TRANSFORMER-BASED FOUNDATION MODELS AND HIGH-PERFORMANCE COMPUTATIONAL TOOLS FOR CHROMATIN ACCESSIBILITY ANALYSIS

by

Nathan J. LeRoy

A DISSERTATION

Submitted to the School of Engineering and Applied Science
University of Virginia

In partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

December 2025

Acknowledgements

There are too many people for which I am grateful who got me to this point – I could not possibly list them all. However, I would like to specifically acknowledge a few key individuals.

First, my advisor, Dr. Nathan Sheffield, for his amazing guidance and support throughout my PhD journey. His mentorship has been invaluable, and his passion for science is truly inspiring.

To my parents, Bob and Lynn LeRoy. Your unwavering love and support have been the foundation of my success. Thank you for always believing in me and encouraging me to pursue my dreams.

To Alex and Donald, my labmates and friends. Your camaraderie and collaboration have made this journey so enjoyable and fulfilling. Thank you for the countless discussions, brainstorming sessions, and shared experiences. You've made me a better scientist, engineer, and person.

To Barnabas, my closest friend. Your unwavering support, encouragement, and friendship have meant the world to me. Thank you for always being there through the ups and downs. It feels like only yesterday we were discussing our dreams and aspirations at Purdue, and now here we are.

And finally, to Hannah, my partner in life. Your love, patience, and understanding have been my anchor throughout this journey. Thank you for believing in me and supporting me every step of the way. This dissertation is as much yours as it is mine.

Abstract

Chromatin accessibility profiling through scATAC-seq has emerged as a powerful tool for understanding gene regulation and cellular heterogeneity, yet existing analytical methods fail to leverage the nearly 10,000 datasets in the public repositories and remain computationally intensive for routine use. While existing tools adequately handle basic single-cell chromatin accessibility analysis, they are not architected to leverage modern deep learning approaches that could unlock insights across biological contexts and enable integration into advanced analytical workflows at scale. Recent advances in machine learning suggest a path forward. Transfer learning, particularly with large pre-trained models, has shown promise in scRNA-seq applications by enabling improved cell type clustering, identification, and dataset integration across studies. However, the application of transfer learning to scATAC-seq remains underexplored, despite similar structural and analytical challenges shared between the modalities.

This dissertation presents a comprehensive framework for scATAC-seq analysis that addresses these limitations through transformer-based transfer learning. The central innovation is to conceptualize genomic regions as discrete linguistic tokens, enabling direct adaptation of natural language processing techniques to epigenomic data. This approach facilitates the creation of open, generalizable foundation models that improve computational efficiency while enabling biological discovery across diverse datasets.

We first introduce the gtars toolkit, which provides efficient Rust-based utilities for creating consensus genomic interval vocabularies and tokenizing datasets into shared representations. These tools address critical infrastructure gaps in systematic vocabulary creation and efficient dataset mapping. Building on this foundation, we present scEmbed, a Word2Vec-inspired model that demonstrates the feasibility of tokenized chromatin accessibility modeling through fixed region embeddings that encode shared regulatory context. While scEmbed enables rapid clustering and cross-dataset cell-type annotation, its static embeddings reveal limitations in capturing cell-state-specific regulatory dynamics.

To address these constraints, we develop Atacformer, a transformer-based foundation model pre-trained on 1.2 million cells from 30 tissues. Through self-attention mechanisms, Atacformer generates contextualized representations that capture cell-level chromatin accessibility patterns and contextual dependencies within regulatory profiles. Unlike existing foundation models that use continuous representations and require large parameter counts, Atacformer maintains discrete token-level

embeddings, achieving comparable performance with dramatically fewer parameters while preserving interpretability. We further introduce CRAFT, a dual-encoder architecture pairing Atacformer with Geneformer to enable cross-modal alignment between scATAC-seq and scRNA-seq data.

Across multiple benchmarks, these approaches demonstrate strong performance in zero-shot clustering, annotation, and batch correction while revealing biologically meaningful regulatory relationships at the single-region level, including the discovery of weak promoter elements. Together, these contributions establish a scalable, transferable framework that bridges isolated experiments with unified biological insights, democratizing deep learning approaches for chromatin accessibility analysis and laying the groundwork for standardized analysis of chromatin accessibility data across diverse biological contexts.

