TRANSFORMING OUR DNA — GENOTYPING STRUCTURAL VARIANTS USING VISION TRANSFORMERS

Benjamin Allan-Rahill

Advisor: Professor Linderman

A Thesis

Presented to the Faculty of the Computer Science Department

of Middlebury College

May 2023

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION AND BACKGROUND

Structural Variants (SVs), are changes from the reference DNA, greater than 50 DNA base-pairs in size. SVs play a significant role in disease [3, 47]. However, discovery—detecting new variants—and genotyping—determining the number of copies of a known SV—in short-read sequencing (SRS) data is limited in accuracy due to the variants' large sizes relative to sequencer read length [3, 12].

Initial attempts to improve SV genotyping accuracy use statistical models that operate on data aggregates [CITATION]. Fine-tuned for a particular sequencing pipeline or variant type, these statistical models are limited in their generalization. Machine learning classifiers begin to alleviate this issue, using hand selected features as inputs to that models that classify the SV genotype from short-read sequencing data.

These machine learning approaches are limited by using a subset of relevant data. Using images generated from sequencing data, deep-learning approaches are able to learn features from higher resolution data, outperforming classic statistical and machine learning models.

However, all of these approaches are limited by the availability of high-quality training data. Reframing SV genotyping as an image similarity problem, rather than direct classification, begins to eliminate the issues presented by limited and erroneous SV datasets. Combining an image similarity paradigm with simulated data further improves the accuracy of these deep-learning approaches [30, 31].

Transformers, a novel neural network that uses a self-attention mechanism were first applied to machine translation and natural language processing. Improvements on the original architecture have been widely used, most famously as the backbone for the GPT family of models from OpenAI. The recent application of transformer neural networks to computer vision tasks has achieved state-of-the-art (SOTA) or competitive results for

tasks such as image classification and object detection [17, 25, 29].

In this thesis, I first provide background on SVs and the foundational approaches to SV genotyping. I then detail the transformer model and discuss the benefits and trade-offs of swapping a Convolutional Neural Network (CNN) backbone for a transformer, described in Chapter 2. Evaluating these trade-offs, I propose a novel approach that uses image similarity and Vision Transformers (ViT) to improve the accuracy of SV genotyping in SRS genomes.

1.1 Structural Variants (SVs)

Genetic variation is defined as differences between a sample genome and the reference genome—an agreed upon DNA sequence for comparison. These DNA sequence changes can include insertions, deletions, inversions and other more complex edits [3, 12]. Illustrated in Fig. 1.1, the simplest variant types to understand are insertions—new sequences of DNA that are in the sample genome but not within the reference—and deletions—DNA missing from the sample when compared to the reference.

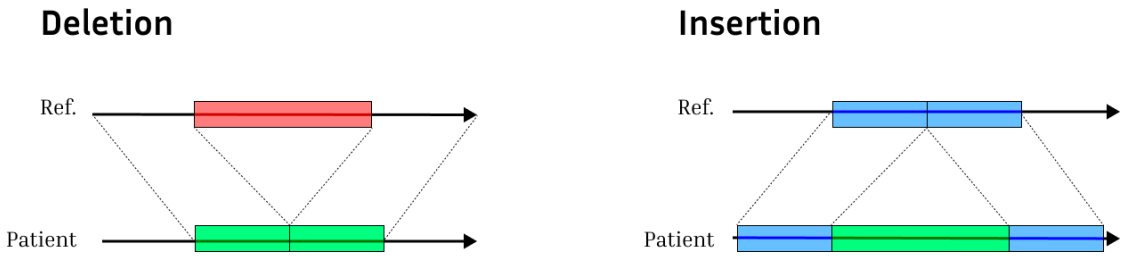

Figure 1.1: Types of Structural Variation (SV) — adapted from Alkan et al [3].

I focus on a subset of genetic variants, SVs, as defined above. The clinical relevance of SVs is high. Structural variants contribute a significant portion of total genetic variation within the human population [3]. SVs account for 0.5-1% of sequence variation among individuals while Single Nucleotide Variants (single base-pair changed) account for 0.1% [3, 12, 47]. Some SVs rarely appear among the population (less than 1% of individuals), but because of their size, >1000 base-pairs (kbp), affect large portions

of DNA. Specifically, these large, uncommon variations have been linked with genetic disorders and other neurological conditions like autism [3, 41].

1.1.1 Sequencing Technologies

To understand SV presence in and impact on the human genome, two primary categories of sequencing procedures exist: long-read sequencing (LRS) and short-read sequencing (SRS). Each process works to most accurately determine the genetic makeup–base-pair sequence–of a genome; however, tradeoffs exist between cost and accuracy.

SRS works by breaking up the genome into small fragments (∼35-700bp) and sequencing both ends of each fragment [23]. Software takes these sequencing fragments as input and realigns the fragments back to a reference genome; this is referred to as the alignment process. Secondary data features such as read coverage—the number of reads aligning to, a position in the reference sequence— are calculated from the aligned fragments. Genetic variation can be probabilistically detected and assessed from these secondary features of the aligned sequenced reads. This fragmentation process involved in short-read sequencing is non-deterministic: sequencing an the same sample multiple times does not yield identical data [3, 23].

Due to the short length of SRS fragments(∼35-700bp) compared to SV length (¿50bp) [6] and the stochastic nature of fragmentation, SRS technologies cannot detect many SVs that are identified by LRS. A recent study that used LRS to assemble genomes identified 107,590 SVs, *68%* of which were not discovered by SRS [18].

These accuracy benefits of LRS do not come without trade-offs—long-read sequencing can identify a larger set of SVs, but is more expensive and offers lower overall throughput [22]. Therefore, short-read sequencing remains a widely used approach for sequencing new samples. One project, *All of Us*, announced the release of 98,560 new SRS genomes, demonstrating the broad availability of SRS genome data [**?**].

1.1.2 Genotyping

Two steps define the SV "calling" process: discovery and genotyping. Discovery is the process of identifying SVs within a sample whereas genotyping determines the copy-number (0, 1, or 2) of an already identified SV, present within a sequence. Because of how DNA is combined from parents, it's possible to have zero, one, or two copies of a variant. These different genotypes are referred to as homozygous-reference, heterozygous, and homozygous-alternate respectively. Due to the limitations and random error in sequencing technologies discussed above, the initial discovery step can be highly inaccurate. Some methods for SV discovery report false discovery rates as high as 89% [12]. For the purposes of this thesis, I concentrate my efforts on the genotyping facet of the calling process.

