancer Metastasis Rev* 8, 98-101 (1989).

12 Fidler, I. J. & Kripke, M. L. The challenge of targeting metastasis. *Cancer Metastasis Rev* 34, 635-641, doi:10.1007/s10555-015-9586-9 (2015).

13 Budczies, J. *et al.* The landscape of metastatic progression patterns across major human cancers. *Oncotarget* 6, 570-583, doi:10.18632/oncotarget.2677 (2015).

14 You, S. *et al.* Integrated Classification of Prostate Cancer Reveals a Novel Luminal Subtype with Poor Outcome. *Cancer Res* 76, 4948-4958, doi:10.1158/0008-5472.CAN-16-0902 (2016).

15 Bendinelli, P. *et al.* Microenvironmental stimuli affect Endothelin-1 signaling responsible for invasiveness and osteomimicry of bone metastasis from breast cancer. *Biochim Biophys Acta* 1843, 815-826, doi:10.1016/j.bbamcr.2013.12.015 (2014).

16 Kimbung, S. *et al.* Transcriptional Profiling of Breast Cancer Metastases Identifies Liver Metastasis-Selective Genes Associated with Adverse Outcome in Luminal A Primary Breast Cancer. *Clin Cancer Res* 22, 146-157, doi:10.1158/1078-0432.CCR-15-0487 (2016).

17 Gao, Y. *et al.* Metastasis Organotropism: Redefining the Congenial Soil. *Dev Cell* 49, 375-391, doi:10.1016/j.devcel.2019.04.012 (2019).

18 Minn, A. J. *et al.* Genes that mediate breast cancer metastasis to lung. *Nature* 436, 518-524, doi:10.1038/nature03799 (2005).

19 Landemaine, T. *et al.* A six-gene signature predicting breast cancer lung metastasis. *Cancer Res* 68, 6092-6099, doi:10.1158/0008-5472.CAN-08-0436 (2008).

20 Korde, L. A. & Gralow, J. R. Can we predict who's at risk for developing bone metastases in breast cancer? *J Clin Oncol* 29, 3600-3604, doi:10.1200/JCO.2011.35.7038 (2011).

21 Skardal, A., Devarasetty, M., Forsythe, S., Atala, A. & Soker, S. A reductionist metastasis-on-a-chip platform for in vitro tumor progression modeling and drug screening. *Biotechnol Bioeng* 113, 2020-2032, doi:10.1002/bit.25950 (2016).

22 Kang, Y. *et al.* A multigenic program mediating breast cancer metastasis to bone. *Cancer Cell* 3, 537-549, doi:10.1016/s1535-6108(03)00132-6 (2003).

23 Taylor, I. W. *et al.* Dynamic modularity in protein interaction networks predicts breast cancer outcome. *Nat Biotechnol* 27, 199-204, doi:10.1038/nbt.1522 (2009).

24 Chen, L. L., Blumm, N., Christakis, N. A., Barabasi, A. L. & Deisboeck, T. S. Cancer metastasis networks and the prediction of progression patterns. *Br J Cancer* 101, 749-758, doi:10.1038/sj.bjc.6605214 (2009).

25 Zhou, X. & Liu, J. A computational model to predict bone metastasis in breast cancer by integrating the dysregulated pathways. *BMC Cancer* 14, 618, doi:10.1186/1471-2407-14-618 (2014).

26 Costa-Silva, B. *et al.* Pancreatic cancer exosomes initiate pre-metastatic niche formation in the liver. *Nat Cell Biol* 17, 816-826, doi:10.1038/ncb3169 (2015).

27 Vakoc, C. R. & Tuveson, D. A. Soils and Seeds That Initiate Pancreatic Cancer Metastasis. *Cancer Discov* 7, 1067-1068, doi:10.1158/2159-8290.CD-17-0887 (2017).

28 Liu, Z. *et al.* Predicting distant metastasis and chemotherapy benefit in locally advanced rectal cancer. *Nat Commun* 11, 4308, doi:10.1038/s41467-020-18162-9 (2020).

29 Colaprico, A. *et al.* TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data. *Nucleic Acids Res* 44, e71, doi:10.1093/nar/gkv1507 (2016).

30 Wickham, H. *et al.* Welcome to the Tidyverse. *Journal of Open Source Software* 4, 1686, doi:10.21105/joss.01686 (2019).

31 Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. SMOTE: Synthetic Minority Over-sampling Technique. arXiv:1106.1813 (2011). <https://ui.adsabs.harvard.edu/abs/2011arXiv1106.1813C>.

32 Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12, 2825-2830 (2011).

33 Hao, J. G. & Ho, T. K. Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language. *Journal of Educational and Behavioral Statistics* 44, 348-361, doi:10.3102/1076998619832248 (2019).

34 Saeys, Y., Inza, I. & Larranaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics* 23, 2507-2517, doi:10.1093/bioinformatics/btm344 (2007).

35 GeneOverlap: Test and visualize gene overlaps. R package version 1.24.0 (2020).

36 Yu, G., Wang, L. G., Han, Y. & He, Q. Y. clusterProfiler: an R package for comparing biological themes among gene clusters. *OMICS* 16, 284-287, doi:10.1089/omi.2011.0118 (2012).

37 Yu, G. *et al.* GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics* 26, 976-978, doi:10.1093/bioinformatics/btq064 (2010).

38 Gu, Z. *simplifyEnrichment: Simplify Functional Enrichment Results.*, 2020).

39 Mi, H., Muruganujan, A., Ebert, D., Huang, X. & Thomas, P. D. PANTHER version 14: more genomes, a new PANTHER GO-slim and improvements in enrichment analysis tools. *Nucleic Acids Res* 47, D419-D426, doi:10.1093/nar/gky1038 (2019).

40 Conway, J. R., Lex, A. & Gehlenborg, N. UpSetR: an R package for the visualization of intersecting sets and their properties. *Bioinformatics* 33, 2938-2940, doi:10.1093/bioinformatics/btx364 (2017).

41 Cancer Genome Atlas Research, N. *et al.* The Cancer Genome Atlas Pan-Cancer analysis project. *Nat Genet* 45, 1113-1120, doi:10.1038/ng.2764 (2013).

42 Tomczak, K., Czerwinska, P. & Wiznerowicz, M. The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge. *Contemp Oncol (Pozn)* 19, A68-77, doi:10.5114/wo.2014.47136 (2015).

43 Cancer Genome Atlas Research, N. *et al.* Integrated genomic characterization of oesophageal carcinoma. *Nature* 541, 169-175, doi:10.1038/nature20805 (2017).

44 Kullback, S. & Leibler, R. A. On Information and Sufficiency. *Ann. Math. Statist.* 22, 79-86, doi:10.1214/aoms/1177729694 (1951).

45 Friedl, P. & Gilmour, D. Collective cell migration in morphogenesis, regeneration and cancer. *Nat Rev Mol Cell Biol* 10, 445-457, doi:10.1038/nrm2720 (2009).

46 Donoghue, M. T. A., Schram, A. M., Hyman, D. M. & Taylor, B. S. Discovery through clinical sequencing in oncology. *Nature Cancer* 1, 774-783, doi:10.1038/s43018-020-0100-0 (2020).

CHAPTER 3

SPARCE: STATISTICAL PREPROCESSING OF ATTRIBUTES VIA RECURSIVE CROSS ELIMINATION

[1] Michael Skaro, Andrea Sboner, Jonathan Arnold. Submitted to IEEE transactions on Computational Biology and Bioinformatics. March 29, 2022.

3.1 Abstract:

We have developed an end-to-end preprocessing software for omics feature selection. The sparce software can operate on all forms of omics data arising from DNA, RNA and methylation experiments. The software uses an ensemble of automatic feature selection algorithms wrapped with a soft voting classifier to grade and select features in genomic window bins. The sparce software dynamically adjusts feature bin density and bin size according to feature prevalence across each bin in each chromosome. The software discretizes overly dense bins where the feature to sample ratio is overly imbalanced and conversely dynamically extends and merges bins to evaluate all omics features equally on each chromosome. Selected features are automatically normalized and standardized for deep learning and other classification wrapper algorithms. The SPARCE software provides a straightforward mechanism for feature selection following standard omics preparation pipelines.

3.2 Introduction:

Data generation has driven a new era of software development in the field of artificial intelligence that as of 2012 was parallel to Moore's law but, is now far out pacing it[1]. This phenomenon has been especially true in the field of genomics. A common problem bioinformaticians are facing is the data volume to sample ratio. This ratio is better known as the Big-p, Little-n ($p >> n$) problem, where there are far more predictors, p, than number of samples, n. These high dimensional datasets are produced by machines that can measure hundreds of thousands of genomic features within an organism which far exceeds the number of samples we are fiscally capable of generating. Fitting an ML model with more features than samples will invariably introduce unresolvable overfitting[2]. This introduces an important and non-trivial problem of feature selection for genomics data.

Feature selection from genomics data begins with feature engineering. As most genomics data arrive in a raw and unorganized format, the features within the data must be aligned, quantified, normalized, and often standardized across runs. Generating these features is a proxy for representing the organism in a measured format that can be manipulated for modeling biological phenomena. Feature selection is the process of identifying the value of each measurement and how they affect the derived biological conclusions from the differences from within a study population. Feature selection is increasingly important as the curse of dimensionality is growing with the expansion of new assays that can generate hundreds of thousands of concurrent measurements.

There are well known gene selection approaches documented in the genomics literature[3-15]. These selection techniques may be split into three categories; supervised, semi-supervised and unsupervised feature selection. Supervised gene selection is the process of measuring gene value

using labeled data. The data labels are the target, and the methods use discrimination between the target classes to categorize features, while unsupervised methods are not concerned with the classification outcomes and attempt to categorize samples based on intricacies of the data that are inherent within the feature measurements, such as variance, distance and quantile stratification. Features can be categorized into one of four categories; (I) completely irrelevant and noisy features, (II) weakly relevant and redundant features, (III) weakly relevant and non-redundant features, and (IV) strongly relevant features. They suggest the theoretically optimal set of features used is a combination of sets III and IV where, you can describe your classes using the strongly relevant and non-redundant features[16]. An optimal feature selection method will reduce the features from categories I and II and, maximize the features from categories III and IV. The effectiveness of a feature selection approach may be judged by the homogeneity of the clusters or groups produced in an unsupervised approach or by a wrapper classifier. If the wrapper classifier has high evaluation metrics when tested on the hold out set, the selection method is validated as finding valuable features.

Multiple software packages have been developed to select features from omics data using wrapper methods[3-6,11,12,16]. A well-known wrapper method developed in 2017 was packaged as a Bioconductor package for feature selection in omics data[6]. This software uses a three-step approach for feature selection, starting with a dimensionality reduction principal component analysis feeding into a recursive feature elimination loop and wrapped in a classifier. While this package is simple to use, it only slightly improved on the benchmark models it was compared against. This well-known wrapper method draws upon no genomic topology during feature selection which may draw false interactions, is a single threaded design, and the evaluation of this workflow was on small and outdated benchmark datasets[6]. In the last two years multiple algorithms

have been created that are outfitted for feature selection in high-dimensional omics data[17-34]. A main objective of many of these papers has been to classify the tumor type from omics data. Tumor type classification from omics data is an extremely valuable classification as it directly pertains to the clinical care for patients. The development of side-by-side tools for cancer type classification is critical to improve patient outcomes. We have previously investigated the use of feature selection in classification of tumor metastases[12].

Here we present multiple extensions of our previously published work to select features in transcriptomics data and use our approach to select features in an extreme case of Big-p, Little-n ($p >> n$) in which we select features from a legacy Illumina methylation array with 485,000 features and 32 classes for classification. The SPARCE method was wrapped with a neural network wrapper for the classification of cancers of unknown primary (CUPs) using methylation data[12]. The objective of this study is to demonstrate the broad applicability of the SPARCE software to be applied to any genomic information in a multi-class classification problem and extend the accompanying ML model to the EPIC Infinium methylation Array that is currently offered by Illumina. We apply the model that was trained on the primary tumor data using the legacy Illumina 450k array and apply it to tumor data mined from the GEO database generated on the latest infimum array platform (GSE116298)[35,36].

3.3 Methods:

Data mining TCGA training data:

Tumor methylation data was downloaded from the The Cancer Genome Atlas (TCGA) database using the TCGABiolinks Bioconductor software[37]. A project query with the four-letter code signifying the cancer type was created for each of the thirty-two cancer types. Patient data from each of the projects were downloaded and combined into a three-dimensional tensor with dimensions [32,485000,10000], where there are thirty-two cancer types, 485,000 methylation probes, and 10,000 patient samples. The methylation probes were organized by chromosomal locations using the GRanges Bioconductor package and Illumina methylation reference list for hg19[38,39].

Data mining External Data Sets:

External evaluation required two steps, data mining and data normalization prior to deployment of MetNet on the GEO dataset. The data mined from the GEO dataset was retrieved using the GEO query R package published on Bioconductor. The GSE ID was used in the getGeo function to return a GSE GEO Series (GSE) (lists of GSM files that together form a single experiment) and GEO Dataset (GDS) object[40]. The GSE matrices were exported to two-dimensional arrays. The two-dimensional arrays were indexed and only features selected from the feature selection step were intersected. 4760 of the 5000 selected probes co-occurred in the Infinium MethylationEPIC array and were carried over for external classification. A two-step standardization and data normalization were performed.

Data Preprocessing, normalization, filtering, and standardization:

The EPIC array samples were generated on a later iteration of the Illumina platform. We added a data standardization step to migrate only the methylation probes that were co-occurring

on the Illumina 450k array and the Illumina Infinium Methylation EPIC. Quantile normalization of selected probes and batch normalization were performed with the combatR package prior to model classification of external data sets[41]. The two-dimensional arrays were concatenated into a three-dimensional tensor of dimesons [4760, 48,50] where the 4760 probes are fed into the input layer, the 48 samples with a batch size of 50. A note* as the input features did not match the input layer the original model, a second iteration was trained, this did not affect the performance or overall design.

Feature selection:

The feature selection process was expanded form Skaro et al 2021[12]. The expansions of the feature selection methods were in the the diversification in the data types for evaluation, features were discretized into chromosomal bins, the manipulation of those bins and the wrapper classifier that evaluated the performance of the feature selection. The evaluation of the features in this experiment was accomplished using the bin manipulation module and feature selection module in the sparce pypi package. The bin manipulations module accepts nFeatures and nBins as arguments. The nFeatures argument was set to 5000 features which is approximately 1% of the data set and for the nBins argument was set to 97 bins to expand across all non-sex chromosomes. The features for each bin were evaluated with the feature selection module in the sparce pypi package. Only features supported by all algorithms were kept from each bin. Remaining features from each bin were concatenated for wrapper evaluation.