Table of contents

List of figures

List of tables

Chapter 1: Introduction to gene regulation, ATAC-seq and current analytical challenges

A brief history of gene regulation and the study of chromatin accessibility

Why does the same DNA makes different cells?

Inside the nucleus of all 30 trillion of our cells is about six feet of DNA tightly packed and coiled up into compact structures called chromosomes[1]. Remarkably, this DNA sequence is nearly identical across all cell types within an individual, yet it gives rise to extraordinary levels of heterogeneity in cell types, tissues, disease states, and physiological responses. This fundamental paradox raises a critical question: *how does a single, static genetic blueprint generate such diverse biological outcomes*?

While the DNA sequence itself is largely invariant across cell types, its interpretation is highly dynamic and context-specific. Cells achieve this through complex layers of gene regulation, which govern when, where, and to what extent specific genes are transcribed into RNA. At the heart of this regulatory machinery are elements like promoters, which initiate transcription near gene start sites, and enhancers, which can activate transcription at distant genomic loci in a cell-type–specific manner[2]. These elements recruit combinations of transcription factors, activators, and RNA polymerases to modulate gene expression. However, for these proteins to access their target DNA sequences, the local chromatin must be in an *open* or accessible state. This is why these interactions are fundamentally influenced by the chromatin landscape, which determines the physical accessibility of DNA to regulatory proteins.[3] (Figure 1.1).

Briefly, promoters are DNA sequences located near gene start sites that serve as binding platforms for RNA polymerase and initiate transcription. Enhancers are regulatory sequences that can be located thousands of base pairs away from their target genes and boost transcription levels when bound by appropriate transcription factors. Transcription factors are proteins that bind to specific DNA sequences and either activate or repress gene expression. RNA polymerase is the enzyme responsible for transcribing DNA into RNA. Finally, chromatin refers to the complex of DNA and histone proteins that packages genetic material in the nucleus, with its structure determining which regions are accessible for transcription.

Thus, taken together, biological heterogeneity arises *not only* from subtle differences in genetic code but from the selective and combinatorial usage of regulatory elements across different cell types, developmental stages, and environmental conditions. Because of this, understanding gene

regulation is key to deciphering how the same DNA sequence can lead to diverse cellular identities and functions. In turn, this knowledge is critical for unraveling the molecular basis of health and disease, as dysregulation of these processes underlies many pathological conditions including cancer, developmental disorders, and immune dysfunction.