1.2 Genotyping Approaches

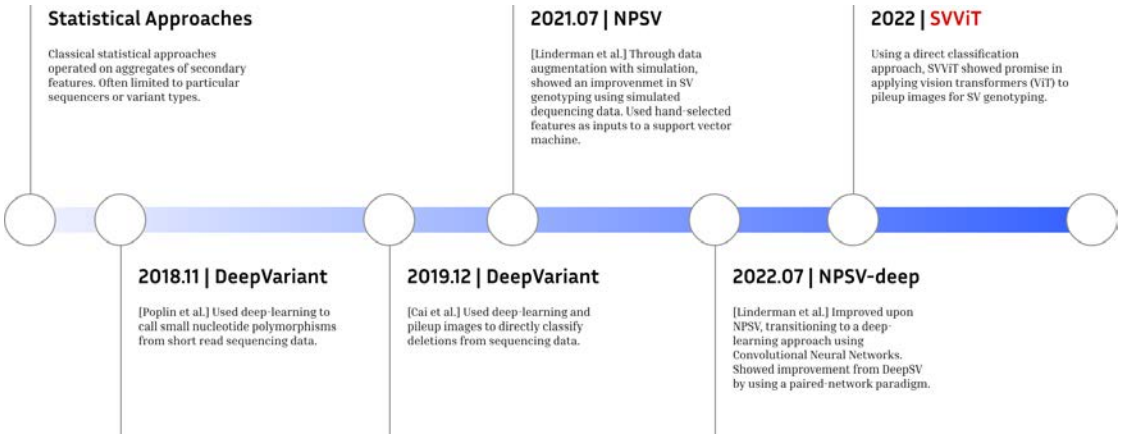

Figure 1.2: Timeline of Genotyping Approaches.

1.2.1 Statistics

Due to the the length of SVs (>50bp) being larger than common SRS read length (∼30-700bp), the genetic variation cannot be detected directly by sequencing the entire variant. Therefore, SVs must be detected probabilistically through secondary features—metrics calculated from the sequencer's output. These secondary features include metric such as read coverage—how many reads match to a genetic location—and fragment size—the inferred total size of the fragment after alignment back to the reference sequence. Operating on aggregate metrics of these features, statistical models have been used to probabilistically detect and genotype structural variation.

1.2.2 Machine Learning

These statistical models were tuned to a particular sequencing platform and thus, lack generalization. Developed to improve upon these statistical, parameterized approaches, the Non-Parametric Structural Variant (NPSV) genotyper used a machine learning technique, support vector machines (SVM), to genotype structural variants [31]. Using labeled input data, SVMs are classifiers that operate on engineered features—attributes extracted from the data—to distinguish between classes.

NPSV was not the only approach to improve upon statistical methods with machine learning [5, 9, 13, 15, 39]. Similar to NPSV but limited to insertions and deletions, GINDEL, built by Chu et al., also extracts features from SRS data as input to an SVM for classification. Extracting a fixed set of features from the SRS output limited the data used by the model for classification. Another indication of the potential to improve this approach is that it's different than genotyping approach used by industry experts. These experts use *pileup images* to make genotyping decisions.

1.2.3 Pileup Images

Pileup images are multi-channel charts that display data for a specific genetic locus. As shown in Figure 1.3, each pixel column in the image represents a single base-pair and each pixel represents a single sequencing read. The image generation pipeline builds an image for a padded window around the putative SV's breakpoints to ensure that the putative SV is always centered in the image. Each color channel within the image represents a particular feature, such as read coverage. The final color is a composite of this evidence and, thus, black regions represent an absence of data.

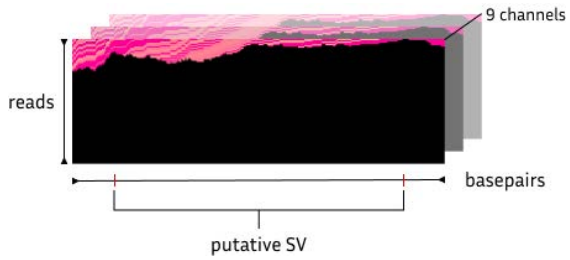

Figure 1.3: Pileup Image Breakdown.

1.2.4 Deep Learning

The distinction between the "classical" machine learning approaches, detailed above, and deep learning (DL) is the feature set that the model uses to make a classification. Classical machine learning classifiers are engineer by a researcher or software engineer who *hand-selects* the features from the dataset from which a decision tree or other classifier should make its decision. In DL, an artificial neural network (ANN) learns these features as part of the training process. This delineation is shown in the context of SV genotyping in Fig. 1.4.

The benefit of this approach is that the model learns features from the data, rather than only learning weights for a fixed set of features; however, due to the hidden layers and deep connectivity (i.e. many possible data paths) of these ANNs, it's often hard to

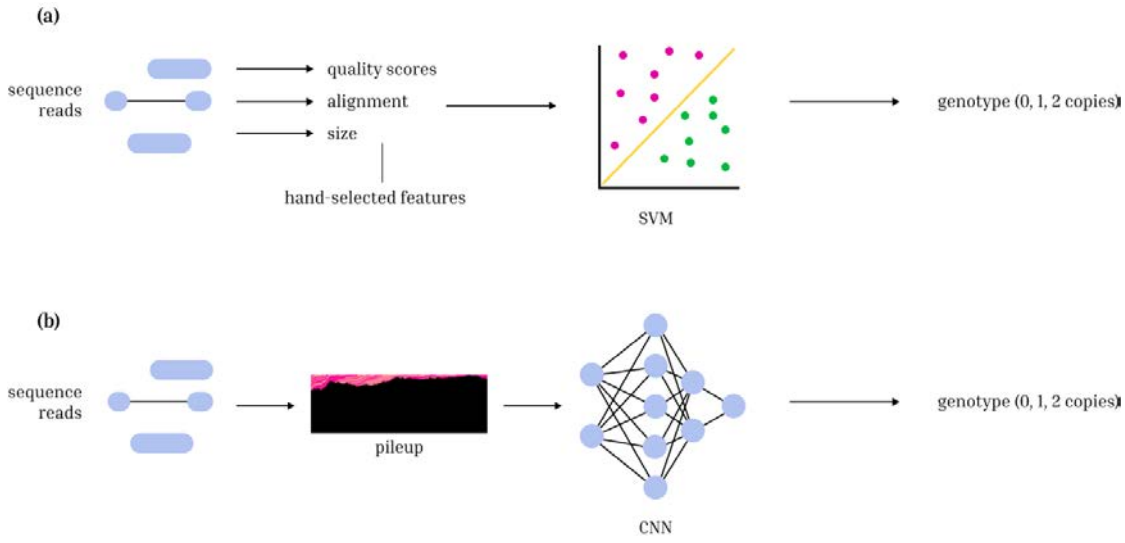

Figure 1.4: Classical vs Deep Learning approaches — (a) Classical machine learning approach that uses hand-selected features and a support-vector machine for direct classification. (b) Deep learning approach that first generates pileup images from sequencing data and trains a CNN to genotype from these images.

determine what features the model uses to make its classification, creating a classical "black box."

Experimenting with DL for genotyping, researchers at Google implemented a learned feature, using CNNs (described later in Section 2.2.1) to detect and genotype SNVs and small indels—insertions and/or deletions. Mimicking the approach a human expert would take to genotype, the researchers generated pileup images from the aligned sequencing reads. This work improved upon The Genome Analysis Toolkit (GATK) [33], a complex ensemble of statistical models that used hand-crafted features, demonstrating the effectiveness of a pileup imaged based approach.