The five algorithms used for feature selection mechanism for the SPARCE approach were chi square, random forest regression, random forest classification, lasso regression. Models were wrapped in a recursive feature eliminator. The SelectKBest, Chi2 and MinMaxScaler Libraries from Sklearn and feature selection module were used[42,43]. The RFE was trained on each bin of

features and the importance of each feature was obtained through the feature importance attribute from the RFE method. If the window encountered a bin in which the model had less instances than observations the mini bin method in the bin manipulations module was called. The RFE and LogisticRegression libraries from Sklearn and feature selection module were used. If the models encountered a bin with less than one standard deviation below the mean number features in a bin on the current chromosome, the bin expanded to capture more features by utilizing the expand bin method in the bin manipulations module. Subsequent bin start and end positions were adjusted. For each bin we cross validated the results from each algorithm by extracting support values using the get support methods. Total support from feature support Booleans vector was summed for each feature in each bin across all five methods. Features receiving either five votes from the feature selection algorithms or the top one percent of features were kept in each bin [42,43].

MetNet Training, validation, and testing:

Selected features were prepared into a three-dimensional tensor as the input layer for the artificial neural network. 5000 array probes were selected and inputted into the first layer of a multilayer perceptron. The multilayer perceptron was prepared with the tensorflow and keras packages[44]. The neural network design was simple: A sequential layer, followed by three dense layers of 80 neuron units, with uniform kernel initialization, input dimensions of 5000, a drop rate of 0.5 and tahn activation function. The dense layers were followed by a dense output layer of 30 units, uniform kernel initialization, and softmax activation function. The optimization was stochastic gradient descent with a learning rate of 0.001. The loss was evaluated with categorical cross entropy. The metric was accuracy. The final MetNet model was trained for 200 epochs after grid search for optimal hyperparameters. The grid search cycled combinations of eight hyperparameters; batch size, epochs, optimization, learning rate, momentum, dropout, activation,

and neuron units. Classification of tumor types were split; 80% of the tumors were used to train the MetNet model, while 20% of the data was held for validating the model. The model evaluations were drawn from the validation set.

Evaluation on External data:

An external data set was evaluated using the trained MetNet model. A total of 47 tumors were evaluated as a test data set with no further training to simulate a set of cancers of unknown primary. The tumor data were classified into one of the 32 cancer types as a multi-class classification. The evaluation of the model precision, recall, and accuracy are reflected exactly as the legacy training.

Mini Bins:

The mini bins function in the bin manipulation class is a recursive function that breaks down bins based on the imbalanced big-p, little-n ratio. If the number of features in a genomic bin are greater than the number of samples used to describe the features, the SPARCE software will break the genomic bins into mini bins, 10 by default. The mini bins are assessed by the feature selection class, and the mini bins are filtered for selected features. The selected features are merged, and the merged bin is reassessed. This step has a computational complexity of:

$$m^r f(L_a)(L_b)(L_b)(L_c)(L_d)(L_e)(CV_i),$$

where m is the number of mini bins made in this step, r is the recursive depth of binning, f is the number of features and L is the algorithm invoked to grade each feature, CV is the cross validation of the five algorithms, and i is the iteration of cross validation. This step is extremely computationally costly and as the depth of the recursion increases the computational time increases exponentially. The dynamic bin adjustment method was developed to abate this inflation across all the windows. All parameter, returns, and return types are documented in the documentation.

Expand Bins:

The expand bin function is a function that is intended to be called on the fly while iterating over a feature call file or gene annotation file. If a user intends to assess the value of all the features in a chromosome or genome using the SPARCE method, the genomic feature density should ideally be approximately equal. The expand function is called if the bin being assessed is sparse. The bin will expand to capture approximately equal features within the bin compared to the average number of features in the previously scanned bins. The expand bin function then adjusts the start site of the successive bins and adjusts the bin ID for all successive features. The adjustment of bins size and location following the invocation of sparce creates a dynamically changing final bin for each chromosome. In the event the final bin on a chromosome has been reduced due to expand bin calls, the user may merge the last two bins on a chromosome without significantly decreasing the feature weight. The method is bounded such that the merged number of features cannot be 1.5x the average number of features in previous bins.

Dynamic Bin Adjustment:

The dynamic bin adjustment method is the suggested method to reduce the computational time in assessing extremely dense sections of the genome. The user may opt to invoke the dynamic bin adjustment to make approximately equal size bins for the remaining space on a given chromosome or genome such that the bins never exceed the number of features and reduces the need to call the mini bins functionality. The method creates a new set of bins from which to grade features. This method does not account for the weight of features and assumes that vfeatures are equal in value.

Bin Merge:

The bin merge assesses the size of the next on a chromosome using a look first approach; if the last bin is sparsely filled compared to the preceding bins and merging the last two bins will not significantly decrease the estimated weight of features within a genomic window bin, the software will merge the current and next on a chromosome.

Parse gene annotations:

The parse gene annotations class is made up of only two methods; parse file and bin genome. The parse file expects a gene formatted file in standard GFF3 format (ref). The bin genome expects a parsed GFF3 file. These methods are the base methods that are expected to process the data prior to bin manipulation and feature selection. The methods return pandas data frames that can be returned and passed for further processing or written to comma separated files. All parameter, returns and return types are documented in the documentation.

Weighted features:

The weighted features method allows the user to delegate a score they would like to use to attach to features in the case they are a discrete variable, such as a feature in a feature call file. While the other methods assume the features are single points, the weighted features method assigns a weight by the genomic length as the size of the feature by default. Feature bins are re-adjusted to carry approximately equal weight. The weighted features may be passed into the feature selection class. A function was built using the bounded knapsack algorithm to evenly distribute the features and features weights into genomic bins[45]. For a set of features, the score of the feature is recorded in a tensor. The algorithm is bounded by the number of samples the user supplies as a parameter. The bounded knapsack algorithm was implemented in the recursive form with a

computational complexity of O(log n) where n is the number of combinations of features in the a bin window.

Features and Tensors:

The SPARCE software assumes the user will intend to select the features so as to optimize a wrapper classification algorithm. SPARCE has four methods intended to prepare files into tensors. The match feature bins and gff bins function reduces the feature set only to values occurring in the GFF file. The transpose features method in the gene annotations module parses the file into a tensor. The method expects a classification annotation file describing which genotypes belong to which classes, and the data file in long format. A three-dimensional torch tensor is returned for assessment with SPARCE. Two helper methods were built to extract a data frame in tidy format from the tensor to prepare the data for feature selection. The user may opt to prepare the data frame based on a feature or genotype.

3.4 Results:

In Figure 1. we display an overview of the workflow from the SPARCE method as a guide through the feature selection software. Illumina 450k array data was mined form the GDC data commons using the TCGAbiolinks Bioconductor package[37]. The data was similarly processed into a three-dimensional tensor of dimensions [32, 485,000,10,000], where the dimensions signify the thirty-two cancer types, 4805,000 array probe features and the 10,000 patient samples available in the TCGA database. The sparce method was applied to select features from that discriminate between tumor type in the methylation array. We deployed a neural network classifier as our wrapper classifier using the selected features as the input layer. The trained neural network, we named MetNet, was tested on a holdout set of legacy Illumina 450k array data. MetNet was then tested on an external dataset of tumors arrayed with the EPIC array data.

MetNet classified tumors into one of the thirty-one tumor types detailed in the TCGA database (Figure 2). The confusion matrix from the hold out set of primary tumors datamined from the TCGA database is displayed. The model performed in the excellent range for 28 of the thirty-one cancers evaluated with some notable exceptions. The notable failure was in Cholangiocarcinoma (CHOL). This is a notable failure addressing the needs of CUP patients[46]. As the standard of care for patients presenting with cholangiocarcinoma the approach is to treat the patient as if they have CHOL. if a pathologist fails to properly diagnose the CUP and our model encounters a CHOL patient, we will fail to properly diagnose the patient using MetNet as a side-by-side tool[46]. Setting this failure aside, the overall performance of the model was however in the excellent range. The accuracy was 0.95, precision 0.945, recall 0.945 and f1-score 0.945 (Figure 3).

We deployed our model on an external dataset mined from the GEO database deposited by Wenger et al[36]. The tumors were classified in a multi-class classification problem with 31 possible outcomes (Figure 4.). A total of 47 of the 48 tumors were classified as arising from the Brain (98%). 34 of the tumors were properly classified as the Glioblastoma grade. A total of 13 of the tumors were classified as low-grade gliomas. No tumors were misclassed into a tumor type other than glioma.

In this study we extended the SPARCE method in two distinct ways beyond the feature selection developed in our previous model. We have added multiple binning functionalities that allowed for the successful selection of features from an extremely wide dataset and demonstrated the method works on another domain of omics data. In Figure 5. we show the how SPARCE deals with extremely dense bins. In Figure 5A. the density of CpG islands in the human chromosome 10 is plotted with four bin sizes 100kb, 200kb, 500kb and 1MB, respectfully. The bins with a dense feature occurrence, where the number of features outnumbers the samples for a given window, were broken into smaller mini bins (Figure 5B.). Features were recursively assessed within the mini bins by SPARCE building small multivariate wrapper models within each mini bin (Figure 5C.). Selected features from all mini bins were combined and reassessed prior to sparce moving onto the next genomic window (Figure 5D.).

Sparse genomic windows were dynamically adjusted in size to increase the number of array probes to approximately assess all features equally in each bin (Figure 6B.). Downstream bin start locations are updated and the features within the extended bin are assessed with the sparce method (Figure 6C. and 6D.). The dynamic binning allowed the SPARCE method to eliminate recursively low value probes in one pass; however, in the event that the SPARCE method encountered an extremely dense data set where the first pass does not produce less features than sample number

the user is able to run, the SPARCE method is recursively called using the selected features from the first pass.

Two algorithms were programmed to accommodate adjustment of genomic windows and bin modification. If a bin is extremely small due to the adjustment of previously analyzed bins, the software offers a merge bin function (Figure 7). The SPARCE mini bin function exponentially increases the computational complexity of feature selection based on the depth bin minimizations. In significantly dense regions of the genome a user may use the dynamic bin function to fit all successive bins such that the features within that bin never outnumber the samples in the experiment (Figure 7A). This decreases the number of mini bin function calls and bin reassessments called on a given chromosome (Figure 7B). Finally, a bin may be dynamically adjusted using a genomic weight in combination with feature calls. The SPARCE software can evaluate both point mutations and large features. The recursive implementation of the bounded 0-1 knapsack algorithm is designed to balance the weights of genetic features. The SPARCE software preprocesses the vcf file into a sparse tensor filled with binary values describing the feature observations in each sample. The values from the tensor serve as the score or the algorithm[45]. The implementation is bounded by the number of features in the experiment (Figure 7B). Briefly, the bounded 0-1 knapsack algorithm implementation deterministically fills all bins in a chromosome given the genomic weight, variant binary scores and upper bound. The recursive implementation operates in O(n) computational time complexity.

3.5 Discussion:

The SPARCE software was designed as an ensemble method for selecting transcriptomics features to predict site-specific metastases from primary tumors. We demonstrated in the Skaro et al. that feature selection was an effective approach in identifying the biological determinants underlying metastatic organotropism[12]. In this research we demonstrate this SPARCE approach was an effective method selecting feature in high dimensional methylomics data sets and that the selected features may be used to train a neural network to classify tumor types in the current Illumina Infinium array.

We applied the SPARCE method to an extremely high dimensional dataset with approximately 485,000 features and about 10,000 patient samples from TCGA (Figure 1). We developed two extensions from the previous methodology to accommodate for the extremely wide dataset. Feature selection in a sliding window binned features into dynamically size adjusted and recursively assessed feature bins. The dynamic bin evaluation allowed SPARCE to equally assess the value of array probes for discriminating between cancer type (Figure 1). Similar to Skaro et al 2021, this problem was a multiclass classification problem with 32 cancer types of data[12]. The SPARCE method effectively reduced the dimensionality of the dataset from 485,000 features to 5000 selected features for tumor classification (Figure 1). The final trained and tuned architecture MetNet, classified the test set of primary tumor data. The model performed in the excellent range at an average accuracy of 0.95, precision 0.945, recall 0.945 and f1-score 0.945 (Figure 2). The confusion matrix with all classified tumor populations showed the strong performance across the cancer types. The broad and consistent performance of the model demonstrated the SPARCE method is capable of selecting continuous features in problems suffering from even extremely imbalanced big-p, little-n ratio.

The MetNet classifier was applied to an external dataset data mined from the GEO database (Figure 1). A total of 47 glioblastoma tumors were assessed by the trained model, simulating a scenario of 47 cancers of unknown primary to the model (Figure 4). Two steps were required to standardize the information from the external dataset. The selected features from the training data were intersected with migrated array probes. This created a final feature set of 4760 probes. Quantile normalization was applied to the external dataset, and each tumor was a subset to the selected probes and classified into one of the 32 trained classes. MetNet model overall performed in the excellent range if you consider all tumors were properly classified aa gliomas, the only failures were to differentiating between grades (Figure 4). Almost all Grade II lesions eventually progress to High Grade Glioma (grade III/IV or HGG). Grade IV gliomas are that arise from LGG are termed "secondary GBM" to differentiate them from "primary" or "de-novo" GBM[36,47]. The etiology of the pathologies differs in the oncogenic determinants of disease and clinical characteristics[48].

The SPARCE software has been expanded from a feature selection wrapper model to an omics preprocessing software. The package has 4 modules containing 5 classes and 24 functions for omics preprocessing. The bins manipulations module contains the min-bin methods for feature selection in dense regions (Figure 5). In a sliding window the mini-bin function will discretize the bin into ten smaller bins by default. The feature selection module can be called in each of the bins, and the module recurses for each bin in which the number of features outnumbers the samples in each window (Figure 5). This method is depicted in a cartoon in which the cytoband of chromosome 10 in the human genome is plotted across the horizontal axis of the figure. A density plot of the EPIC array probes is plotted in blue above the chromosome cytoband with four different window widths annotated on the vertical axis (Figure 5A). The bins representing the sliding

window highlighted is plotted under a dense region in chromosome 10. The bin is broken into mini bins, and the SPARCE method scores the features. Selected features are pooled and reassessed in the pooled bin. Features with Boolean support from all five automatic feature selection algorithms, are kept and the rest are discarded (Figure 5B).

In parallel to a dynamic adjustment for dense regions, methods in the bin manipulations module in the SPARCE software were added to accommodate for sparse regions in the genome. In Figure 6A. we mirrored the same chromosome 10 human cytoband plot from Figure 5. The sliding window bin highlighted in red is plotted under a low-density region in the chromosome. The expand bin method increases the window size to capture approximately equal probes to the average number of features in the previous bins. The successive bin start and end windows are adjusted and updated. The SPARCE method can be called, and score features in the balanced windows (Figure 6B).

The expand bin method created a ‘thin bin dilemma’. Once again, the cytoband plot represents the method, and the red bin shows the window in consideration by the model. The final bin on a chromosome after successive bin manipulations may be considerably small (Figure 7A). We developed two methods for resolving the thin bin dilemma. By default, the final bins will fall off the annotations, and features are updated into the bin prior to if the start of the bin begins beyond the limit of a chromosome. However, in the case in which a bin begins at the end of the chromosome and is considerably small, we developed the bin merge method. The bin merge method uses a look-first to one bin ahead approach. If merging the bins does not increase the size of the current bin by greater than 1.5-fold compared to the average number features of the bins on the chromosome, the bins merge (Figure 7B).