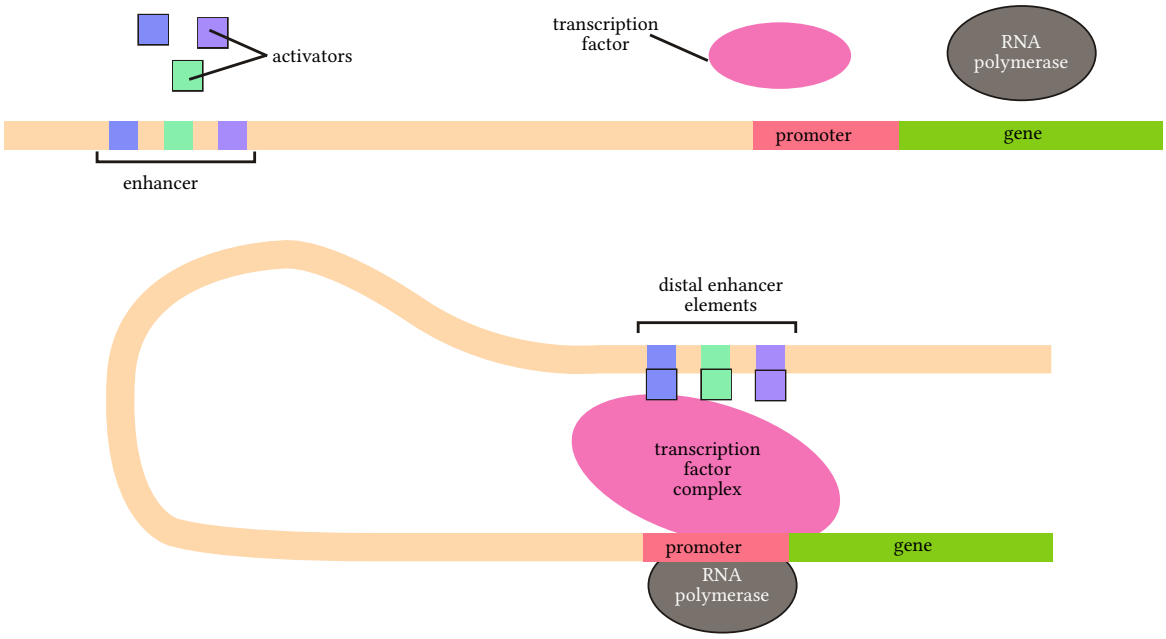


Figure 1.1: A schematic representation of gene regulation.

The rise of epigenomics assays

The completion of the Human Genome Project[4] in 2003 enabled various 'omics technologies which, as a result, facilitated the exploration of gene regulation and epigenomics. Simultaneously, genome-wide association studies (GWAS) revealed that approximately 90% of disease-associated single nucleotide polymorphisms (SNPs) are located in non-coding regions of the genome, underscoring the crucial importance of studying gene regulation and epigenomics in human health and disease for driving this biological heterogeneity[5,6]. This discovery catalyzed the development of numerous experimental approaches to study regulatory elements (Figure 1.2).

To comprehensively map the regulatory landscape, researchers have developed complementary experimental strategies that interrogate different aspects of chromatin biology. These approaches can be organized into three conceptual categories based on what they measure: (1) epigenetic modifications that mark regulatory states, exemplified by DNA methylation analysis; (2) chromatin accessibility and protein-DNA interactions that indicate active regulatory elements, including DNase-seq, FAIRE-seq, ATAC-seq, ChIP-seq, and CUT&TAG; and (3) three-dimensional genome organization that constrains regulatory interactions, captured through Hi-C. Together, these techniques provide a

multi-dimensional view of gene regulation. Each approach is described in detail below, organized by the biological question it addresses.

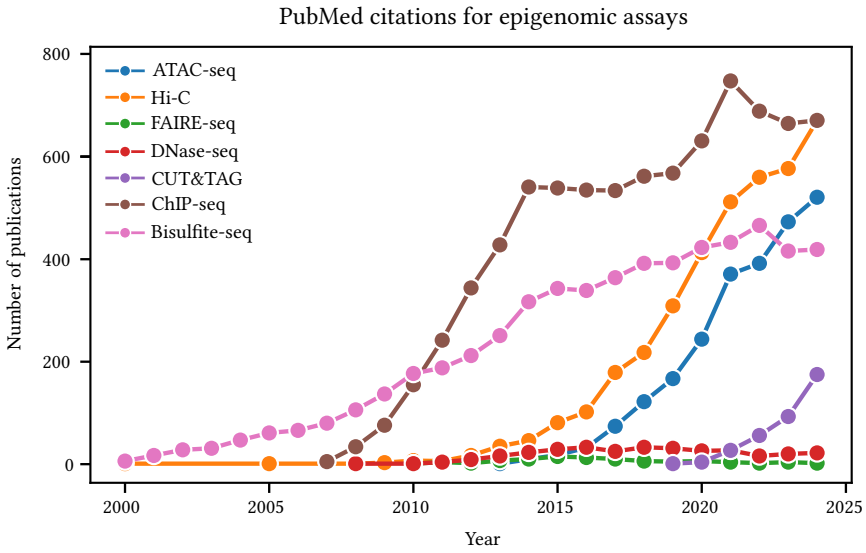


Figure 1.2: Trends in epigenomic assay publications over time. Data from PubMed searches for key epigenomic assays.

Mapping epigenetic modifications

DNA methylation represents one of the most stable epigenetic marks, serving as a foundational layer of gene regulation that can persist through cell division and development.