Their model, DeepVariant, out-performed SOTA SNV variant calling tools with $>$99% accuracy and achieved "Best Overall" in 3 out of 4 categories in the PrecisionFDA v2 Truth Challenge [1], a challenge designed to assess variant calling performance. These results prove the efficacy of DL, computer-vision based models to analyze sequencing data; however, as it was designed for smaller variants, its accuracy

at SV genotyping is undefined [37].

DeepSV and SamplotML extended the approach of DeepVariant to focus on structural variants. Instead of SNVs, their DL-based approach focused on long deletions. The researchers used a CNN to directly classify pileup images [9, 11].

I hypothesize that the limited results of these SV studies stem from their approach to directly classify (i.e., genotype) from images. Direct classification models take an single input and directly predict the genotype. This is a common and valid approach for many classification tasks where datasets contain good exemplars—distinct and distinguishable labeled examples. Unfortunately, SV datasets do not fit this criterion and thus, require augmented approaches to overcome the limitations in training data.

1.3 Data augmentation with simulation

Typical CNNs and other deep-learning backbones (i.e., the base network that is used) require large datasets. For reference, the dataset used to train Google's DeepVariant CNN contained over 500 million examples. Unfortunately, SV datasets have no more than 3.4 million examples (0.7% of the DeepVariant dataset). Furthermore, there is an over representation of homozygous-reference examples (i.e., no variant present) within these datasets, meaning we have even less data for certain genotypes. Additionally, SV datasets often contain duplicate or overlapping descriptions of the same SV. Lastly, the stochastic nature of the SRS process creates a large variance in SV descriptions.

Data augmentation through simulation of the SRS process can begin to alleviate these limitations. NPSV demonstrated this by augmenting their training data with simulated sequencing reads [31]. The researchers begin to model the pipeline-specific biases by simulating the SRS process [28].

1.4 Metric Learning

Deep-neural networks, such as Convolutional Neural Networks (CNNs) can struggle to learn accurate representations for data with minimal similar examples [32]. This problem commonly arises in spaces such as image search and facial recognition.

Researchers pioneered signature verification by treating the problem as image-similarity using paired networks. A paired architecture takes two inputs, runs them each through the same network with the same weights, and computes an energy function between the outputs of the model. With image similarity, the weight-sharing holds that two similar inputs will be mapped to similar areas of the feature space. Often, euclidean distance, a common energy function, can be used as a metric for similarity.

This approach improves models restricted to observing a single or zero examples of a class before making a prediction. These learning paradigms are called one-shot and zero-shot learning, respectively, or more generally, few-shot learning. Few-shot learning aims to solve the problem of learning feature representations from little data. The problem commonly arises in image verification tasks like signature verification and facial recognition.

1.4.1 Contrastive and Other Loss Functions

To facilitate a metric-learning objective in the training loop, a specific loss function must be used. At the end of each training loop, the loss function is calculated to determine the changes needed for each weight. To learn similarity, the model must learn its weights to map similar inputs to a similar place within the final feature space—the embedding space, demonstrated by Fig. 1.5.

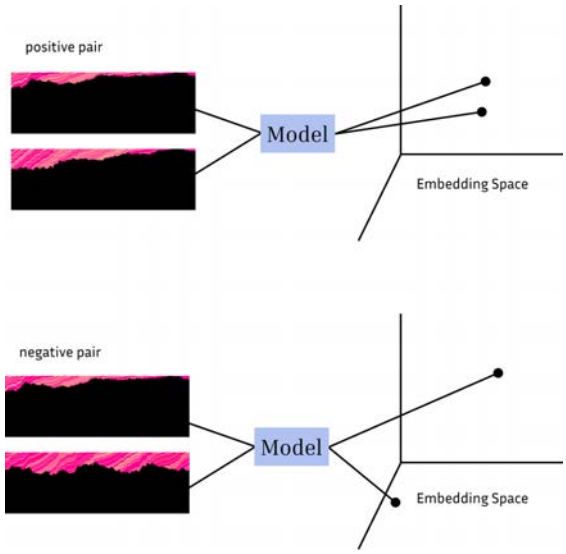

Figure 1.5: Embedding Space — adapted from Bangert et al. [8].

The most common and simplistic approach is to calculate the distance between the pairs of inputs. Each pair will have an accompanying label describes whether the pairs are similar or not. Similar pairs are referred to as positive and dissimilar as negative. The contrastive loss function (Eq. (1.1)) measures how close to 0 positive pairs are and how far past a threshold (m) negative pairs are [24]. In this equation Y represents the label (0 for negative pairs and 1 for positive), D is the distance between the pairs' embeddings, and m is a margin constant (typically 1). Using this loss function to optimize the model's weights aims to bring positive pairs together and push negative pairs apart in the embedding space.

$$Y * D^2 + (1 - Y)\, max\,(0, m - D)^2 \tag{1.1}$$

Contrastive loss has been shown to be an effective method for learning similarity; however, in many cases, similarity within a pair is not binary. For example, in SV genotyping, having two copies of a variant is more similar to having a single copy than to having none. If this distinction were to be accounted for in the model, one would need to employ a different kind of loss function. One approach to solve this calculates the

loss among a triplet of inputs—an anchor, a positive example, and a negative example—assuring that the distance between the anchor and the negative is further than the distance between the anchor and positive. See Fig. 1.6.

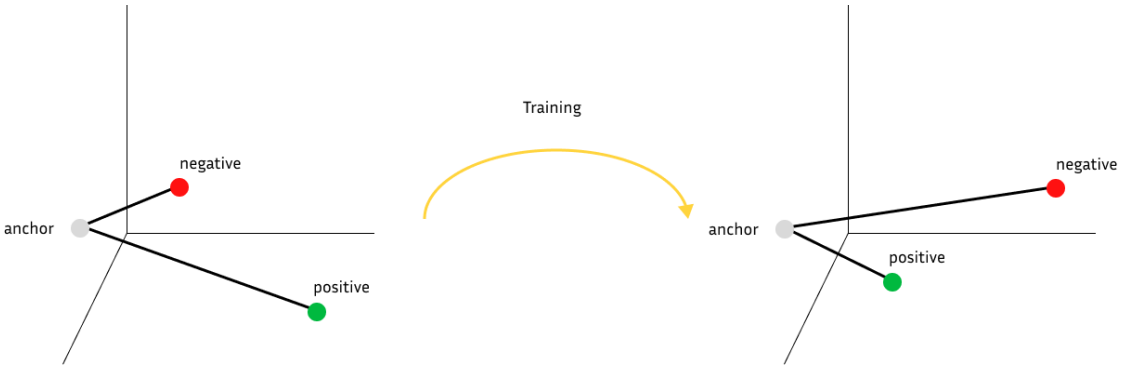

Figure 1.6: Triplet loss.