Finally, we added methods for feature selection to accommodate a study using genomic variants as the data type. We developed the feature preprocessing module and expanded the bin manipulations module. The variant preprocessing and gene annotation modules ingest, and format variant call files formatted in HapMap format into a three-dimensional tensor that is ready for SPARCE analysis. Variant selection is significantly different than selecting a continuous attribute from a counts file or intensities dataset. Large genomic variants, hundreds of nucleotides in size, may be a main source of bin size variation, requiring function calls to the bin manipulation module methods in an otherwise sparse tensor. We implemented a weighted variant method for feature balancing in the bin manipulation module. The weighted variation method is an adaptation of the bounded 0-1 knapsack algorithm[45]. It is implemented in the recursive form in which the weight of the genomic variant is the length in nucleotide of the variant. The score of the variant is the observation state binary value of 0 or 1. The bin is bounded by the number of features such that the number of bins and the bin identity for all the variants is deterministic is O (n) time where n is the number of features.

The next steps for this model are to generalize the software and outfit the approach for genome wide association feature selection. The data generated in genome wide association studies are often extremely deep and are therefore well suited for this kind of selection technique. We plan to take advantage of large linkage disequilibrium blocks present in the sorghum bicolor genome and bin variants into the genomic decay windows of 50kb[49,50]. One possible method is to preempt the SPARCE method with an unsupervised co-linearity reduction method such as UMAP or PCA. These approaches deal with the small amounts of variability in an extremely sparce tensor reducing the computational complexity of selecting features and avoiding false positive selections. We plan to use these unsupervised approaches due to the failures of SPARCE.

3.6 Figures and tables:

Figure 1. Method Flow for methylation data

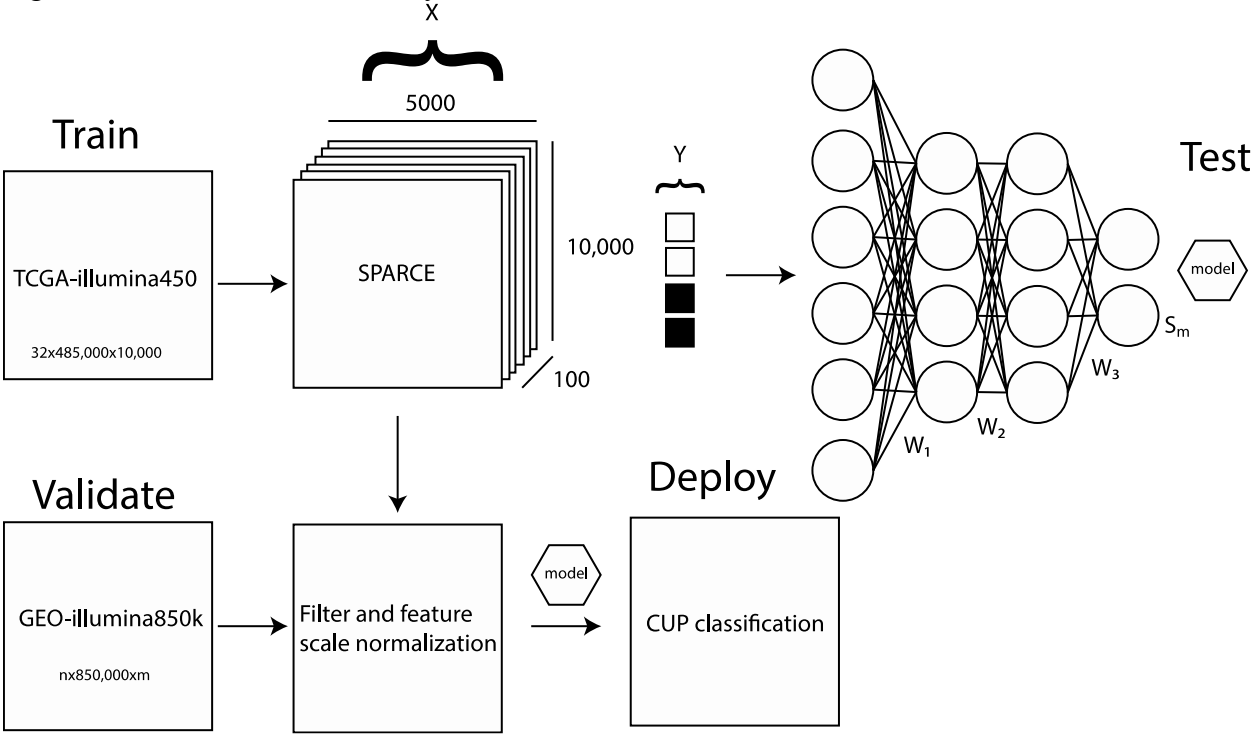

Figure 1. Data mined from public data base is cleaned and preprocessed into a three-dimensional tensor. User provides the software with a GFF files describing the methylation data that is being used to analyze their system. The SPARCE software breaks the genome into genomic windows according to the genetic annotation data (genes, transcripts, probes etc.). User may opt to provide a BinSize which will specify the number of features or genomic window size to bin features into for assessment. Each bin is fed into the feature selection ensemble. An artifical neural network classification model is built based on the selected features. The classification model is trained and validated prior to test in the hold out set. The selected features are retained for biological investigation. Migrated probes were standardized, and quantile normalized. Batch correction normalization was applied between GEO external dataset and training dataset. The model is applied to external GEO dataset.

Figure 2. Confusion matrix for classification of tumors using methylation

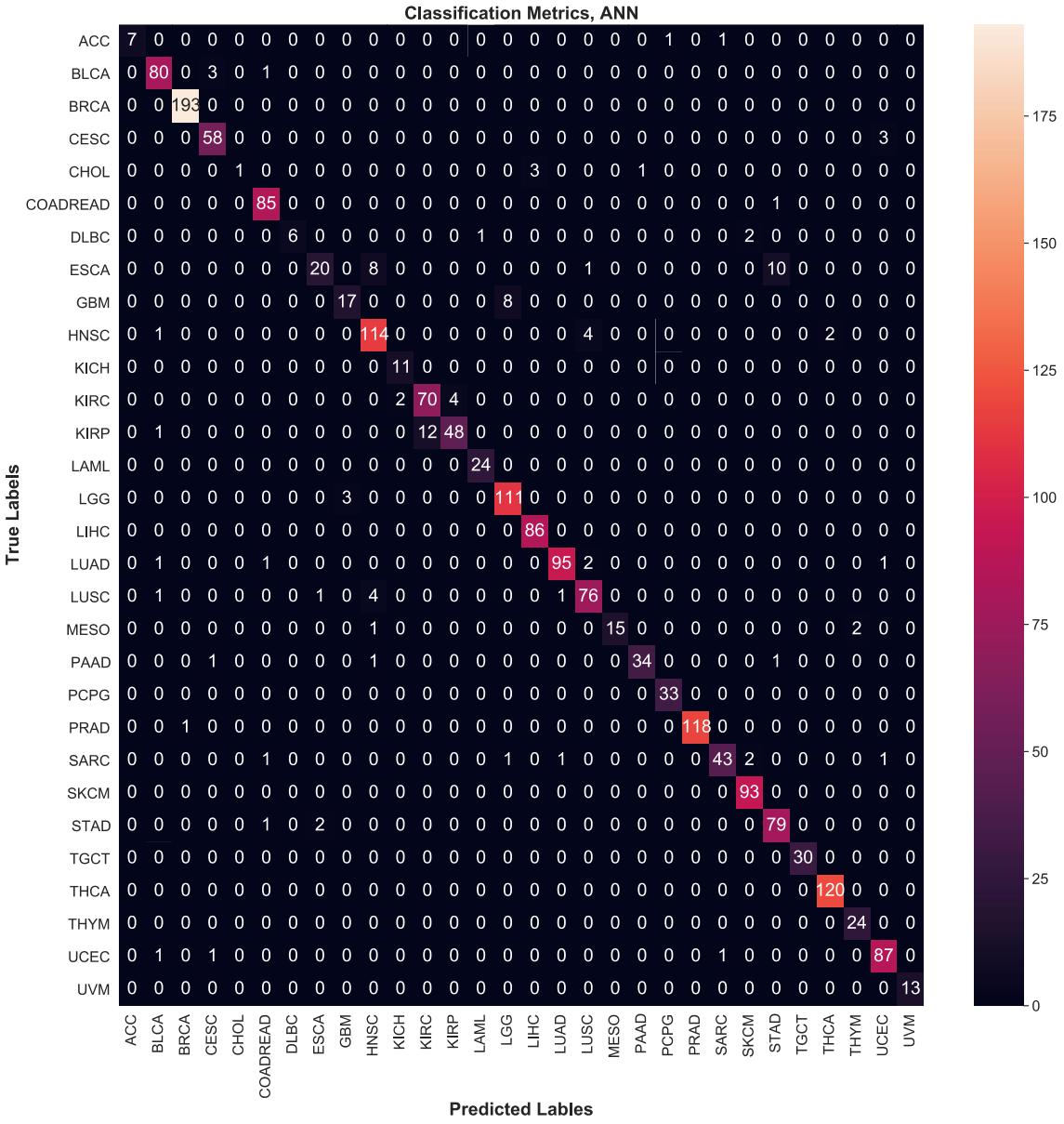

Figure 2. Thirty-one cancer type methylation Illumina450k array data were datamined from the TCGA database. A classification model was trained to predict the tumor type from selected methylation probes values. Across the horizontal axis thirty-one cancer types are labeled with their four letter TCGA code ID. The predicted labels were the inferred identities of the methylation data from the trained model in the test set. On the vertical axis thirty-one cancer types are similarly labeled. The True labels represent the true identity of the methylation data classified by the MetNet model. The scale bar represents the frequency of the values printed in the heatmap. Low values are shaded black while high values are shaded red. The high concentration of values across the major diagonal indicates a highly accurate and precise model trained for classification of tumor type.

Figure 3. Evaluation of MetNet classification of cancer type

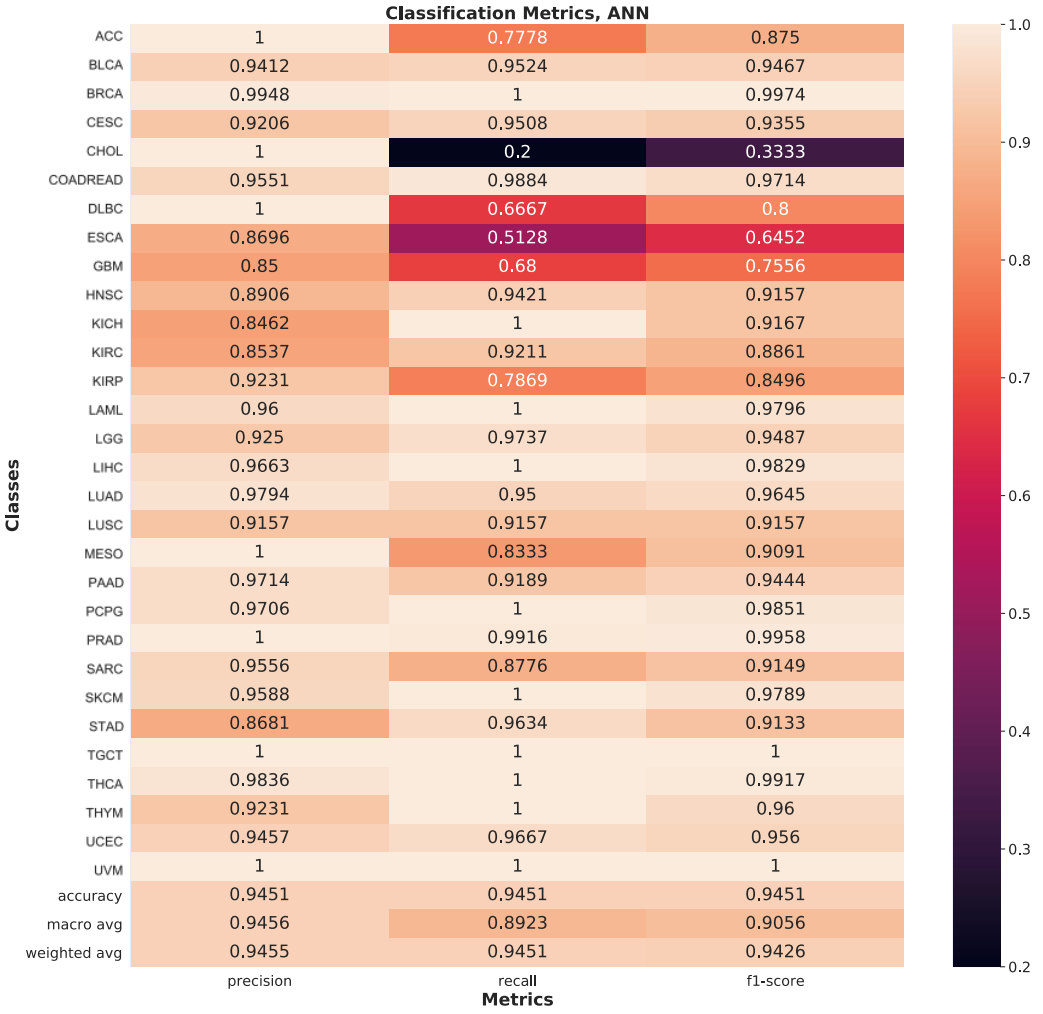

Figure 3. Thirty-one cancer type methylation Illumina450k array data were datamined from the TCGA database. A classification model was trained to predict the tumor type from selected methylation probes values. Across the horizontal axis the precision, recall and f1-score evaluating the MetNet model on the primary tumor data is shown. On the vertical axis thirty-one cancer types are labeled with their TCGA four letter code ID. Below the cancer classes the accuracy, marco average and weighted average values for the model are shown. The scale bar represents ranges from zero to one. Low scores are shaded in black while high scores are shaded in red to tan. The values in the heatmap are observed score for each cancer type. The consistently high scores show a very accurate and precise model.