DNA methylation analysis through bisulfite sequencing

DNA methylation is a key epigenetic modification that plays a critical role in regulating gene expression by influencing chromatin structure and transcription factor accessibility[7,8]. DNA methylation analysis through bisulfite sequencing assesses cytosine methylation status at single-base resolution[9,10]. This technique involves treating DNA with bisulfite, which converts unmethylated cytosines to uracils, while leaving methylated cytosines unchanged. Subsequent sequencing of the treated DNA identifies methylated versus unmethylated cytosines, providing insights into the epigenetic regulation of gene expression through DNA methylation patterns.

Profiling protein-DNA interactions

Beyond static epigenetic marks, the dynamic binding of regulatory proteins to DNA determines which genes are actively transcribed. Two complementary approaches have emerged to map these interactions genome-wide.

Chromatin immunoprecipitation sequencing (ChIP-seq)

Transcription factors and histone modifications are central regulators of gene expression, and mapping their binding sites is essential for understanding gene regulatory networks[11,12]. ChIP-seq is a powerful technique used to analyze protein-DNA interactions on a genome-wide scale[13]. It involves

crosslinking proteins to DNA, followed by shearing the DNA and immunoprecipitating the protein of interest using specific antibodies. The associated DNA is then purified and sequenced, to identify binding sites and regulatory elements.

CUT&TAG

Building on the principles of ChIP-seq, CUT&TAG offers a more refined approach with improved sensitivity and reduced input requirements. CUT&TAG is a method used to profile protein-DNA interactions with high resolution and low input requirements[14]. It involves the use of a Tn5 transposase fused to an antibody that targets a specific protein of interest. This allows for the selective tagging and subsequent sequencing of accessible chromatin regions associated with the protein, providing insights into its regulatory role. Compared to traditional ChIP-seq, CUT&TAG offers improved signal-to-noise ratios and reduced background, making it particularly valuable for studying low-abundance proteins or working with limited cell numbers.

Measuring chromatin accessibility

While protein binding studies reveal where regulatory factors associate with DNA, chromatin accessibility assays provide a broader view of which genomic regions are available for regulatory interactions. These techniques have evolved from enzymatic digestion approaches to transposase-based methods, each offering unique advantages for mapping open chromatin.

DNase hypersensitivity sequencing (DNase-seq)

Because regulatory elements such as promoters and enhancers reside in regions of open chromatin, mapping DNase I hypersensitive sites has long been a key strategy for identifying functional regulatory elements across the genome[15]. DNase-seq is a technique used to map regions of open chromatin by identifying sites of DNase I hypersensitivity[16]. It involves treating nuclei with DNase I, which cleaves accessible DNA, followed by sequencing the resulting fragments. This method provides insights into chromatin accessibility and the regulatory landscape of the genome.

Formaldehyde-Assisted Isolation of Regulatory Elements (FAIRE-seq)

FAIRE-seq leverages the fact that active regulatory elements are often nucleosome-depleted, providing a complementary approach to DNase-based assays for detecting accessible chromatin. FAIRE-seq is a method used to identify nucleosome-depleted regions of the genome, which are indicative of regulatory elements[17]. It involves crosslinking chromatin with formaldehyde, followed by phenol-chloroform extraction to isolate DNA from nucleosome-depleted regions. The purified DNA is then sequenced to map these regulatory elements.

Assay for Transposase-Accessible Chromatin using sequencing (ATAC-seq)

Representing a technical advance over both DNase-seq and FAIRE-seq, ATAC-seq combines simplicity with sensitivity to provide robust accessibility maps. The Assay for Transposase-Accessible Chromatin

using sequencing (ATAC-seq)[18] is a method used to assess chromatin accessibility. It involves the use of a hyperactive Tn5 transposase to insert sequencing adapters into accessible regions of the genome. This technique allows for the identification of open chromatin regions and the characterization of regulatory elements. ATAC-seq has become a widely adopted method due to its simplicity, speed, and ability to generate high-resolution maps of chromatin accessibility from small amounts of input material.

Capturing three-dimensional genome organization

The spatial organization of chromatin in three dimensions adds another critical layer of gene regulation, as physical proximity between distant genomic elements enables long-range regulatory interactions.