For example, FaceNet, designed by Schroff et al., trained a CNN using triplet loss to account for nuance in inter-class dissimilarity. FaceNet achieved new SOTA results on the widely used Labeled Face in the Wild (LFW) dataset, reaching 99.63% accuracy [40]. Additionally, this work achieved high representational efficiency—accuracy from minimal data (128-bytes per face). The researchers carefully selected the triplets for training, ensuring minimal examples of the same face were present in each batch. This careful creation of training batches is essential when using these sophisticated loss functions. This triplet loss function may be effective at capturing the distinction in similarity among SV genotypes; however, without careful selection of triplets, using this loss function can impact the model's performance [35].

1.4.2 Combining Simulation and Metric Learning

Metric learning begins to aid the issues of limited training data, but the lack of good exemplars in SV datasets necessitates generating these exemplars through sequencing simulation. Linderman et al. combined metric learning and data simulation in NPSV-

deep [31]. The sequencing simulation was adapted to generate pileup images for each genotype. These images became input to a paired-network with a metric learning objective.

Additionally, comparing NPSV-deep against other DL approaches [9, 39], the image similarity approach was more successful than direct classification [30]. NPSV-deep performed with an genotyping accuracy (for deletions) of 90.6% for the Genome in a Bottle (GIAB) v0.6 Tier 1 SV dataset. For comparison, an approach that used the same model but with direct classification only attained 74.8% accuracy. Not only does the few-shot learning approach combat the lack of training examples, combining few-shot learning with simulated data begins to alleviate the issues presented by the stochastic process of sequencing.

The research above demonstrates a natural progression from hand-selected features to learned features using pileup images. Additionally, shifting the genotyping problem from direct classification to image similarity has improved accuracy. However, there still exists limitations of a paired-network approach with image similarity.

CHAPTER 2

TRANSFORMER NEURAL NETWORKS

In the following section, I discuss the history of the transformer neural network in natural language processing (NLP) and more recently, in computer vision (CV). I contrast transformers with another computer vision approach, CNNs, and discuss the potential benefits and trade-offs of using this model in the SV genotyping process.

2.1 Machine Translation

Transformer neural networks, initially developed in the the field of natural language processing for machine translation, can incorporate long-range context using their *self-attention* mechanism, described in detail in Section 2.1.1 [46]. Using the encoder-decoder architecture shown in Fig. 2.1, a transformer maps an input sequence (of length n) to an encoded sequence of equal length. From this encoded sequence, the decoder can generate an output sequence of arbitrary length (length m). The self-attention module is built into both the encoder and decoder, allowing the decoder to "attend over all positions in the input sequence" [46].

Transformers were developed to solve issues of memory and computational efficiency in recurrent neural networks (RNNs), long short-term memory (LSTM), and other sequence-to-sequence models. RNNs use a sequential model to incorporate relevant data from separate sequence parts. Due to memory constraints, the sequential model limits the sequence length that the model can account for in its recurrent mechanism. Transformers solve this issue by relying solely on self-attention, a data-parallel calculation.

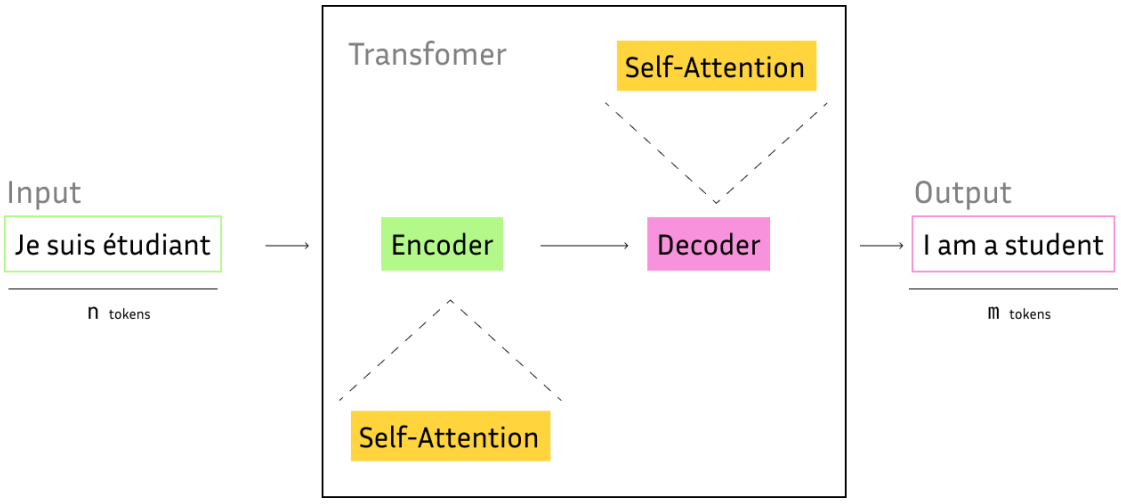

Figure 2.1: Classical NLP Transformer Architecture — adapted from Han et al. [25].

2.1.1 Self-Attention

The self-attention mechanism of a transformer is the most specialized aspect that makes it successful at translation, text generation, and other NLP tasks. For a given word, the query word, the attention mechanism calculates a vector attention score for the rest of the words in the sequence. Each word's attention score represents the relative importance of that word to the query word. This builds a representation of pairwise relationships within a sequence that can successfully capture global context in sequence-to-sequence modeling. The core attention mechanism was designed by Bahdanau et al. [7] and is run by multiple attention heads in the encoder and decoder of a transformer [46]. Furthermore, unlike RNNs, this calculation does not have sequential dependencies and thus can be run in parallel.

Similar to the improvements upon contrastive loss, more sophisticated mechanisms for self-attention have been built [10, 14, 26, 44, 48]. One approach, pioneered by Facebook AI, improves upon the attention span—how many tokens can be accounted for—of the transformer [44].

2.2 Vision

Many attempts have been made to incorporate the transformer attention mechanism into vision models [27,36,38,49]. A rote attention application to each pixel in an image (i.e., each pixel attends to every other pixel) would scale quadratically with image size. However, one approach, the Vision Transformer (ViT), successfully reduced this complexity with minimal modification of the original transformer architecture [17].

With a largely out-of-the-box Transformer backbone, ViT first splits an image up into square patches. Next, the transformer builds a 1D sequence of these patches as input tokens, analogous to words in the NLP application. From there, a positional encoding is added to the patches. Because the self-attention calculation can be run in parallel, the model requires a method to track the position of the query patch. This positional encoding helps to ensure that if the image patches were rearranged, the inputs to the self-attention mechanism will be different. An adapted version of this model architecture is shown in Fig. 2.2.