Figure 4. Confusion matrix of MetNet classification of external dataset

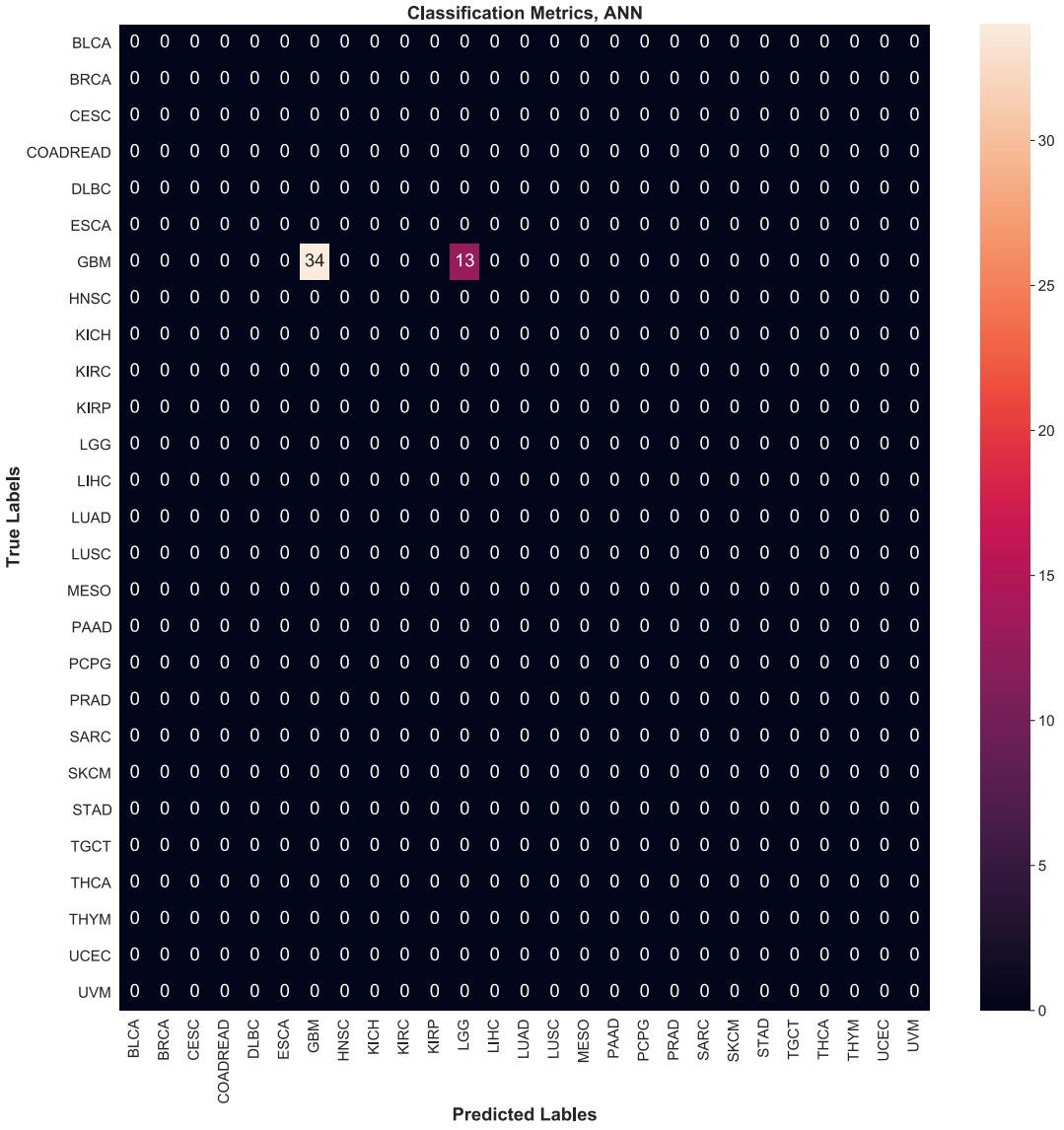

Figure 4. Thirty-one cancer type methylation Illumina450k array data were datamined from the TCGA database. A classification model was trained to predict the tumor type from selected methylation probes values. Across the horizontal axis thirty-one cancer types are labeled with their four letter TCGA code ID. The predicted labels were the inferred identities of the methylation data from the trained model in the test set. On the vertical axis thirty-one cancer types are similarly labeled. The True labels represent the true identity of the methylation data classified by the MetNet model. The scale bar represents the frequency of the values printed in the heatmap. Low values are shaded black while high values are shaded red. The model predicts 34/47 of the tumors correctly and identifies all but one of the tumors were brain tumors from the epic array data demonstrating the MetNet as a viable side-by-side diagnostic companion tool for pathologists to deploy for CUPs.

Figure 5. Dynamic binning for dense bins

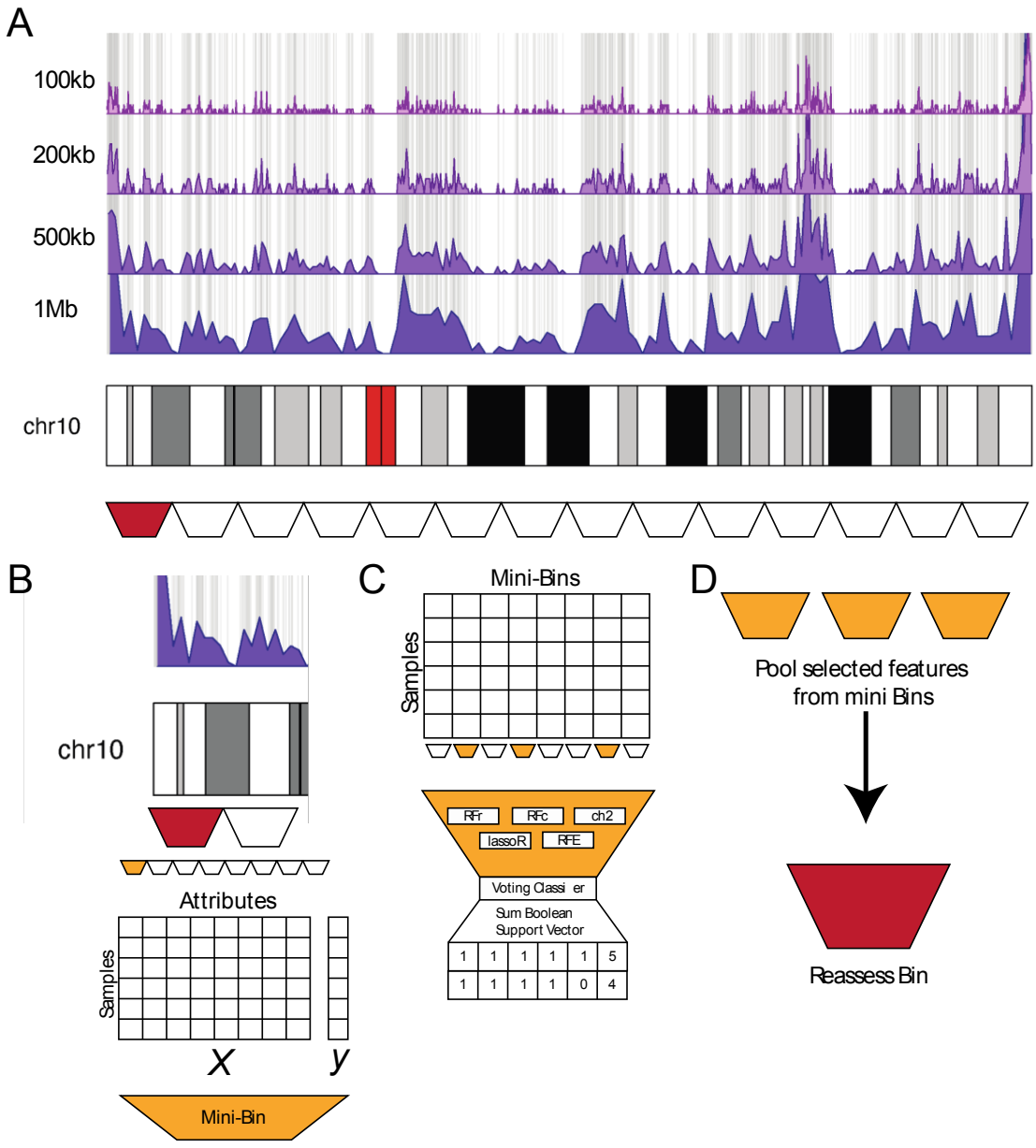

Figure 5. The density of Human Chromosome 10 CpG islands visualized using variable genomic window sizes of 100kb, 200kb, 500kb and 1MB (A). Dense bins in which more features are within the genomic bin are broken into approximately sized mini bins (B). Each mini bin is assessed using with five ML algorithms. A boolena support vector from each algorithm is returned for each features and summed. The summed features with the highest support sum are selected by a voting classifier. (C). Selected features from each mini bin are combined and reassessed as a group by sparce (D).

Figure 6. Low density Bins extended Sparce methods

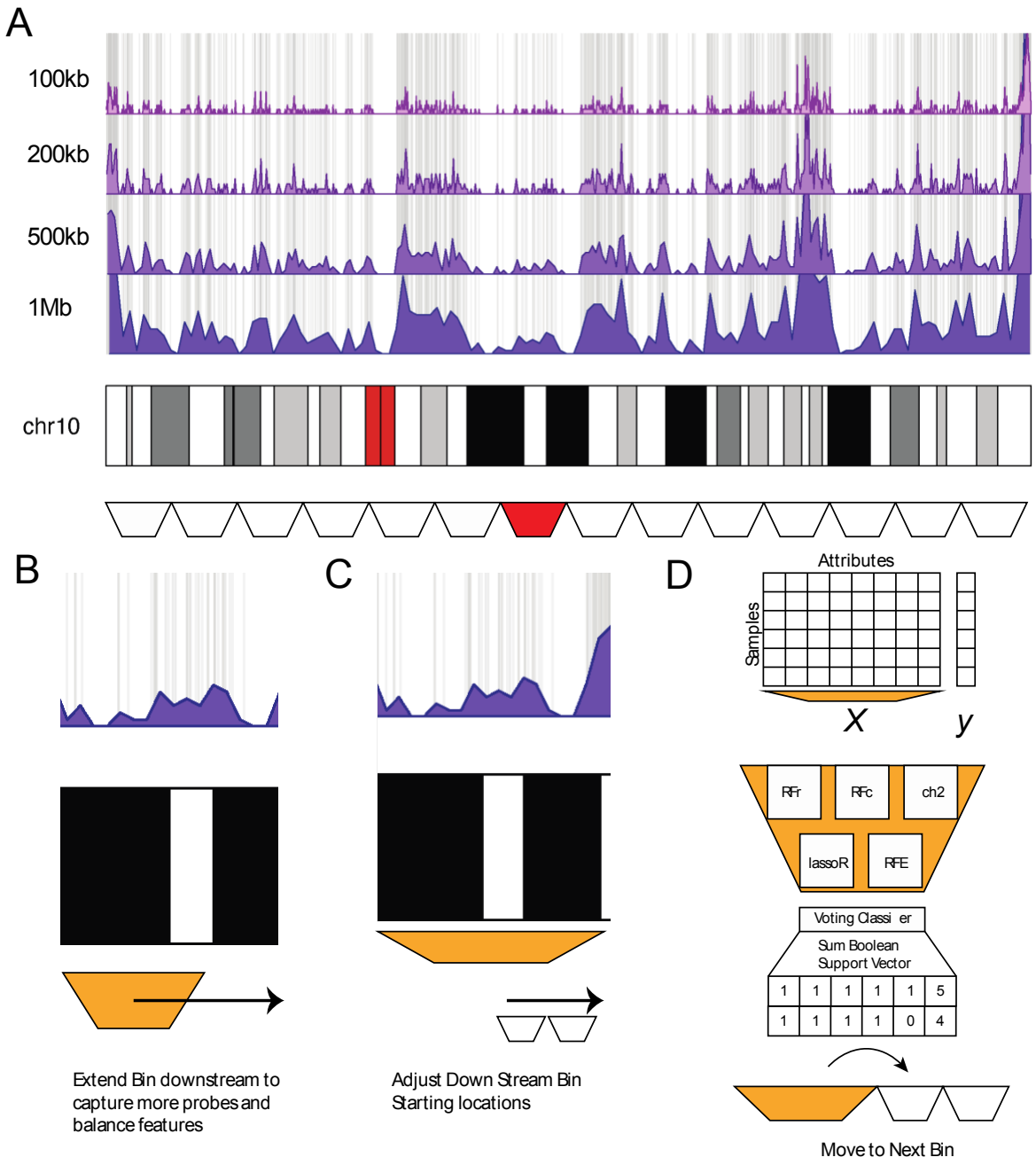

Figure 6. The density of Human Chromosome 10 CpG islands visualized using variable genomic window sizes of 100kb, 200kb, 500kb and 1MB. (A) Low density bins with very few features within a genomic window are dealt with by dynamically extending bin size requiring on the fly adjustment of downstream bin locations (B). Downstream bins starting locations are adjusted for non-overlapping windows (C). The features within the extended bin are assessed by sparce method and the sliding window moves forward (D).

Figure 7. Thin bin Dilemma, Merge bin solution

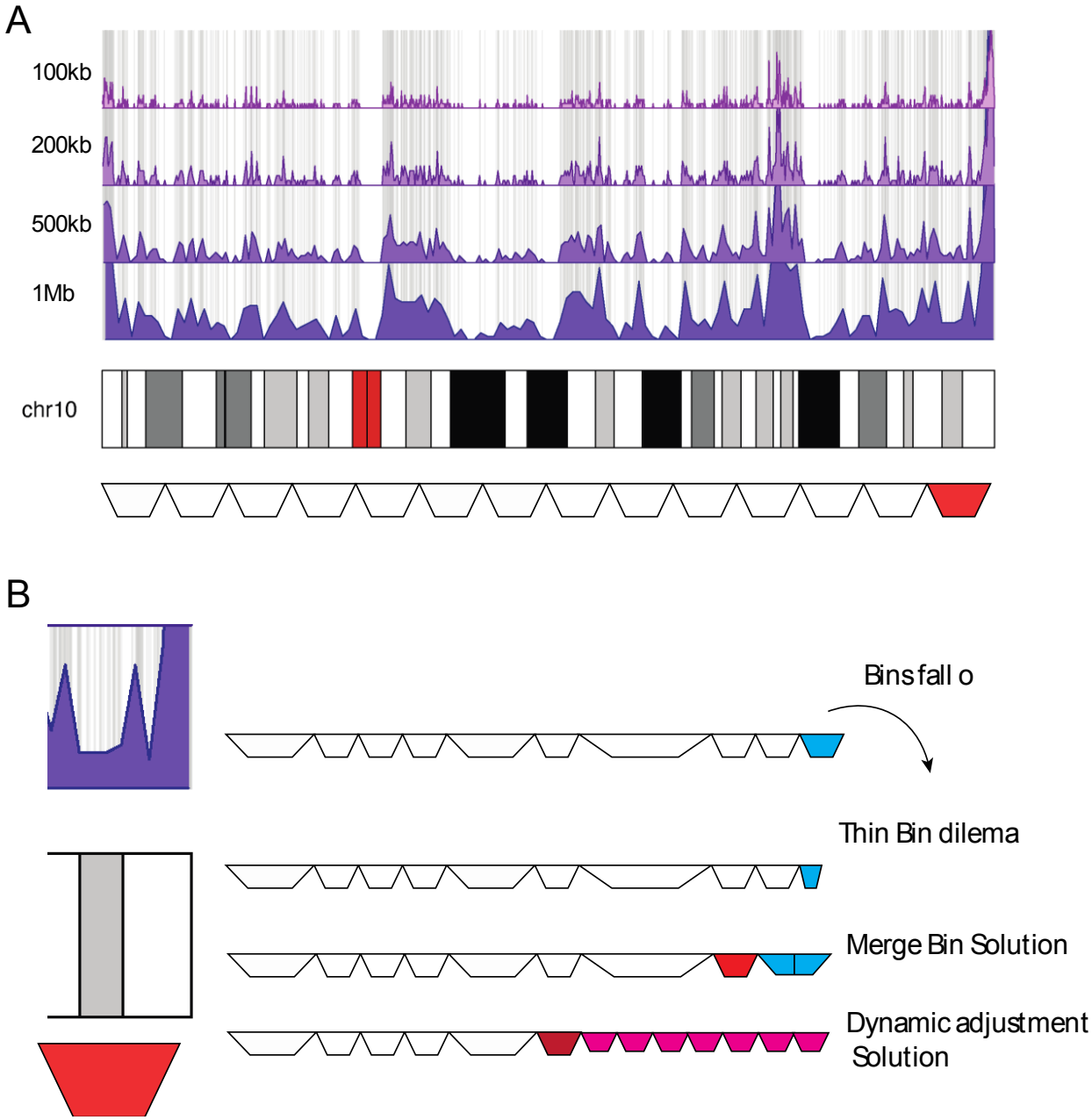

Figure 7. The density of Human Chromosome 10 CpG islands visualized using variable genomic window sizes of 100kb, 200kb, 500kb and 1MB. (A) A problem occurs when the remaining number of features on a chromosome are significantly smaller than the other bins. We use a look forward approach to solve this dilemma. SPARCE will look forward in the next two windows and count the number of remaining features in the remaining bins. If the remaining features in the last two bins is <1.5x the average number of features in the previous bins, SPARCE will merge the last two bins in the chromosome to resolve the thin bin dilemma. SPARCE offers an alternative solution using an on-the-fly dynamic adjustment method that can be invoked while binning features. The dynamic adjustment method will calculate the optimum bin size to reduce the features in the bin below the number of samples and limit the number of recursive mini-bin evaluation steps. The dynamic adjustment will make approximately equal sized bins and equally filled with the remaining features and genomic space within a chromosome or genome (B)

3.7 Bibliogrpahy

1 HAI, S. U. *Stanford University Artifical intelligence annual report*, <https://hai.stanford.edu/ai-index-2019> (2019).