Chromatin Conformation Capture (Hi-C)

Because genome folding constrains which regulatory elements can physically contact their target genes, Hi-C provides a powerful way to link chromatin architecture with gene regulation[19,20]. Hi-C is a technique used to study the three-dimensional organization of the genome. It involves crosslinking chromatin, followed by digestion with a restriction enzyme and ligation of the resulting fragments. This method identifies interactions between distant genomic regions, providing insights into the spatial organization of the genome[21]. Specifically, Hi-C can identify topologically associating domains (TADs) and chromatin loops that bring enhancers into close proximity with their target promoters.

Integration and single-cell advances

The techniques described above provide complementary views of gene regulation, from epigenetic modifications to protein binding, chromatin accessibility, and three-dimensional organization. Importantly, many of these approaches have been successfully adapted to single-cell formats[22–24], enabling the dissection of regulatory heterogeneity within complex tissues and the identification of rare cell states that would be masked in bulk measurements. These methods are summarized in Table 1.1.

Table 1.1: Summary of epigenomic assays and their biological insights

Epigenomic assay	What it measures
DNA Methylation	Cytosine methylation status via bisulfite conversion
DNase-seq	Chromatin accessibility via DNase I cleavage
ChIP-seq	Protein-DNA binding via immunoprecipitation
ATAC-seq	Chromatin accessibility via Tn5 transposase insertion
FAIRE-seq	Nucleosome-depleted regions via phenol-chloroform extraction
CUT&Tag	Targeted chromatin profiling via antibody-Tn5 fusion
Hi-C	3D chromatin conformation via proximity ligation

Assay for Transposase-Accessible Chromatin using sequencing (ATAC-seq) and its single-cell counterpart

Among these assays, ATAC-seq has emerged as a widely adopted method for studying gene regulation due to its simplicity, speed, and ability to generate high-resolution maps of chromatin accessibility from small amounts of input material (Figure 1.2). Unlike earlier methods, it requires no antibodies or extensive sample prep, making it especially well-suited for rare or primary cells. Its adaptability to single-cell formats has enabled high-throughput profiling of cell-type–specific regulatory landscapes, fueling discoveries in development, cancer, and immunology. By revealing open chromatin regions that mark active promoters, enhancers, and other regulatory elements, ATAC-seq provides critical insights into how gene expression is controlled across diverse biological contexts[18].

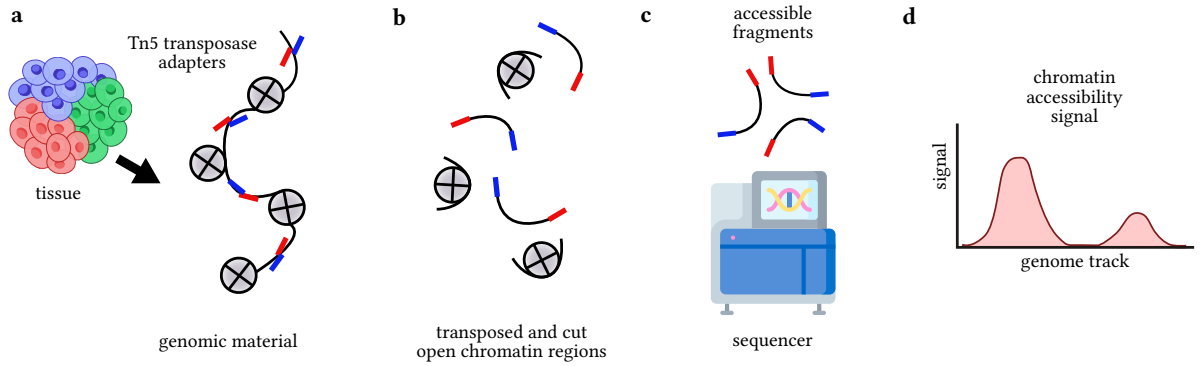


Figure 1.3: Overview of the ATAC-seq procedure.

The fundamental goal behind ATAC-seq is to identify regions of open chromatin across the genome, which are indicative of regulatory elements such as promoters and enhancers that control gene expression. The assay leverages the hyperactive Tn5 transposase enzyme, which preferentially inserts sequencing adapters into accessible regions of the genome where nucleosomes are absent or loosely bound[18] (Figure 1.3A). The Tn5 transposase simultaneously cleaves the DNA and ligates sequencing adapters in a process known as "tagmentation." This results in a library of DNA fragments that are significantly enriched for open chromatin regions (Figure 1.3B).