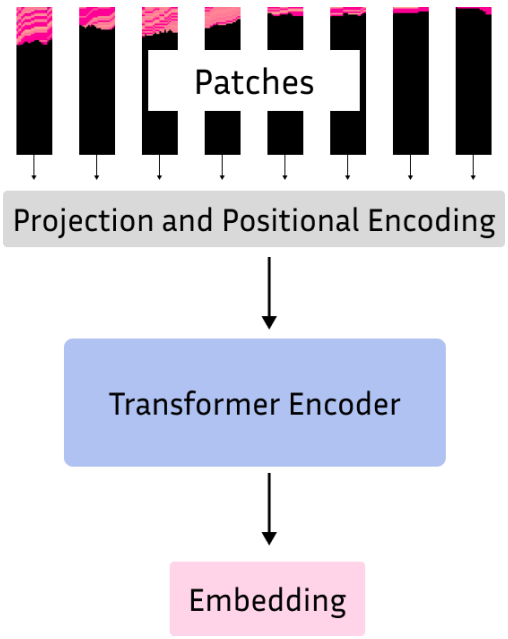

Figure 2.2: Adapted architecture of a transformer for a pileup — adapted from Dosovitskiy et al. [17].

ViT outperforms widely used computer vision models for image classification and image-similarity tasks [17]. However, this result was achieved only after extensive pre-training on the large JFT-300M dataset, with hundreds of millions image examples. Though requiring lots of training data, these transformers were efficient to train, reaching SOTA accuracy with 2-4X less compute. Aiming to limit the need for large pre-training, researchers at Facebook developed Data Efficient Image Transformers (DeiT) that require smaller datasets for training by using substantial data augmentation [45].

Variable Length Inputs

Though ViT was trained on fixed dimensional inputs, transformers can be trained with variable length inputs: Gong et al. used transformers to classify audio spectrograms of variable length. Additionally, the researchers employed transfer learning for ViT on ImageNet to improve the accuracy of their model [21].

2.2.1 Contrast with CNNs

CNN Architecture

CNNs are built from a sequence of convolutional layers that pass over the image. These convolutional layers function as filters—each layer calculating a function over patches of the images. A deep CNN is built from many of these convolutional layers in a hierarchical system that allows for deep complex representation from the stacking of these filters.

The convolutional layers of CNNs are trained to detect representations of increasingly complex entities. For example, a CNN trained to classify images of dogs, lower layers consist of convolutions that will detect simple edges, then corners from these edges, building up to higher layers that can detect dogs.

Inductive Biases

When a machine learning model is built, assumptions are made about its expected inputs and outputs, termed *inductive biases*. For example, a simple linear regression model assumes that its output is a linear function of its inputs. Inductive bias is needed within machine learning models to limit the parameter space to help the model learn and generalize from training examples.

An underlying difference between CNNs and ViTs is the set of inductive biases present within the model. Due to the connected network of convolutional layers, CNNs possess the trait that if the input image were to be transformed (e.g., scaled or rotated) the the output for each convolution would be equally transformed. The trait is called translational equivariance. In addition to translational equivariance, CNN's filtering over patches of the image creates a two-dimensional neighborhood structure helpful for representing locally related information [17]. For example, a CNN expects that relative importance between two data points is a function of their distance, i.e., closer pixels are more important to each other. This exists as a bias in the model because close pixels will effect each other through convolutions.

2.3 Metric Learning with ViT

Different inductive biases, compared to CNNs, can help transformers learn more general representations of images; however, as demonstrated by [17], a larger training set is required to obtain this result.

Initial research shows that metric-based learning approaches for image similarity using transformers can improve upon convolutional approaches [16, 19, 20]. El-Nouby et al. trained a paired network of transformers and used a contrastive loss function as the metric learning objective. This model architecture out-performed SOTA tools by as

much as 2.6% for three separate public image retrieval benchmarks (OP, CUB-200 and In-Shop) [19].

2.4 Genomic Applications

Tasseo, an image classifier built with transformers, was trained to detect chromosomal abnormalities and identify chromosomes in karyotype images [42]. The researchers used a variant of the ViT model, TopViT, that encodes distances between image patches in the original image rather then a simple 1D encoding. This novel strategy helps to maintain the 2D-topology of the chromosome in the image. Furthermore, the research showed success of transformers, when the training dataset is limited .

GeneBERT, uses transformers to model the interactions between different regulatory elements of a genome. Mo et al. expanded upon CNN-based solutions that did not account for interaction within a genomic sequence [34].

Motivating my research, collaborators at Google built SVViT with an approach similar to DeepVariant, genotypes SVs through direct classification of pileup images [2]. Although not published, SVViT performs with an accuracy of 88.6% (non-reference) for the GIAB dataset. This improves upon the analogous direct classification approach with a CNN, but is less accurate than NPSV-deep, a paired-network with CNNs.

CHAPTER 3

DISCUSSION

Transformers, first built for language tasks, succeed when adapted to vision applications. The ability to capture long-range context through their self-attention mechanism has been shown to build effective image representations and outperform traditional convolutional approaches. Additionally, the computational efficiency of training these model architectures over CNNs can improve the training and development process.

Most importantly, the self-attention model of transformers relates global information rather than just local information. This can help the model incorporate long-range dependencies and build more global representations of relative importance in the image. This could be transformational in the problem space of SV genotyping.

Because of the relational nature of DNA, we can hypothesize the utility of this attention mechanism to learning representations of pileup images. There may be long-range relationships, between distant genetic loci, represented within pileup images. These relationships will not be captured by a CNN architecture, which focuses on local relationships, but would be accounted for by transformer's self-attention.

Though attention may be promising for genomic applications, the inductive biases presented by CNNs can be helpful for image classification. These properties, like translational equivariance, help ensure that if an object is in any part of the image or the image in transformed, it will be classified correctly. Regarding SV genotyping, these inductive biases are less critical for helping constraining our problem. Since our pileup images are generated from sequencing data, the data appears upon a defined axis—each pixel column corresponds to a genomic position (see Fig. 1.3). To control the structure of the images, the pileup image is generated with the SV in the center. This is analogous to only using head-shots as input to a facial-recognition model. These constraints may help to minimize the trade-offs of swapping the model architecture for a transformer.

Lastly, to the broad definition of SVs, allowing the model to accommodate variable length inputs may improve the genotyping process. Current approaches, such as NPSV-deep, constrain the inputs to a fixed (100x300 pixels) size by compressing larger SVs. This can impact the resolution of the image and underlying data. If an SV exceeds 300 base-pairs, the image will lose the one-to-one correspondence between pixel columns and base-pairs. We have to aggregate data across multiple base-pairs, losing resolution and impacting the genotyping accuracy for these large variants. Using a transformer architecture that accepts variable length inputs, as shown by the Audio Spectrogram Transformer, may improve the genotyping accuracy.

To recap, applying transformers to SV genotyping presents a few possible benefits. The self-attention mechanism may account for complex genetic relationships, undetectable by CNNs, by building a representation of global pair-wise relationships of sequenced bases. Different inductive biases compared to CNNs could help the model learn more general representations, potentially lessening the impact of variance in SV size and location of evidence within the images. Finally, variable size input images would allow for an increase in data resolution for large variant types.