2 Tibshirani, T. H. a. R. (2003).

3 Zhao, Y., Wong, L. & Goh, W. W. B. How to do quantile normalization correctly for gene expression data analyses. *Sci Rep* **10**, 15534, doi:10.1038/s41598-020-72664-6 (2020).

4 Saeys, Y., Inza, I. & Larranaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics* **23**, 2507-2517, doi:10.1093/bioinformatics/btm344 (2007).

5 Radovic, M., Ghalwash, M., Filipovic, N. & Obradovic, Z. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinformatics* **18**, 9, doi:10.1186/s12859-016-1423-9 (2017).

6 Perez-Riverol, Y., Kuhn, M., Vizcaino, J. A., Hitz, M. P. & Audain, E. Accurate and fast feature selection workflow for high-dimensional omics data. *PLoS One* **12**, e0189875, doi:10.1371/journal.pone.0189875 (2017).

7 Mohebbi, M., Ding, L., Malmberg, R. L. & Cai, L. 102-120 (Springer International Publishing).

8 Mirzal, A. 223-230 (Springer Singapore).

9 Minn, A. J. *et al.* Genes that mediate breast cancer metastasis to lung. *Nature* **436**, 518-524, doi:10.1038/nature03799 (2005).

10 Liu, J. X., Xu, Y., Zheng, C. H., Wang, Y. & Yang, J. Y. Characteristic gene selection via weighting principal components by singular values. *PLoS One* **7**, e38873, doi:10.1371/journal.pone.0038873 (2012).

11 Ding, C. & Peng, H. Minimum redundancy feature selection from microarray gene expression data. *J Bioinform Comput Biol* **3**, 185-205, doi:10.1142/s0219720005001004 (2005).

12 Skaro, M. *et al.* Are we there yet? A machine learning architecture to predict organotropic metastases. *BMC Med Genomics* **14**, 281, doi:10.1186/s12920-021-01122-7 (2021).

13 Kobak, D. & Berens, P. The art of using t-SNE for single-cell transcriptomics. *Nat Commun* **10**, 5416, doi:10.1038/s41467-019-13056-x (2019).

14 Bonilla Huerta, E., Duval, B. & Hao, J.-K. 250-261 (Springer Berlin Heidelberg).

15 Arun, A. S., Tepper, C. G. & Lam, K. S. Identification of integrin drug targets for 17 solid tumor types. *Oncotarget* **9**, 30146-30162, doi:10.18632/oncotarget.25731 (2018).

16 Yu, L. & Liu, H. Efficient Feature Selection via Analysis of Relevance and Redundancy. *J. Mach. Learn. Res.* **5**, 1205-1224 (2004).

17 Ye, X., Zhang, W. & Sakurai, T. Adaptive Unsupervised Feature Learning for Gene Signature Identification in Non-Small-Cell Lung Cancer. *IEEE Access* **8**, 154354-154362, doi:10.1109/access.2020.3018480 (2020).

18 Yuan, X., Gao, M., Bai, J. & Duan, J. SVSR: A Program to Simulate Structural Variations and Generate Sequencing Reads for Multiple Platforms. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **17**, 1082-1091, doi:10.1109/tcbb.2018.2876527 (2020).

19 Zhang, Y. *et al.* ELMO: An Efficient Logistic Regression-Based Multi-Omic Integrated Analysis Method for Breast Cancer Intrinsic Subtypes. *IEEE Access* **8**, 5121-5130, doi:10.1109/access.2019.2960373 (2020).

20 Liu, W., Li, D. & Han, H. Manifold Learning Analysis for Allele-Skewed DNA Modification SNPs for Psychiatric Disorders. *IEEE Access* **8**, 33023-33038, doi:10.1109/access.2020.2974292 (2020).

21 Okwori, M. & Eslami, A. Investigating the Impact of Gene Cofunctionality in Predicting Gene Mutations of E. coli. *IEEE Access* **8**, 167397-167410, doi:10.1109/access.2020.3023662 (2020).

22 Placzek, A., Pluciennik, A., Kotecka-Blicharz, A., Jarzab, M. & Mrozek, D. Bayesian Assessment of Diagnostic Strategy for a Thyroid Nodule Involving a Combination of Clinical Synthetic Features and Molecular Data. *IEEE Access* **8**, 175125-175139, doi:10.1109/access.2020.3026315 (2020).

23 Prabhakar, S. K. & Lee, S.-W. Transformation Based Tri-Level Feature Selection Approach Using Wavelets and Swarm Computing for Prostate Cancer Classification. *IEEE Access* **8**, 127462-127476, doi:10.1109/access.2020.3006197 (2020).

24 Shao, W. *et al.* Integrative Analysis of Pathological Images and Multi-Dimensional Genomic Data for Early-Stage Cancer Prognosis. *IEEE Transactions on Medical Imaging* **39**, 99-110, doi:10.1109/tmi.2019.2920608 (2020).

25 Wang, C. *et al.* A Cancer Survival Prediction Method Based on Graph Convolutional Network. *IEEE Transactions on NanoBioscience* **19**, 117-126, doi:10.1109/tnb.2019.2936398 (2020).

26 Wang, M., Huang, T.-Z., Fang, J., Calhoun, V. D. & Wang, Y.-P. Integration of Imaging (epi)Genomics Data for the Study of Schizophrenia Using Group Sparse Joint Nonnegative Matrix Factorization. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **17**, 1671-1681, doi:10.1109/tcbb.2019.2899568 (2020).

27 Capo, M., Perez, A. & Lozano, J. A. A Cheap Feature Selection Approach for the K-Means Algorithm. *IEEE Transactions on Neural Networks and Learning Systems* **32**, 2195-2208, doi:10.1109/tnnls.2020.3002576 (2021).

28 Elseddeq, N. G., Elghamrawy, S. M., Salem, M. M. & Eldesouky, A. I. A Selected Deep Learning Cancer Prediction Framework. *IEEE Access* **9**, 151476-151492, doi:10.1109/access.2021.3124889 (2021).

29 Huang, H.-H. & Liang, Y. A Novel Cox Proportional Hazards Model for High-Dimensional Genomic Data in Cancer Prognosis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **18**, 1821-1830, doi:10.1109/tcbb.2019.2961667 (2021).

30 Moon, K. R., Sricharan, K. & Hero, A. O. Ensemble Estimation of Generalized Mutual Information With Applications to Genomics. *IEEE Transactions on Information Theory* **67**, 5963-5996, doi:10.1109/tit.2021.3100108 (2021).

31 Mulenga, M., Kareem, S. A., Sabri, A. Q. M. & Seera, M. Stacking and Chaining of Normalization Methods in Deep Learning-Based Classification of Colorectal Cancer Using Gut Microbiome Data. *IEEE Access* **9**, 97296-97319, doi:10.1109/access.2021.3094529 (2021).

32 Qaraad, M. *et al.* A Hybrid Feature Selection Optimization Model for High Dimension Data Classification. *IEEE Access* **9**, 42884-42895, doi:10.1109/access.2021.3065341 (2021).

33 Raghu, V. K. *et al.* A Pipeline for Integrated Theory and Data-Driven Modeling of Biomedical Data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **18**, 811-822, doi:10.1109/tcbb.2020.3019237 (2021).

34 Spirko-Burns, L. & Devarajan, K. Supervised Dimension Reduction for Large-Scale "Omics" Data With Censored Survival Outcomes Under Possible Non-Proportional Hazards. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **18**, 2032-2044, doi:10.1109/tcbb.2020.2965934 (2021).

35 Clough, E. & Barrett, T. The Gene Expression Omnibus Database. *Methods Mol Biol* **1418**, 93-110, doi:10.1007/978-1-4939-3578-9_5 (2016).

36 Wenger, A. *et al.* Intratumor DNA methylation heterogeneity in glioblastoma: implications for DNA methylation-based classification. *Neuro Oncol* **21**, 616-627, doi:10.1093/neuonc/noz011 (2019).

37 Colaprico, A. *et al.* TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data. *Nucleic Acids Res* **44**, e71, doi:10.1093/nar/gkv1507 (2016).

38 Lawrence, M. *et al.* Software for computing and annotating genomic ranges. *PLoS Comput Biol* **9**, e1003118, doi:10.1371/journal.pcbi.1003118 (2013).

39 Maksimovic, J. (ed Murdoch Childrens Research institute) (2021).

40 Davis, S. & Meltzer, P. S. GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor. *Bioinformatics* **23**, 1846-1847, doi:10.1093/bioinformatics/btm254 (2007).

41 Johnson, R. B., Onwuegbuzie, A. J. & Turner, L. A. Toward a Definition of Mixed Methods Research. *Journal of Mixed Methods Research* **1**, 112-133, doi:10.1177/1558689806298224 (2007).

42 Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825-2830 (2011).

43 Hao, J. G. & Ho, T. K. Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language. *Journal of Educational and Behavioral Statistics* **44**, 348-361, doi:10.3102/1076998619832248 (2019).

44 Abadi, M. *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv:1603.04467 (2016). <https://ui.adsabs.harvard.edu/abs/2016arXiv160304467A>.

45 Martello, S., Pisinger, D. & Toth, P. Dynamic Programming and Strong Bounds for the 0-1 Knapsack Problem. *Management Science* **45**, 414-424 (1999).

46 Rizvi, S., Khan, S. A., Hallemeier, C. L., Kelley, R. K. & Gores, G. J. Cholangiocarcinoma - evolving concepts and therapeutic strategies. *Nat Rev Clin Oncol* **15**, 95-111, doi:10.1038/nrclinonc.2017.157 (2018).

47 Vaske, C. J. *et al.* Inference of patient-specific pathway activities from multi-dimensional cancer genomics data using PARADIGM. *Bioinformatics* **26**, i237-245, doi:10.1093/bioinformatics/btq182 (2010).

48 Verhaak, R. G. *et al.* Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1. *Cancer Cell* **17**, 98-110, doi:10.1016/j.ccr.2009.12.020 (2010).

49 Paterson, A. H. *et al.* The Sorghum bicolor genome and the diversification of grasses. *Nature* **457**, 551-556, doi:10.1038/nature07723 (2009).

50 Cooper, E. A. *et al.* A new reference genome for Sorghum bicolor reveals high levels of sequence similarity between sweet and grain genotypes: implications for the genetics of sugar metabolism. *BMC Genomics* **20**, 420, doi:10.1186/s12864-019-5734-x (2019).

CHAPTER 4

MYCORRHISEE: A WEBTOOL AND COMPUTER VISION SUITE FOR SEGMENTATION AND SCORING OF GRASS ROOTS COLONIZED BY MYCORRHIZAL FUNGI

[1] Michael Skaro, Yue Wu, Shufan Zhang, Jonathan Arnold. To be Submitted. Oxford Journal of Bioinformatics.

4.1 Abstract:

Background: Mycorrhizal species form intricate hyphal networks below the surface of the ground in the rhizosphere of 80% of terrestrial plants. In symbiotic cases, this relationship results in improved acquisition of nutrients (e.g., phosphate and nitrates) for the plant. In return, the plant provides fixed carbon (e.g., sugars and fatty acids) to the fungus. Characterizing and quantifying this relationship is currently performed by manual curation of fungal structures, classification, and hand counting. This process is slow, arduous, and not particularly reproducible. However, using computer vision, we can automate this process and complete the microscopic characterization using high throughput methods and a reproducible model. Methods: We deployed an instance segmentation model that captures and estimates the colonization percentage of AM fungi and its associated structures in root images. Results: We have deployed the model as a readily available webtool for image segmentation and structure annotation. Conclusions: This model builds on previously published frameworks to provide the fungal community with a new image segmentation tool for structure annotation for images generated in the lab and in the field.

4.2 Introduction:

Mycorrhizae are fungi that develop symbiotic associations with terrestrial plants. The co-evolution of terrestrial plants and mycorrhizal fungi is estimated to date back as far as the first plant root systems, nearly 500 million years ago[1]. Mycorrhizae are obligate autotrophs that form complex integrated hyphal networks to exchange nutrients with their terrestrial host[1]. The exchange results in improved acquisition of nutrients for the plants (e.g., phosphate and nitrates) and, in return, the plant provides photosynthetically fixed carbon (such as carbohydrates and fatty acids) to the fungi[2]. Networks of fungal hyphae form inside and outside the plant root systems. Arbuscular mycorrhizae (AM) are the most common and well characterized mycorrhizae in agricultural and natural ecosystems. These fungi provide water and minerals to the plant by penetrating root cells within the root cortex[3]. External mycorrhizae, broadly categorized as ectomycorrhizae (EM), are less common, interacting with about 10% of terrestrial plants[2]. EMs form hartig network sheaths around the outside of the cells where the hyphae penetrate the epidermis but do not extensively ramify inside the root and are common with dipterocarp, eucalyptus, oak, pine, and rose families of plants. Mycorrhizal mycelia of both AM and EM species extend from one plant's roots to another plant's roots to form common mycorrhizal networks (CMNs) in multiple species and genera[3]. CMNs facilitate nutrient flux between groups and increase uptake for individual organisms[3]. Harnessing this relationship of mycorrhizal fungi in agricultural cultivars could be an essential step for engineering more productive and more resilient crop varieties in the face of climate change[3].

To improve agriculture globally, AM and EM utilization provide the opportunity to leverage these symbiotic relationship to minimize artificial inputs (like fertilizer and water application) to agricultural crops[4-7]. Symbiotic mycorrhizal fungi have been shown to increase

plant drought tolerance and disease/pest resistance[5,8]. *Sorghum bicolor*, an agricultural staple in hot and drought-prone regions of the world, shows increased drought tolerance and significant resistance to *Striga hermonthica*, a devastating parasitic plant native to sub-Saharan Africa[9], when colonized by AM fungi. There is now significant evidence that the terrestrial host plays a crucial role in establishing and regulating its root-associated microbial communities[10-13]. A critical step towards understanding how the plant manages its root residents is the development of computational tools to characterize the resident species, including their structural arrangements[10,14-16].