These DNA fragments are then PCR-amplified to add sample-specific barcodes and sequencing primers, followed by high-throughput sequencing, typically using Illumina platforms (Figure 1.3C). The resulting sequencing reads are aligned to a reference genome, and peaks of read enrichment are identified using computational tools such as MACS2[25]. These peaks correspond to regions of accessible chromatin, providing insights into the regulatory landscape of the cell type under study (Figure 1.3D).

ATAC-seq remains a staple in epigenomic research, however, its adaptation to single-cell formats (scATAC-seq) has revolutionized the field by enabling the profiling of chromatin accessibility at

the resolution of individual cells[24,26]. This advancement allows researchers to dissect the regulatory heterogeneity *within* seemingly homogeneous cell populations, identify rare cell types, and trace developmental trajectories based on chromatin dynamics. In scATAC-seq, individual cells are isolated using microfluidics or droplet-based platforms, and the ATAC-seq protocol is adapted to work with the limited DNA content of single cells. The resulting data provides cell-type–specific accessibility profiles that can be integrated with single-cell RNA sequencing to create comprehensive regulatory maps linking chromatin state to gene expression across diverse cell types and developmental stages.

Computational challenges in single-cell ATAC-seq analysis

The high-dimensionality and inherent sparsity of scATAC-seq data

While scATAC-seq has opened new avenues for understanding gene regulation, it also presents unique challenges, such as increased technical variability and the need for specialized computational tools to analyze the resulting data. Addressing these challenges is crucial for fully realizing the potential of single-cell epigenomics in uncovering the complexities of gene regulation.

A scATAC-seq dataset is often represented as a binary matrix where rows correspond to cells, and columns correspond to genomic loci (Figure 1.4). These matrices are frequently characterized by their high dimensionality and inherent sparsity, posing significant challenges for computational analysis. Because each cell only contains two copies of each chromosome, and because the assay measures accessibility at upwards of 1 million genomic loci, the resulting data matrix is often extremely sparse and high-dimensional. It is not uncommon for matrices to exceed a hundred thousand rows and one million columns with only a small fraction of the entries filled. This sparsity can make it difficult to accurately infer regulatory interactions and identify cell-type-specific patterns of chromatin accessibility. Furthermore, the high dimensionality of scATAC-seq data can make traditional statistical methods less effective[26] while simultaneously presenting significant computational challenges both in terms of memory and processing power.

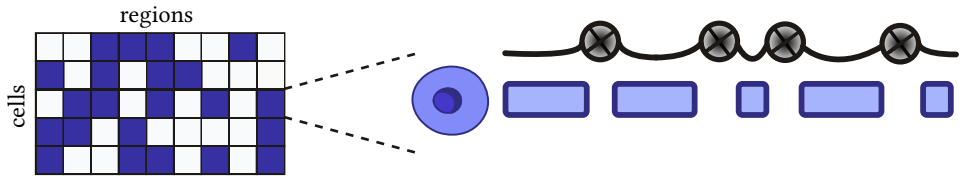


Figure 1.4: Schematic of a scATAC-seq count matrix.

To address these challenges, researchers have developed new computational methods and tools specifically designed for scATAC-seq data. These approaches aim to improve data normalization, enhance signal detection, and facilitate the integration of scATAC-seq data with other single-

cell modalities, such as scRNA-seq. By leveraging advances in machine learning and statistical modeling, these methods hold promise for unlocking the full potential of single-cell epigenomics in understanding gene regulation.

Traditional single-cell ATAC-seq analysis methods

Soon after the onset of scATAC-seq, several computational tools were developed to address the unique challenges posed by the data modality. Among those were chromVAR[27], BROCKMAN[28], Cicero[29], cisTopic[30], SnapATAC[31], SCRAT[32], and EpiScanpy[33]. Each of these tools employs distinct strategies to extract regulatory and cell-type information.