CHAPTER 4

PROTOTYPED APPROACH AND PROPOSED WORK

In the following section, I discuss the in-progress plan to incorporate a transformer-based backbone with a metric learning and simulation paradigm and detail the extent of progress. Our work will combine the learnings from NPSV-deep and SVViT to improve genotyping accuracy‘. I hypothesize the transformer backbone will improve the accuracy of genotyping.

4.1 Adapting SVViT

Motivated by NPSV-deep and SVViT [2], my work proposes a combination of the transformer-based classification approach performed by SVViT with the paired-network approach of NPSV-deep. Modifying the direct classification network of SVViT will involve building a specialized wrapper around SVViT that takes real and simulated data as input [2].

Additionally, I will need to modify the network from a direct classification architecture to a paired-network approach with a metric learning objective. This will involve swapping the network's loss function to use contrastive loss.

4.2 ViT Variant Selection

Many variants of the original ViT have been built for specialized purposes. As the computational complexity of attention scales quadratically with input lengths, the approaches are focused on improving the efficiency of the self-attention mechanism [14, 43]. Since SVViT uses X-ViT by default, a paired-network approach with this backbone has been prototyped, but results have not been collected. After these results are

collected, other methods, such as TopVit, that use improved positional-encoding methods and have shown success in biological image analysis [42], should be investigated.

4.3 Dataset Creation

I have modified the dataset creation pipeline of SVViT to incorporate the simulated data used by NPSV-deep. The dataset from NPSV-deep includes distinct multi-dimensional arrays per putative SV. Each array contains a real image and a set of simulated images for each of the three possible genotype. Each genotype simulation is run N times, creating N replicates and $3 * N$ simulated images.

My prototyped approach uses this dataset from NPSV-deep out of the box, building positive and negative pairs for use with a contrastive loss training loop. However, due to the semantic design of this dataset, with each array representing a single putative SV, the dataset is extensible to more sophisticated loss functions as discussed in Section 1.4.1.

4.3.1 Paired Network

To adapt the direct-classification approach of SVViT, I modified the backbone network structure. SVViT is built on top of a machine-learning package, Scenic. Scenic provides abstracted functions and utilities to train and test attention-based networks, e.g., transformers.

The original approach of SVViT uses single image inputs in a direct classification paradigm. Given a single SV represented as a pileup image, the X-ViT model predicts the genotype. I have prototyped a modification of this approach that instead uses a paired network, as detailed in Fig. 4.1. The paired network works multiple input images (simulated and real), runs each through identical X-ViT network with the same weights, and then computes the contrastive loss function based on the outputs of these models.

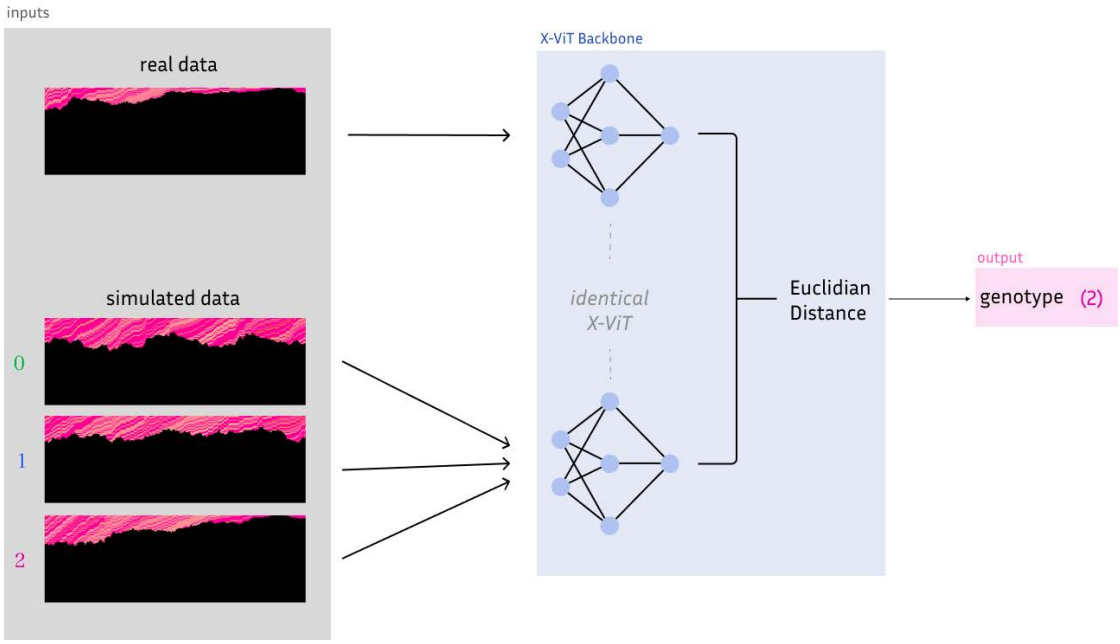

Figure 4.1: Structure of the Paired ViT.

4.3.2 Loss Function

Initially, it will be most straightforward to modify the SVViT approach to operate on a pair of input images and compute the contrastive loss. I have prototyped this initial approach, but am unable to collect results at this time. A simple contrastive loss, operating on pairs of simulated and real images, broke the semantic grouping of inputs as single variants but required the least modification to the codebase.

A proposed modification will transition the inputs to single variants: how the NPSV-deep network expects its inputs. Specifically, these images will be a single real image, called the query, and N simulated images (per genotype), termed the support images.

4.3.3 Parameter Modifications

There are many parameters of the X-ViT model that are easily customized for our use-case. In my prototype I have made no modifications to these parameters.

The first parameter modification to test would be the patch size. This parameter

determines how the input images are split into tokens. The default configuration breaks the image into square patches, but I hypothesize that a column patch would be more effective. Due to the inherent coordinate system within a pileup image, column patches would split the image into groups of contiguous base-pairs. The patches could be a single pixel wide and represent individual base-pairs. This would accomplish the task of splitting the image into distinct semantic entities, base-pair positions. Doing so will help the model understand global relationships between the genome sections relevant to each SV.

Another modification enabled by transformers would be a custom positional encoding. The positional encoding is a 1D value that helps to ensure patch-wise translational equivariance, defined in Section 2.2.1. This encoding allows the model to incorporate the position into its learning. For SV genotyping, the position of each sequence read is important relative to the putative breakpoints of the SV. For example, reads near and around the breakpoints will be the most relevant to the genotyping process. Therefore, it may be beneficial to use a custom positional encoding that encodes the relative distance to the breakpoints of the SV.

Finally, many hyper-parameters can be swept—iterated over, and optimized—given enough time and compute resources. A few parameters determine the materialized structure of the backbone network, while other parameters like learning rate modify the training process.