Research and development in the field of Deep Learning, specifically the sub-specialty of computer vision, has exploded in popularity[17]. Computer vision can be used for classification, semantic and instance segmentation, object detection, form recognition, and much more[17]. The applications of computer vision have been particulary powerful in projects focusing on facial recognition, crowd detection, human abnormal behavior detection, illegal parking detection, speeding vehicle detection, self-driving enablement, cell biological analysis, and digital pathology detection[18-23]. There are new disciplines emerging at the intersection of agroeconomics and field robotics. One such study explored the use of automated classification for plant diagnosis, including real time tissue pathology diagnosis, using a field-scanning robot[14]. Leveraging the most cutting-edge machine learning techniques in biological research studies that explore instance segmentation for object detection are driving model development in computer vision research[24-26].

Object detection is a two-step process in which the computer both classifies and identifies the locations of objects in an image. These tools often require thousands of training instances to effectively categorize the objects in a target image. Generating objective-specific, deep and diverse data sets for every application can be costly and time consuming []. An alternative approach

leverages transfer learning from pre-trained models. Transfer learning is a process by which inductively learned knowledge is inherited from a pre-configured architecture and pretrained network[26,27]. The model backbone is downloaded from a publicly available database and the final layers are either frozen or fine-tuned to classify, locate and estimate the target values in the new dataset[26]. Transfer learning, particularly in the spaces of natural language processing, image classification and object detection, has been demonstrated to achieve excellent performance even in cases where the new targets do not occur in the original training sets[28].

There is currently a paucity of research that has developed computer vision tools for the characterization of soil mycorrhizae in their colonization of roots. The first step forward in this field was taken by Evangelisti *et al.* in the development of AMFinder[10,15,16]. This tool employs two convolutional neural networks (CNNs) for image tile classification. Thirty images were broken into 90x20 and 90x40 (column x row) tiles to train the CNN1 and CNN2 models, respectively[15]. The team employed tile level classification to estimate root colonization of AM fungi and characterize fungal structures[10]. While the tiling classification approach shows strong results in image classification, the tiling approach does not accurately estimate colonization percentage of fungal structures and, by design, compounds its error during the calculation. This method counts the image tiles that contain fungal structures and divides that number by the number of tiles that contain roots. It does not calculate an estimated percentage of the root area that is colonized by the AM fungi. This method suffers from the same bias as the McGonigle method, that model error exponentially compounds for every tile that a fungal structure concurrently inhabits in the image[29]. Further, this method suffers from underestimation errors if two or more structures are completely encapsulated in a tile classified as, 1, colonized by AM fungi[29]. We have made several significant improvements on this method. Our method improves on the AMFinder research tool in several key

areas. First, accurately segmenting the colonization areas using an object detection architecture, Detectron2 deploying Second, colonization estimation error. AMFinder does not calculate a colonization percentage using the area of the root or area of the fungal structures. Our tool is demonstrated to accurately quantify the root area colonized by the segmented fungal structures compared to hand annotation. Finally, our workflow is deployed as an easily accessible webtool powered by Amazon Web services.

4.3 Methods:

Plant cultivation:

Accessions were derived from a mapping population consisting of 191 RILS F3:5 from a cross between *Sorghum propinquum* and inbred line TX7000 of *S. bicolor*. Three seeds from each accession were planted on October 5, 2020 in UGA pine bark mix in 2.5 gallon pots in the UGA Botany Greenhouse. Seedlings grown on a 11 hour (h) light cycle were transferred to a 4:1 mix of UGA pine bark mix and soil from Ironhorse Farm, GA. (Table 1) in individual 2.5 gallon pots on October 20, 2020 by. Seedlings were grown to maturity on a 11 hour Light/Dark cycle with watering as needed every 4 h. Plants were fertilized with 1 tablespoon Osmocote. In addition, we harvested roots of one commercial hybrid forage sorghum plant from Richardson, TX that was being used as a pollen barrier (Crop and Soil Science, UGA) in a switchgrass nursery at Iron Horse Farm on October 13, 2020. We also harvested roots of one commercial accession (#101 in 414 sorghum forage) courtesy of Dr. Mailhot (Crop and Soil Science, UGA) on November 12, 2020 at Iron Horse Farm, GA.

Root cutting:

One half of each root sample from the field was investigated to find the best roots for AMF visualization (small diameter, lateral roots on main ones, intact cortex). A small aliquot of these roots (~0.25g) was frozen and stored at -20◦C until use.

Staining roots with ink and vinegar for quantification of AM colonization:

Hydrated roots were washed with tap water and then transferred to Simport biopsy cassettes (Fisher Scientific Co. Pittsburgh PA, USA catalog #15-182-700). Roots were cleared by soaking them in 2.5% KOH at room temperature overnight. Alternatively, a user may soak for 30 minutes in 2.5% KOH at 90^{o} C until solution is light brown to get the same results. KOH was neutralized

by adding vinegar to the KOH solution until it reaches a pH of 6-7 (pH paper stains green). Cassettes were washed once in tap water. Cassettes were boiled together in 1L of 5% ink-acetic acid solution under the hood for 5 minutes. Cassettes were moved to a new 1L beaker and iteratively washed with tap water at room temperature. Finally, cassettes were added to a clean 1L beaker of water with 10 drops of XX% acetic acid and soaked overnight.

Root imaging:

To phenotype AMF colonization, 1.25g of fine roots were randomly excised from the ink-stained roots described above. Samples were stored in 70% ethanol at 4^{o}C. Root samples were cleared in 10% alkaline H¬2O2 for 2 hours and in 5% KOH overnight at room temperature. The stained roots were straightened and flattened onto glass slides prior to imaging. Mounted root samples were imaged at 200X magnification with a Zeiss Primo Star microscope equipped with an Axicam 105 color camera. One hundred ninety-two fields of view were scanned on every 0.25g of root sample. Fields of view were 0.5cm equidistantly spread across 75 x 25mm glass slides.

Public data mining:

Publicly available data deposited on the zenodo data portal from Evangelisti *et al.*[15] were downloaded for side-by-side model comparison and our model development. The zenodo-get software method was used on the zenodo ID 10.5281/zenodo.5118948 to retrieve the dataset[30,31]. The jpg images were added to the existing dataset for training and testing. The Evangelisti *et al.*[15] annotations were not used because they were not applicable to our approach.

Root image annotation:

Root image annotation was conducted using the VGG annotator tool. The fungal structures were annotated using the polyline option. Classes identified in the image were circled on the image, and the points were exported in JSON format. Each annotated image was binned into the training,

testing or validation bins. The image annotations were combined into conglomerate training, testing and validation data annotations files. The file structures for images and the associated annotations followed the COCO format as directed by the detectron2 documentations.

Root Image Augmentation:

Image augmentation was conducted using the albumentations package in python. Images were rotated, flipped, and modified using contrast and zoom manipulations. The image augmentations diversified the training set for better model performance. Hand annotated image segmentations were analyzed to produce segment bounding boxes. Each bounding box was cropped from the image as an independent tile. Each of the bounding boxes were modeled as a convex hull. A convex hull C is represented as a sorted series of cartesian coordinates $[\{\{x_1,y_1\},\{x_2,y_2\},\ldots\}]$ where c[0][0] is the xmin and the ymin of the bounding box. Given that the hull is a simple four-sided polygon, the four vertices are used to find the geometric center of each bounding box. Each list of segmentation points annotated in the image was passed to the Raycasting algorithm to find out if the point is in the convex hull. The algorithm followed the even-odd-rule to calculate if a point is in each polygon. The objective function for this rule is if a point crosses an odd number of sides of the hull to connect the geometric center of the polygon, the the point was judged as outside the box. In contrast, if the point crossed an even number of sides, is a vertex point, or was along an edge; the point was judged as inside the polygon. This search runs in O(n)(p), where n is the number of edges of the polygon and p is the number of points annotated in the image.

In the event that a second structure was inside the bounding box, we added an optional contextual augmentation. The context of the segment may be critical for determining instance class. Adding segments inside the bounding box patch that was cropped for each structure allowed

the algorithm to see instances of objects annotated in context of other classes in the dataset. The segmentations were mapped to the intersecting regions in the tiles. Tiles were added to the training, validation, and testing sets. An approximately equal split of field generated images and datamined images were sent to the training, validation, and test sets. All image augmentations are available in the train.py file provided in the repository.

Network training, tuning and evaluation:

The model was trained on the Georgia advanced computing resource center (GACRC). We implemented the automatic method to segment and annotate fungi and root structures through Mask R-CNN[24,32]. A total of 746 images and 3577 polygon segmentations were used for training, testing and validation with approximate ratio 8:1:1. After merging classes with few instances into 'others', the following classes were used: root, AMF internal hypha, AMF external hypha, AMF arbuscule, AMF vesicle, AMF spore, and others.

The Mask R-CNN model was implemented in Detectron2 and is composed of the backbone, the region proposal network (RPN), and heads[24,32]. The ResNet 50 and FPN (Feature Pyramid Network) backbone extracts feature maps from images[24,32]. RPN proposes candidate regions. Heads produce bounding box, mask, and class inferences. The Mask R- CNN model was pretrained on the COCO dataset with 3x schedule[21,33]. Based on the pretrained model, we continued training for 50 epochs on microscopic images with batch size 2 and the default learning rate schedule. For each image, the best fine-tuned model was selected based on total loss in validation set during training. Different hyperparameters were tested and each was repeated three times with different random seeds. We tested two learning rates: 0.001 and 0.002. We varied the number of frozen or fine-tuned backbone modules, where the 'FREEZE_AT' parameter ranged from 1 to 3. Two augmentation options were implemented: the default option and a more complex

one. The default option includes image random flip and resize, and the latter option adds random crop, rotation, and brightness adjustment. Other parameters were set as default in Detectron2 configuration[24]. We evaluated model performance and selected hyperparameters based on mean Average Precision (mAP). This metric was calculated with varying confidence thresholds and averaged over all classes. In addition, AP50 was calculated at Intersection over Union (IoU) level 50% and AP was averaged over IoU levels from 50% to 95%. Score threshold for inference in test set was set at 0.7. Model training and inference was implemented on sapelo2 at GACRC with one p100 GPU, 4 CPUs, and 20 GB memory. Evaluation code and parameter tuning codes are publicly available in the GitHub repository.

Web application:

The web-application was implemented in the Amazon Web-Services ecosystem. The website is built in HTML5, CSS3, Bootstap CSS and Vue JS.3. The web interface is comprised of four sections. The Home section describes the tools and functionality on the web-portal. The Image Segmentation section is a drag and drop box for image submission into the model. The third section is a description of the team and our most recent news. The fourth section allows the users to get in touch with the labs that work on the project. The website is hosted using the Route 53 hosting service. The hosting service called Amazon Route 53 is a scalable cloud Domain Name System (DNS) web service[34]. Amazon Route 53 serves to connect the web-UI in the model requests infrastructure. Route 53 communicates through the API Gateway directly into an Amazon S3 bucket using a lambda function for pre-signing a submitted image URL for submission directly on a temporary landing bucket[34]. The image submission is handled internally with a series of Amazon S3 triggers. The image deposition triggers a second lambda function to move the image into a staging area and run the MycorrhiSEE model[34]. The model is run on the Amazon Sagemaker studio

inside the Detectron2 docker that is configurable inside the ecosystem[34]. The image that was evaluated and handled by the model directs the segmented image back to the S3 bucket staging area to trigger a final lambda function to return the data back to the user[34]. The model shuts down the docker container. The model is deployed on cloud front to deploy on all domains in the US, Europe and Canada. The use of the model is currently restricted to top registered users while the model is still in active development. The workflow is implemented as a serverless workflow. The data is not saved or retained in the workflow. The maximum number of S3 requests per month is currently 2000[34].

4.4 Results

Fifteen plants were harvested from plants cultivated in the soil from the iron horse farm in Athens, GA. 746 images were produced from field plants. The field sourced images were supplemented with 15 high resolution images from the Evanglasti et al 2020 study from the zenodo public database and were added to the field sourced image repository[15,30,31]. In Figure 1. we show the total population of fungal and root structures labeled in the images from the field and from the public database. There are six classes of information labeled in the images that were summed and displayed in the bar plots. The blue bars represent the structures that are labeled form the Evanglasti et al study and the green bars represent the structures labeled in the images coming from field sourced slides. The Evanglasti et al study produced over one-thousand structures per slide with an average of 821 AMF arbuscules and 732 AMF vesicles per slide. In contrast the field sourced data produced a total of 337 AMF spores and 137 AMF vesicles total. The mined data contained 3014 AMF internal hypha compared to 406 from the field sourced images. The mined data contained 1680 external hyphae compared to 976 in the field sourced data. The root structures in the field source were 1308 and 390 in the data mined sourced data. The total sum from each source for each structure is reported next to the bar.

In Figure 2. we display a cartoon overview of the model development and workflow. The images were annotated with the VGG annotator software (Figure 2A). Annotated images were padded to square the image (Figure 2A). The padded images were tiled, and region coordinates of the segmented structures were annotated. The annotated fungal structure coordinates were intersected with the tile bounding box coordinates (Figure 2B). The bounding box points were modeled as a convex hull. We used the Raycasting algorithm to calculate whether a point is in a given polygon. The Raycasting algorithm sites performs the even-odd-rule to find out whether a

point is in a given polygon. For of the segmented annotations in each image the annotated regions were intersected with image tiles and categorized with class labels (Figure 2C). A pretrained Detectron2 model built on top of the mask-rcnn recurrent convolutional neural network was tuned for the classification and segmentation of fungal structures in the validation dataset (Figure 2D). The test images tiles were segmented by the model and the model efficiency was evaluated (Figure 2E). The annotate image was returned with polygons annotated on top of the original image (Figure 2F).

In Figure 3. We display examples segmented image from the test dataset. The figure is broken into three columns and two rows. The two rows are represent two sections; the top row shows the annotated ground truth information, and the bottom row displays the accompanied model's prediction of the segmented areas in the image (Figure 3A, B, C). across the columns, we present three examples of three separate instance segmentation tasks from the test image dataset. In Figure 3A. The ground truth root was segmented into two sections of root. In the predicted section, MycorrhiSEE was able to discern the root section was continuous. The model was 99% confident in the root location and area. Interestingly, the model merged a continuous section of root that was labeled into two distinct areas. The model identified and found the critical structure AMF internal hypha. The model was 74% confident in the area around the fungal structure. This demonstrated the capacity of the model to calculate the area of key biological features for mycology research (Figure 3A). In Figure 3B the model segments the root, AMF Root hypha and AMF arbuscule class (Figure 3B). The model was 97% confident of the AMF internal hypha and 91% AMF arbuscule. The size and density of the colonization in the roots of the Sorghum bicolor is known to be is directly linked to the efficiency nutrient transfer in the soil (Figure 3B). In Figure 2C we show the instance segmentation architecture identifying side-by-side structures with an

ambiguous class labeling, the model captures all four instances of hypha in the image and discerns between the internal and external hypha in a side-by-side segmentation task (Figure 3C).