Peak-based variability analysis: chromVAR

chromVAR focuses on estimating the variability of chromatin accessibility across peaks that share common features, such as transcription factor motifs or specific k-mers. This method enables researchers to identify regulatory elements that show differential accessibility across cell types or conditions by leveraging known sequence motifs and annotations. By computing deviation scores for each motif in each cell, chromVAR provides insights into the activity of specific transcription factors across single-cell profiles[27].

Sequence-based dimensionality reduction: BROCKMAN

BROCKMAN represents genomic sequences using gapped k-mers within transposase integration sites and applies principal component analysis (PCA) to capture variation in k-mer occupancy. This approach bypasses the need for peak calling by directly analyzing the sequence context of ATAC-seq insertions. The method's focus on k-mer patterns allows it to capture regulatory signals that might be missed by peak-centric approaches, providing a complementary perspective on chromatin accessibility patterns. By treating accessibility as a sequence-based problem rather than a peak-based one, BROCKMAN can identify subtle regulatory motifs and sequence preferences that drive chromatin accessibility differences between cell types.

Gene activity scoring and regulatory networks: Cicero

Cicero leverages and estimates gene activity scores at each peak by computing accessibility at promoters with the regulatory potential of nearby peaks. This method goes beyond simple accessibility measurements by attempting to link distal regulatory elements to their target genes through co-accessibility analysis. Cicero constructs regulatory networks by identifying peaks that are frequently accessible together, enabling the prediction of gene regulatory relationships and the identification of putative enhancer-promoter interactions.

Topic modeling approaches: cisTopic

cisTopic uses latent Dirichlet allocation (LDA), a Bayesian topic modeling method originally developed for natural language processing, to simultaneously uncover cell states and putative regulatory regions.

In this framework, cells are treated as documents and accessible regions as words, allowing the method to identify "topics" that represent co-accessible regulatory programs. This probabilistic approach enables the discovery of regulatory modules and cell types while providing interpretable results about the underlying regulatory architecture.

Latent semantic indexing approaches

Another class of approaches, which follow the Cusanovich2018[34] pipeline, employs latent semantic indexing (LSI). These methods begin by partitioning the genome into windows, normalizing read counts using the term frequency–inverse document frequency (TF-IDF) transformation. Dimensionality is then reduced using singular value decomposition (SVD), and a first round of clustering—termed "in silico cell sorting"—is performed to identify clades and call peaks within them. A second round of TF-IDF and SVD is applied using read counts in the called peaks to refine the clusters. This technique, borrowed from natural language processing, is crucial because it down-weights common, broadly accessible regions (equivalent to 'the' or 'a' in text) while up-weighting rare regions that are uniquely accessible in specific cell types, thereby highlighting cell-type–defining features. This iterative approach combines dimensionality reduction with peak calling to improve both cell clustering and regulatory region identification.

SnapATAC: Bin-based analysis with regression normalization

SnapATAC segments the genome into uniform bins and corrects for library size differences using regression-based normalization, followed by PCA to identify key components for clustering analysis. Specifically, regression-based normalization is employed to adjust for technical confounders such as sequencing depth and mitochondrial read fraction, which can introduce biases in the data. This normalization step helps to mitigate the effects of these confounders, allowing for a more accurate representation of the underlying biological signal. Following normalization, PCA is applied to reduce the dimensionality of the data and identify the principal components that capture the most variance. These components are then used for clustering analysis, enabling the identification of distinct cell populations based on their chromatin accessibility profiles. Unlike peak-based methods, this approach attempts to avoid potential biases introduced by peak calling algorithms by analyzing accessibility in fixed genomic windows. The regression-based normalization specifically addresses technical confounders such as sequencing depth and mitochondrial read fraction, making it particularly robust for large-scale comparative analyses.

End-to-end pipelines and more recent methods

Still, more methods have continued to emerge, incorporating more robust statistical frameworks, end-to-end pipelines, and novel algorithms for dimensionality reduction, clustering, and visualization.