4.4 Conclusion

I have prototyped a paired-network structure with an X-ViT backbone and a metric learning objective. This model builds pairs of simulated and real images from the dataset as input to the network. Extensions of my prototype will first focus on modifying the input pipeline to use a single putative variant, many pairs of images, as input to the

model. After this, modifications should be made to the patch size and positional encoding. Finally, additional ViT variants should be assessed for their attention mechanisms and computational efficiency. The code for my prototype is accessible on GitHub [4].

BIBLIOGRAPHY

[1] Improving the accuracy of genomic analysis with DeepVariant 1.0 – google AI blog. URL: https://ai.googleblog.com/2020/09/improving-accuracy-of-genomic-analysis.html.

[2] SVViT. URL: https://github.com/google-research/scenic/tree/main/scenic/projects/svvit.

[3] Can Alkan, Bradley P. Coe, and Evan E. Eichler. Genome structural variation discovery and genotyping. 12(5):363–376. Number: 5 Publisher: Nature Publishing Group. URL: https://www.nature.com/articles/nrg2958, doi:10.1038/nrg2958.

[4] Benjamin Allan-Rahill. NPSViT, Mar 2023. URL: https://github.com/jiito/scenic.

[5] Danny Antaki, William M Brandler, and Jonathan Sebat. SV2: accurate structural variation genotyping and de novo mutation detection from whole genomes. 34(10):1774–1777. doi:10.1093/bioinformatics/btx813.

[6] Peter A. Audano, Arvis Sulovari, Tina A. Graves-Lindsay, Stuart Cantsilieris, Melanie Sorensen, AnneMarie E. Welch, Max L. Dougherty, Bradley J. Nelson, Ankeeta Shah, Susan K. Dutcher, Wesley C. Warren, Vincent Magrini, Sean D. McGrath, Yang I. Li, Richard K. Wilson, and Evan E. Eichler. Characterizing the major structural variant alleles of the human genome. 176(3):663–675.e19. doi:10.1016/j.cell.2018.12.019.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. URL: https://arxiv.org/abs/1409.0473v7.

[8] Patrick Bangert, Hankyu Moon, Jae Oh Woo, Sima Didari, and Heng Hao. Medical image labeling via active learning is 90% effective. In Kohei Arai, editor, *Advances in Information and Communication*, Lecture Notes in Networks and Systems, pages 291–310. Springer International Publishing. doi:10.1007/978-3-030-98012-2_23.

[9] Jonathan R. Belyeu, Murad Chowdhury, Joseph Brown, Brent S. Pedersen, Michael J. Cormier, Aaron R. Quinlan, and Ryan M. Layer. Samplot: a platform for structural variant visual validation and automated filtering. 22(1):161. doi:10.1186/s13059-021-02380-5.

[10] Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language BERTs. 9:978–994. doi:10.1162/tacl_a_00408.

[11] Lei Cai, Yufeng Wu, and Jingyang Gao. DeepSV: accurate calling of genomic deletions from high-throughput sequencing data using deep convolutional neural network. 20(1):665. doi:10.1186/s12859-019-3299-y.

[12] Varuna Chander, Richard A Gibbs, and Fritz J Sedlazeck. Evaluation of computational genotyping of structural variation for clinical diagnoses. 8(9):giz110. doi:10.1093/gigascience/giz110.

[13] Colby Chiang, Ryan M. Layer, Gregory G. Faust, Michael R. Lindberg, David B. Rose, Erik P. Garrison, Gabor T. Marth, Aaron R. Quinlan, and Ira M. Hall. SpeedSeq: ultra-fast personal genome analysis and interpretation. 12(10):966–968. Number: 10 Publisher: Nature Publishing Group. URL: https://www.nature.com/articles/nmeth.3505, doi:10.1038/nmeth.3505.

[14] Krzysztof Choromanski, Han Lin, Haoxian Chen, Tianyi Zhang, Arijit Sehanobish, Valerii Likhosherstov, Jack Parker-Holder, Tamas Sarlos, Adrian Weller, and Thomas Weingarten. From block-toeplitz matrices to differential equations on graphs: towards a general theory for scalable masked transformers. URL: http://arxiv.org/abs/2107.07999, arXiv:2107.07999[cs], doi:10.48550/arXiv.2107.07999.

[15] Chong Chu, Jin Zhang, and Yufeng Wu. GINDEL: Accurate genotype calling of insertions and deletions from low coverage population sequence reads. 9(11):e113324. Publisher: Public Library of Science. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0113324, doi:10.1371/journal.pone.0113324.

[16] Chun Ding, Meimin Wang, Zhili Zhou, Teng Huang, Xiaoliang Wang, and Jin Li. Siamese transformer network-based similarity metric learning for cross-source remote sensing image retrieval. 35(11):8125–8142. doi:10.1007/s00521-022-08092-6.

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. URL: http://arxiv.org/abs/2010.11929, arXiv:2010.11929[cs], doi:10.48550/arXiv.2010.11929.

[18] Peter Ebert, Peter A. Audano, Qihui Zhu, Bernardo Rodriguez-Martin, David Porubsky, Marc Jan Bonder, Arvis Sulovari, Jana Ebler, Weichen Zhou, Rebecca Serra Mari, Feyza Yilmaz, Xuefang Zhao, PingHsun Hsieh, Joyce Lee, Sushant Kumar, Jiadong Lin, Tobias Rausch, Yu Chen, Jingwen Ren, Martin Santamarina, Wolfram Höps, Hufsah Ashraf, Nelson T. Chuang, Xiaofei Yang, Katherine M. Munson, Alexandra P. Lewis, Susan Fairley, Luke J. Tallon, Wayne E. Clarke, Anna O. Basile, Marta Byrska-Bishop, André Corvelo, Uday S. Evani, Tsung-Yu Lu, Mark J. P. Chaisson, Junjie Chen, Chong Li, Harrison Brand, Aaron M. Wenger, Maryam Ghareghani, William T. Harvey, Benjamin Raeder, Patrick Hasenfeld, Allison A. Regier, Haley J. Abel, Ira M. Hall, Paul Flicek, Oliver Stegle, Mark B. Gerstein, Jose M. C. Tubio, Zepeng Mu, Yang I. Li, Xinghua Shi, Alex R. Hastie, Kai Ye, Zechen Chong, Ashley D. Sanders, Michael C. Zody, Michael E. Talkowski, Ryan E. Mills, Scott E. Devine, Charles Lee, Jan O. Korbel, Tobias Marschall, and Evan E. Eichler. Haplotype-resolved diverse human genomes and integrated analysis of structural variation. 372(6537):eabf7117. doi:10.1126/science.abf7117.

[19] Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval. URL: http://arxiv.org/abs/2102.05644, arXiv:2102.05644[cs].

[20] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning. URL: https://arxiv.org/abs/2203.10833v2.

[21] Yuan Gong, Yu-An Chung, and James Glass. AST: Audio spectrogram transformer. URL: http://arxiv.org/abs/2104.01778, arXiv:2104.01778[cs], doi:10.48550/arXiv.2104.01778.