The model was configured and added as a free to use service deployed completely on the Amazon web services platform (aws). The model is hosted using the Route-53 website hosting service at www.Mycorrhisee.com (Figure4A). The website is static with one function area for image submission. Images submitted with into the MycorrhiSEE model are directly deposited on an s3 bucket using the serverless pre-signed URL and API gateway for an S3 container (Figure 3B). The S3 bucket uses an s3 trigger to enable an aws lamda function to trigger the Amazon Sagemaker. Amazon Sagemaker triggers a dockerfile to activate a virtual environment inside a docker container. The model evaluation script reads the submitted images and segments the images. The images are returned to the user inside the browser. A comma separated file with the annotated classes and the image coordinates is returned to the user. This file contains the information necessary for analyzing underlying biological patterns.

The model was evaluated using the three-standard metrics for an instance segmentation algorithm; Average precision (AP) of each structure class in the image, the Bounding Box average precision, and the Segmentation average precision.

4.5 Discussion:

The data population from the field sourced and the publicly sourced data in the labeled population showed different distributions in nearly all structures and orientations of the structures in the dataset. The difference in the prevalence of the root class is explained by the difference in the mechanism of sampling. The field sourced data were disjointed images captured in independent runs in contrast to a single image that was processed into sub-image tiles. However, the differences in the populations of fungal structures are not due to sampling but is directly linked to the mechanism of data production (Figure 1.). In the Evanglasti et al project, the authors inoculated the roots of 30 plants with AMF fungi. The inoculation and subsequent cell culture produced densely populated images that were rich with AM Fungi, AM Arbuscule and AM Internal hypha (Figure 1). Field sourced data was harvested from plants grown in the soil extracted from the iron horse farm. The samples showed a distinctly higher proportion of AM Spores compared to the cultured data mined from the Evanglasti et al. data (Figure 1). We considered the differences in the distribution of the structures in the two data sets may contribute to loss in the model in samples that are submitted from only the field or only the lab. To address this concern, we trained the model on an approximately equal distribution of the data from each resource. Further, the data from the field and the data from the lab were generated with a different background. The field sourced microscope images background is darker than the lab cultured samples. The differences in the background may contribute to Bounding Box regression loss at the structure boundaries. This was of particular concern for the root, and hyphae structures. The root edges in the field sourced data were darker and thicker. This difference is due to the difference in the root source. The root source in the field sourced data was the Sorghum bicolor, in contrast the root source of the mined data was from a collection of species; *icotiana benthamiana, Medicago truncatula, Lotus japonicus,*

Oryza sativa. The AMF internal hypha and AMF External hypha cross the background and root boundary. The mixed dataset allowed the model a broad diversity of object boundaries for the model to consider in its training.

The image orientations of the data sourced from the cultured samples were almost exclusively horizontal across the images. In contrast the images in the field sourced images were depicted in many orientations. We addressed this issue in the image augmentations in training. We added several image augmentations including; Resize Shortest Edge, Random Brightness, Random Flip, Random Rotation. We added an independent Random Crop to the images sourced from the Evanglasti team as independent samplings from the dataset. This step increased the representation of data from the mined data by over one hundred times instead of training on only the thirty publicly available images. These steps standardized the inputs of the images into the model for training.

The process of the model training was directly linked to the standardization of the image data. The class structures in the field sourced data and mined data were hand annotated using the VGG annotation software (Figure 2A). The points in each segmentation and the images used in the model were formatted into the COCO data set format[33]. Images were cropped into bounding box tiles modeled as a convex hull (2B). The points of the convex hull represented a simple polygon. The raycasting algorithm was used to test if the points in the segmentation were inside the bounding box of the image tile (Figure 2C). The tiles were output as individual images with an accompanied COCO dataset describing the tiles and segmentations (Figure 2D). Adding co-occurring vaues to the bounding box image sent to detectron2 adds biological information to the model and helped us to resolve loss. A specific example arises when a vesicle is identified in a

image that is not on a root. We can remove this classification and segmentation and identify the images in the training or validation set that leading to the erroneous calculations.

The data was fed into the Detectron2 instance segmentation model. The model was trained to identify the custom structures in the microscope images. These structures are not in the pretrained architecture from the COCO dataset that is now comprised of over 300,000 images and 1.5 million total structures primarily depicting in-home attributes, animals, urban living, and natural scenes[33]. The model inherits the inductively learned values from a previously trained architecture and identifies the structures in the microscope images using inductive transfer while tuning the model for the identification of the labeled structures in our custom dataset (Figure 2E). The trained model was deployed on the validation set to reconstruct the original images with annotated masks (Figure 2F).

The three example segmented images depicted in Figure 3 each demonstrate the use and applicability of the MycorrhiSEE model in the mycology research domain. The figure is broken into two sections: the ground truth and predicted masks. In the top row we show the hand annotated ground truth of the field sourced images. In Figure 3A. A root section was labeled with two polygons and an AMF internal hypha was labeled in the right section of the image. The model correctly identified and merged the root area with 99% confidence. The model identified the outline of the AMF internal hypha with a confidence 74%. In Figure 3B. The model identifies the root bounding box and AMF Arbuscule with 99% and 91% confidence. The identification of the AMF Arbuscule was a strong result in that this structure is not highly represented in the field sourced images demonstrating the advantage of mixing the field and cultured datasets. In Figure 3 C. two ambiguous AMF internal Hyphae and AMF External Hyphae were correctly distinguished

on both sides of the image. The model identified was able to discern the slight differences in the fungal hue that is only discernable in a z-stack in human labeling.

The was evaluated on the area of precision (AP) and the bounding box area estimation. In Figure 4. We show a side-by-side estimate of the train and validation loss in the classification and bounding box regression loss over the model epochs. The model loss was minimized in the training to 0.

MycorrhiSEE is deployed on a hosted zone on amazon webservices. We used the route 53 service to publicly host the online portal of our model. A deposited image is marked with a pre-signed URL and deposited into a temporary holding bucket. The deposition of the image in the S3 bucket triggers a cascade of internal aws services programmed around the model. The image is transferred to a staging bucket and a successive s3 trigger engages a model built using the aws Sagemaker studio. A lamda trigger invokes the model to evaluate the image, segment the structures within the image and return the segmented image back to the user. Once the model has returned the output the serverless workflow is shutdown. The entire workflow is deployed on cloudfront such that any user is capable of deploying this workflow seamlessly from their workstation.

4.7 Conclusions:

The MycorrhiSEE model demonstrates a significant advancement from the seminal work published by the Evanglasti et al team. Our model has improved upon their work in two fundamental spaces. First, the model is an instance segmentation model in contrast to tiling and classification model. Our approach identifies the pixels in the image that are depicting the structures and therefore is a real representation of the colonization area in contrast to a tile occupation and conversion. Second, our model is accessible and deployable. The AMFinder software requires field specific expertise, an in house HPC and domain knowledge of deep learning to implement. Our model requires none of these and only requires a user to submit the images into the MycorrhiSEE image drag and drop box.

The next steps for this model are to return colonization and structure counts in an annotations file back to the end user. This service would increase the user engagement with the model and ultimately increase the impact for the mycology researcher.

Special thanks and acknowledgments:
We would like to thank Dr. Mooney and Dr. Mailhot at Texas A&M and UGA Crop and soil sciences for their contribution of data resources in this project. We would like to acknowledge the significant efforts of the undergraduate students that individually labeled the sizeable dataset generated in this project. Lauren Stupp, William Lantz, Camryn Rebecca Felt, and Brooke Lincoln; Thank you for your time and contributions to this project.

4.6 Figures and Tables:

Figure 1. Annotated data populations.

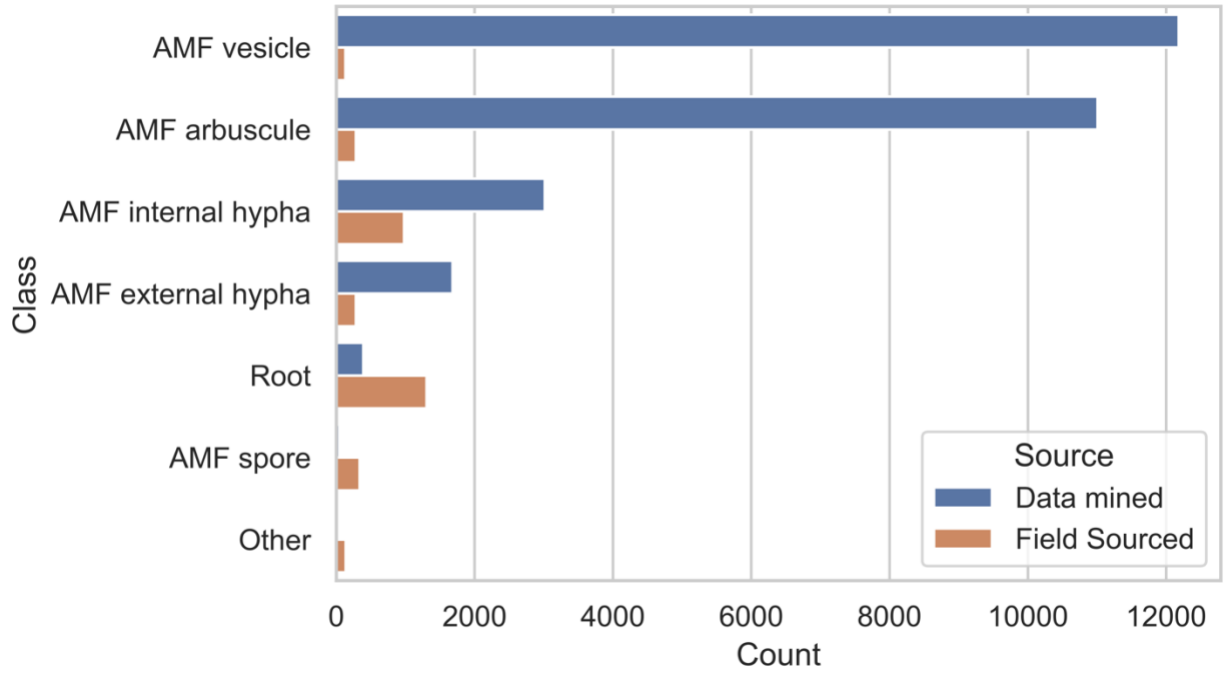

Figure 1. Six classes captured in microscope image were labeled using the VGG annotation software. The classes are organized top to bottom in order of their total count on the vertical axis. The data population is plotted across the horizontal axis. The source of the image class is colored in blue for data sourced from the Zenodo database and green for data that was sourced from harvest plants from the iron horse field. The structure classes are ordered by their prevalence in the data mined source order.

Figure 2. MycorrhiSEE model flowchart

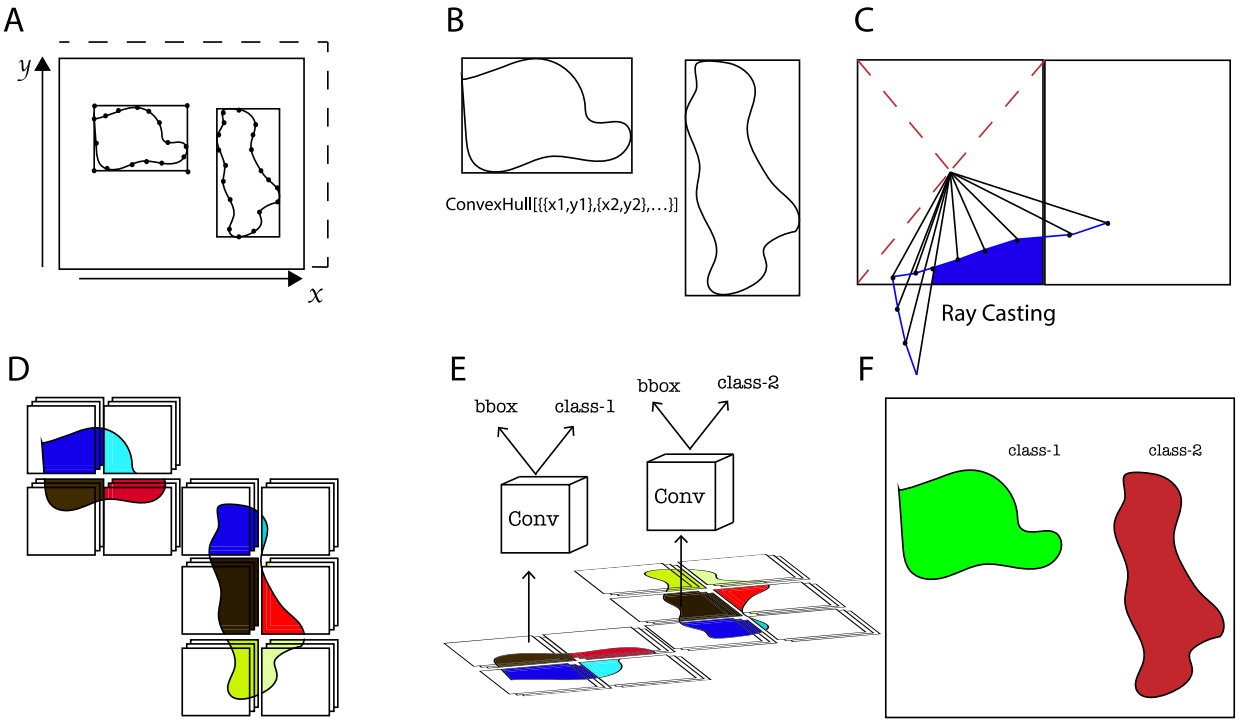

Figure 2. Images of fixed roots were annotated with the VGG annotator software. The images were padded in the x and y direction to square the images (A). The annotated images were tiled into bounding box patches from the hand annotated polygon points. The bounding boxes were modeled as convex hulls intersected with segmentation coordinates(B). The ray casting algorithm implemented with the even-odd-rule was used to determine if other segmented points were in the tile (C). Points in each tile were translated from image coordinates to tile coordinates categorized with the class ID (D). Tiles containing segmented structures were used as the input to tune the pretrained Detectron2 deep neural network (E). The trained neural network segments instances of annotated structures in the test images (F).