[22] Sara Goodwin, John D. McPherson, and W. Richard McCombie. Coming of age: ten years of next-generation sequencing technologies. 17(6):333–351. Number: 6 Publisher: Nature Publishing Group. URL: https://www.nature.com/articles/nrg.2016.49, doi:10.1038/nrg.2016.49.

[23] Peiyong Guan and Wing-Kin Sung. Structural variation detection using next-generation sequencing data: A comparative technical review. 102:36–49. doi:10.1016/j.ymeth.2016.01.020.

[24] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. ISSN: 1063-6919. doi:10.1109/CVPR.2006.100.

[25] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on vision transformer. 45(1):87–110. URL: http://arxiv.org/abs/2012.12556, arXiv:2012.12556[cs], doi:10.1109/TPAMI.2022.3152247.

[26] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. URL: http://arxiv.org/abs/1912.12180, arXiv:1912.12180[cs], doi:10.48550/arXiv.1912.12180.

[27] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3463–3472. IEEE. URL: https://ieeexplore.ieee.org/document/9010392/, doi:10.1109/ICCV.2019.00356.

[28] Weichun Huang, Leping Li, Jason R. Myers, and Gabor T. Marth. ART: a next-generation sequencing read simulator. 28(4):593–594. doi:10.1093/bioinformatics/btr708.

[29] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. 54(10):200:1–200:41. doi:10.1145/3505244.

[30] Michael D Linderman, Daniel Brey, Peter Hansen, Yiran Shi, Alderik van der Heyde, Jacob Wallace, Eliza Wieman, Zahra Shamsi, Jeremiah Liu, Bruce D Gelb, and Ali Bashir. Deep metric learning for structural variant genotyping in genome sequencing data.

[31] Michael D Linderman, Crystal Paudyal, Musab Shakeel, William Kelley, Ali Bashir, and Bruce D Gelb. NPSV: A simulation-driven approach to genotyping structural variants in whole-genome sequencing data. 10(7):giab046. doi:10.1093/gigascience/giab046.

[32] Gary Marcus. Deep learning: A critical appraisal. URL: http://arxiv.org/abs/1801.00631, arXiv:1801.00631[cs,stat], doi:10.48550/arXiv.1801.00631.

[33] Aaron McKenna, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly, and Mark A. DePristo. The genome analysis toolkit: A MapReduce framework for analyzing next-generation DNA sequencing

data. 20(9):1297–1303. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2928508/, doi:10.1101/gr.107524.110.

[34] Shentong Mo, Xi Fu, Chenyang Hong, Yizhen Chen, Yuxuan Zheng, Xiangru Tang, Zhiqiang Shen, Eric P. Xing, and Yanyan Lan. Multi-modal self-supervised pre-training for regulatory genome across cell types. URL: http://arxiv.org/abs/2110.05231, arXiv:2110.05231[cs,q-bio], doi:10.48550/arXiv.2110.05231.

[35] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. URL: http://arxiv.org/abs/2003.08505, arXiv:2003.08505[cs], doi:10.48550/arXiv.2003.08505.

[36] Niki J. Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning (ICML)*, 2018. URL: http://proceedings.mlr.press/v80/parmar18a.html.

[37] Ryan Poplin, Pi-Chuan Chang, David Alexander, Scott Schwartz, Thomas Colthurst, Alexander Ku, Dan Newburger, Jojo Dijamco, Nam Nguyen, Pegah T. Afshar, Sam S. Gross, Lizzie Dorfman, Cory Y. McLean, and Mark A. DePristo. A universal SNP and small-indel variant caller using deep neural networks. 36(10):983–987. Number: 10 Publisher: Nature Publishing Group. URL: https://www.nature.com/articles/nbt.4235, doi:10.1038/nbt.4235.

[38] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2019/hash/3416a75f4cea9109507cacd8e2f2aefc-Abstract.html.

[39] Tobias Rausch, Thomas Zichner, Andreas Schlattl, Adrian M. Stütz, Vladimir Benes, and Jan O. Korbel. DELLY: structural variant discovery by integrated paired-end and split-read analysis. 28(18):i333–i339. doi:10.1093/bioinformatics/bts378.

[40] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. URL: https://arxiv.org/abs/1503.03832v3, doi:10.1109/CVPR.2015.7298682.

[41] Jonathan Sebat, B. Lakshmi, Dheeraj Malhotra, Jennifer Troge, Christa Lese-Martin, Tom Walsh, Boris Yamrom, Seungtai Yoon, Alex Krasnitz, Jude Kendall,

Anthony Leotta, Deepa Pai, Ray Zhang, Yoon-Ha Lee, James Hicks, Sarah J. Spence, Annette T. Lee, Kaija Puura, Terho Lehtimäki, David Ledbetter, Peter K. Gregersen, Joel Bregman, James S. Sutcliffe, Vaidehi Jobanputra, Wendy Chung, Dorothy Warburton, Mary-Claire King, David Skuse, Daniel H. Geschwind, T. Conrad Gilliam, Kenny Ye, and Michael Wigler. Strong association of de novo copy number mutations with autism. 316(5823):445–449. doi:10.1126/science.1138659.

[42] Zahra Shamsi, Drew Bryant, Jacob Wilson, Xiaoyu Qu, Avinava Dubey, Konik Kothari, Mostafa Dehghani, Mariya Chavarha, Valerii Likhosherstov, Brian Williams, Michael Frumkin, Fred Appelbaum, Krzysztof Choromanski, Ali Bashir, and Min Fang. Automated deep aberration detection from chromosome karyotype images. URL: http://arxiv.org/abs/2211.14312, arXiv:2211.14312[cs,q-bio].

[43] Jeonggeun Song and Heung-Chang Lee. X-ViT: High performance linear vision transformer without softmax. URL: http://arxiv.org/abs/2205.13805, arXiv:2205.13805[cs], doi:10.48550/arXiv.2205.13805.

[44] Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. Adaptive attention span in transformers. URL: http://arxiv.org/abs/1905.07799, arXiv:1905.07799[cs,stat], doi:10.48550/arXiv.1905.07799.

[45] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. URL: https://arxiv.org/abs/2012.12877v2.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. URL: http://arxiv.org/abs/1706.03762, arXiv:1706.03762[cs], doi:10.48550/arXiv.1706.03762.

[47] Joachim Weischenfeldt, Orsolya Symmons, François Spitz, and Jan O. Korbel. Phenotypic impact of genomic structural variation: insights from and for human disease. 14(2):125–138. Number: 2 Publisher: Nature Publishing Group. URL: https://www.nature.com/articles/nrg3373, doi:10.1038/nrg3373.

[48] Shu-wen Yang, Andy T. Liu, and Hung-yi Lee. Understanding self-attention of self-supervised audio transformers. URL: http://arxiv.org/

abs/2006.03265, arXiv:2006.03265[cs], doi:10.48550/arXiv.2006.03265.

[49] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10073–10082. IEEE. URL: https://ieeexplore.ieee.org/document/9156532/, doi:10.1109/CVPR42600.2020.01009.