Figure 3. Image Segmentation of root images

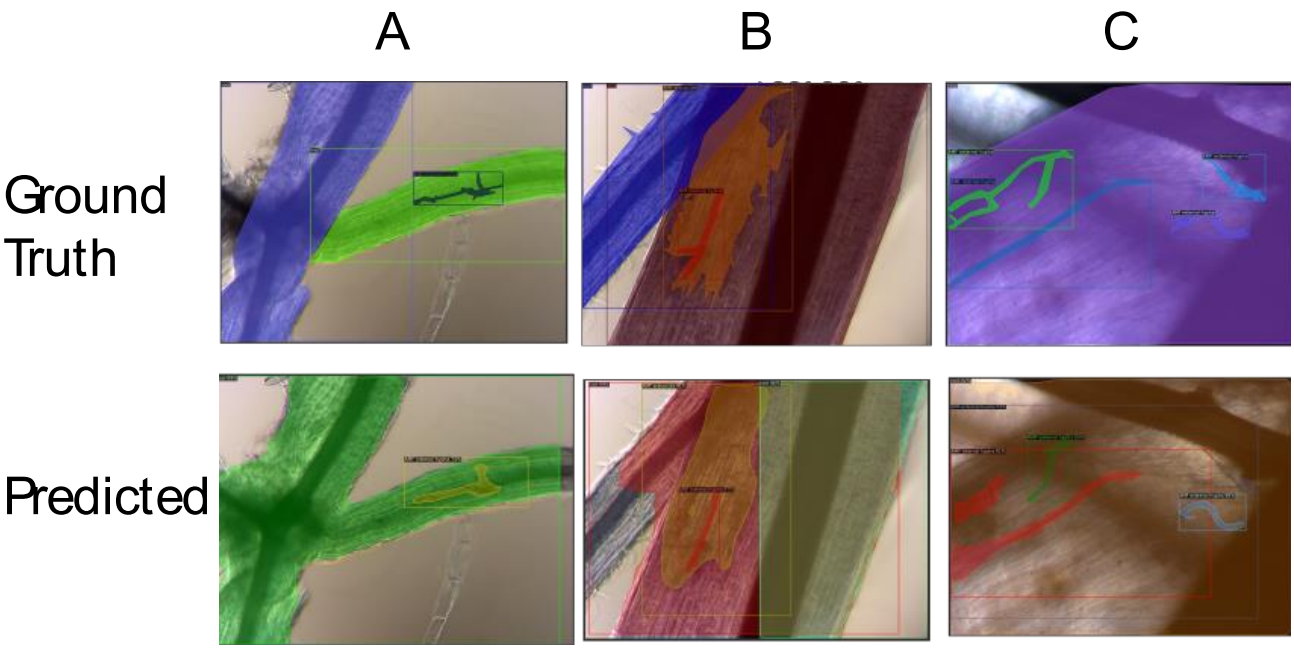

Figure 3. Three groups of example tiles are displayed in three columns and two rows. The columns show three examples of the image segmentation model processing the slide images of the roots. The rows of the figure display the Ground truth segmentation labeling in the top row and the predicted polygons in the bottom row.

Figure 4. Amazon webservices workflow

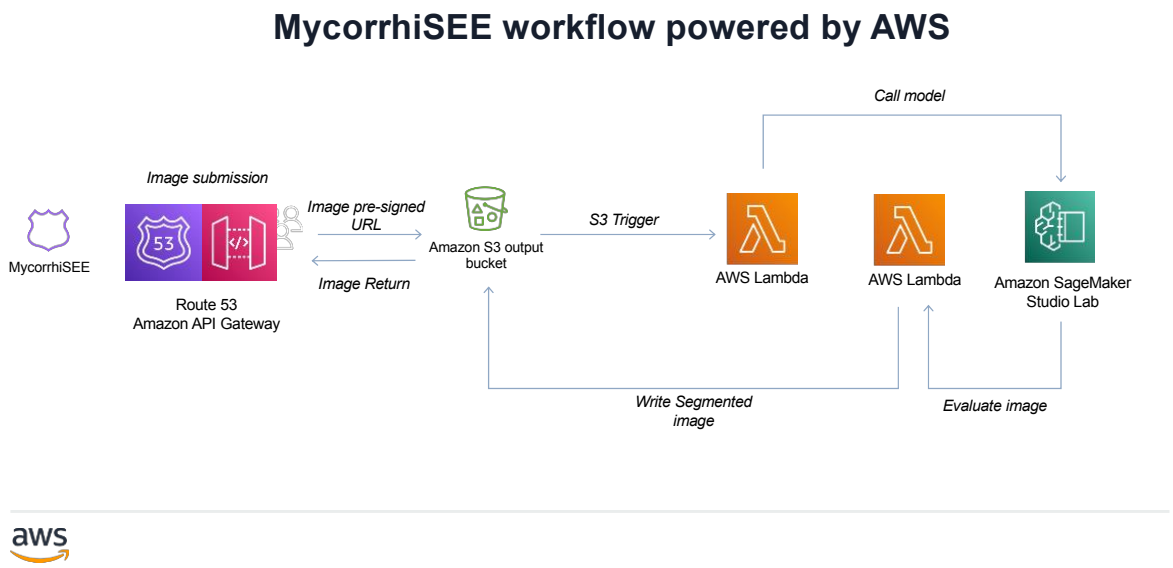

Figure 4. The MycorrhiSEE architecture is hosted on the Route 53 platform. An image is transferred for the user to the simple storage solution bucket. An S3 trigger activates an aws lamda function to activate the docker container and virtual machine to run the MycorrhiSEE model. The configured and trained model runs on the Amazon Sagemaker studio lab and segments the image. The segmented image is saved and an aws lamda function is returned to s3 bucket. The user request is returned to the user in the browser and the request loop is closed.

Figure 5. Model Evaluation of the data

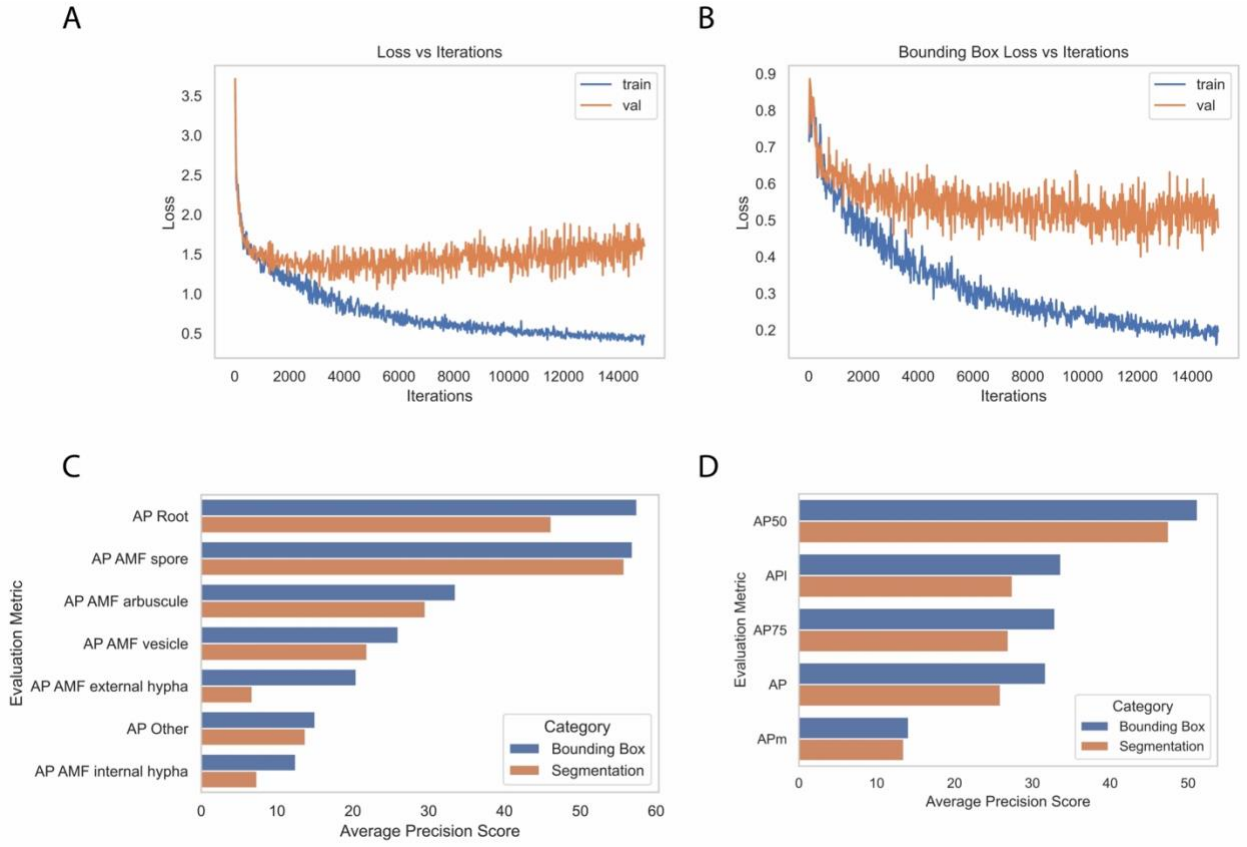

Figure 5. Four panels were plotted visualizing the evaluation of the MycorrhiSEE model. The loss of the model was tracked over the entire set of steps (per/model update). Two losses were calculated to optimize the model. Total loss in classification for the train and validation set are plotted in the blue and orange lines, respectively (A). The Bounding box regression loss was calculated over the over the entire set of steps (per/model update). Total loss in Bounding Box regression for the train and validation set are plotted in the blue and orange lines, respectively (B). The Average precision for each class were calculated for all test set images. The average precision for each class for the segmentation of the polygon and regression for each class are shown in blue and tan, respectively (C). The value performance for each class or ordered on the vertical axis by the performance of the bounding box. The aggregated average precision for AP50, API, AP75, AP and APm for the Bounding Box and Segmentation across all classes are displayed in blue and tan, respectively. The bars are ordered by bounding box score.

4.7 Bibliogrpahy

1 Delaux, P.-M. & Schornack, S. Plant evolution driven by interactions with symbiotic and pathogenic microbes. *Science* **371** (2021).

2 Bonfante, P. & Genre, A. Mechanisms underlying beneficial plant-fungus interactions in mycorrhizal symbiosis. *Nat Commun* **1**, 48, doi:10.1038/ncomms1046 (2010).

3 Figueiredo, A. F., Boy, J. & Guggenberger, G. in *Frontiers in Fungal Biology.*

4 Delgado-Ospina, J., Molina-Hern·ndez, J. B., Chaves-LÛpez, C., Romanazzi, G. & Paparella, A. The Role of Fungi in the Cocoa Production Chain and the Challenge of Climate Change. *Journal of Fungi* **7** (2021).

5 Gautam, S., Mishra, U., Scown, C. D. & Zhang, Y. Sorghum biomass production in the continental United States and its potential impacts on soil organic carbon and nitrous oxide emissions. *Global Change Biology Bioenergy* **12**, 878-890, doi:10.1111/gcbb.12736 (2020).

6 Perrone, G., Ferrara, M., Medina, n., Pascale, M. & Magan, N. Toxigenic Fungi and Mycotoxins in a Climate Change Scenario: Ecology, Genomics, Distribution, Prediction and Prevention of the Risk. *Microorganisms* **8** (2020).

7 Paterson, R. R. M., Ven,ncio, A., Lima, N., Guilloux-Benatier, M. & Rousseaux, S. Predominant mycotoxins, mycotoxigenic fungi and climate change related to wine. *Food research international* **103**, 478-491 (2018).

8 Begum, N. *et al.* Role of Arbuscular Mycorrhizal Fungi in Plant Growth Regulation: Implications in Abiotic Stress Tolerance. *Frontiers in Plant Science* **10** (2019).

9 Lendzemo, V. W., Kuyper, T. W., Mat˚aov·, R., Bouwmeester, H. J. & Ast, A. v. Colonization by Arbuscular Mycorrhizal Fungi of Sorghum Leads to Reduced Germination and Subsequent Attachment and Emergence of Striga hermonthica. *Plant Signaling & Behavior* **2**, 58 - 62 (2007).

10 Evangelisti, E. *et al.* Deep learning-based quantification of arbuscular mycorrhizal fungi in plant roots. *The New phytologist* (2021).

11 Khan, N. *et al.* Insights into the Interactions among Roots, Rhizosphere, and Rhizobacteria for Improving Plant Growth and Tolerance to Abiotic Stresses: A Review. *Cells* **10** (2021).

12 Li, E. *et al.* Rapid evolution of bacterial mutualism in the plant rhizosphere. *bioRxiv* (2020).

13 de la Fuente CantÛ, C. *et al.* An extended root phenotype: the rhizosphere, its formation and impacts on plant fitness. *The Plant journal : for cell and molecular biology* (2020).

14 Wang, H., Cimen, E., Singh, N. & Buckler, E. Deep learning for plant genomics and crop improvement. *Curr Opin Plant Biol* **54**, 34-41, doi:10.1016/j.pbi.2019.12.010 (2020).

15 Evangelisti, E. *et al.*

16 Evangelisti, E. *et al.*

17 Chai, J., Zeng, H., Li, A. & Ngai, E. W. T.

18 Jiao, W. *et al.* A deep learning system accurately classifies primary and metastatic cancers using passenger mutation patterns. *Nat Commun* **11**, 728, doi:10.1038/s41467-019-13825-8 (2020).

19 West, D. A. *et al.* MR image-guided investigation of regional signal transducers and activators of transcription-1 activation in a rat model of focal cerebral ischemia. *Neuroscience* **127**, 333-339, doi:10.1016/j.neuroscience.2004.05.022 (2004).

20 Sundararajan, M., Taly, A. & Yan, Q. Axiomatic Attribution for Deep Networks. arXiv:1703.01365 (2017). <https://ui.adsabs.harvard.edu/abs/2017arXiv170301365S>.

21 He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. arXiv:1512.03385 (2015). <https://ui.adsabs.harvard.edu/abs/2015arXiv151203385H>.

22 Ronneberger, O., Fischer, P. & Brox, T. in *MICCAI.*

23 Dietler, N. *et al.* A convolutional neural network segments yeast microscopy images with high accuracy. *Nat Commun* **11**, 5723, doi:10.1038/s41467-020-19557-4 (2020).

24 He, K., Gkioxari, G., Doll·r, P. & Girshick, R. B. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **42**, 386-397 (2020).

25 Su, W.-H. *et al.* Automatic Evaluation of Wheat Resistance to Fusarium Head Blight Using Dual Mask-RCNN Deep Learning Frameworks in Computer Vision. *Remote. Sens.* **13**, 26 (2021).

26 Khan, M. A., Akram, T., Zhang, Y. & Sharif, M. Attributes based skin lesion detection and recognition: A mask RCNN and transfer learning-based deep learning framework. *Pattern Recognit. Lett.* **143**, 58-66 (2021).

27 Pan, S. J. & Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* **22**, 1345-1359 (2010).

28 Antun, V., Renna, F., Poon, C., Adcock, B. & Hansen, A. C. On instabilities of deep learning in image reconstruction and the potential costs of AI. *Proc Natl Acad Sci U S A* **117**, 30088-30095, doi:10.1073/pnas.1907377117 (2020).

29 Mcgonigle, T., Miller, M. H., Evans, D. G., Fairchild, G. L. & Swan, J. A. A new method which gives an objective measure of colonization of roots by vesicular-arbuscular mycorrhizal fungi. *The New phytologist* **115 3**, 495-501 (1990).

30 Katz, D. S., Chue Hong, N. P., Clark, T., Fenner, M. & Martone, M. E. Software and Data Citation. *Comput. Sci. Eng.* **22**, 4-7 (2020).

31 Chue Hong, N. P. How to cite software: current best practice. doi:10.6084/m9.figshare.8124284.v1 (2019).

32 He, K., Gkioxari, G., Doll·r, P. & Girshick, R. B. Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2980-2988 (2017).

33 Patterson, G. & Hays, J. in *ECCV*.

34 Services, A. W. <https://docs.aws.amazon.com/> (2022).