Examples of this include ArchR[35]: a comprehensive software suite for end-to-end analysis of single-cell chromatin accessibility written in R, Signac[36]: an extension of the Seurat framework for the analysis of single-cell chromatin data, and SnapATAC2[37]: a rust-based package for fast, memory-efficient analysis of scATAC-seq data that leverages a novel spectral embedding dimensionality reduction algorithm called spectral embedding. SnapATAC2, in particular was one of the first python-native tools for scATAC-seq, targeting atlas-scale datasets with millions of cells. As such, it incorporates a number of algorithmic improvements to enable faster processing and lower memory usage. It was recently integrated into the `scverse` ecosystem as part of the `scvi-tools` package[38], which provides a suite of tools for probabilistic modeling and analysis of single-cell omics data.

Together, these tools have significantly advanced the field of single-cell epigenomics, enabling extraction of meaningful biological insights from complex scATAC-seq datasets. Researchers now have access to a diverse array of computational resources in many programming languages and environments, facilitating the study of gene regulation at single-cell resolution. As the field continues to mature and evolve, this has paved the way for more sophisticated methods, including deep learning-based approaches to extract even more intricate patterns of chromatin accessibility.

Deep learning-based methods for scATAC-seq analysis

As deep learning revolutionizes fields like computer vision (CV) and natural language processing (NLP), researchers are exploring its application to single-cell genomics, including scATAC-seq analysis. Deep learning models, particularly those based on neural networks, show promise in capturing complex patterns and relationships within high-dimensional data. The computational methods developed for scATAC-seq analysis can be conceptually organized into two primary paradigms based on their underlying modeling strategy: (1) generative models that learn latent representations of chromatin accessibility patterns from peak-by-cell matrices, exemplified by SCALE and PeakVI; and (2) sequence-based discriminative models that leverage DNA sequence information to predict accessibility, as demonstrated by scBasset. These complementary approaches address different aspects of the scATAC-seq analysis challenge and offer distinct advantages for understanding cellular heterogeneity and regulatory mechanisms.

Generative modeling approaches

Generative models for scATAC-seq data learn probabilistic representations of chromatin accessibility by modeling the underlying distribution of peak-by-cell observations. These methods employ variational inference frameworks to capture complex dependencies in the data while enabling dimensionality reduction, denoising, and clustering.

SCALE: A variational autoencoder for scATAC-seq

One of the first deep learning methods developed for scATAC-seq data was SCALE[39], which employs a deep generative model to learn a low-dimensional representation of scATAC-seq data. SCALE uses a variational autoencoder (VAE) architecture to model the underlying distribution of chromatin accessibility profiles, enabling effective dimensionality reduction and clustering of cells based on their accessibility patterns. SCALE takes as input a binarized peak-by-cell matrix and learns a latent representation that captures the key features of the data while accounting for technical noise and variability. The model is trained using a combination of reconstruction loss and a regularization term that encourages the latent space to follow a prior distribution, typically a standard normal distribution. This approach allows SCALE to effectively denoise the data and identify meaningful biological variation among cells.

PeakVI: A variational inference model with batch correction

Building on the generative modeling framework established by SCALE, PeakVI introduces explicit handling of technical variation and batch effects while decomposing accessibility observations into interpretable components. This method leverages variational inference with deep neural networks to model scATAC-seq data by decomposing chromatin accessibility observations into three key components[40]. PeakVI employs a variational autoencoder architecture where an encoder network infers latent representations from observed data, a decoder network generates accessibility probability estimates, and auxiliary neural networks estimate region-specific scaling factors. This approach enables batch correction, normalized visualization, and clustering while providing interpretable estimates of the true chromatin landscape by separating biological signal from technical confounders.

Sequence-based discriminative approaches

While generative models learn from observed accessibility patterns, sequence-based approaches leverage the DNA sequence itself as input, enabling predictions for arbitrary genomic regions and capturing sequence-specific regulatory logic.

scBasset: A convolutional neural network for sequence-based chromatin accessibility

Representing a fundamentally different modeling paradigm, scBasset[41] leverages a convolutional neural network (CNN) architecture to learn sequence-based representations of chromatin accessibility. scBasset is trained on the actual DNA sequences underlying accessible regions, allowing it to capture sequence motifs and patterns associated with regulatory elements. This approach enables improved prediction of chromatin accessibility and identification of cell-type-specific regulatory features. One key advantage of scBasset is its ability to analyze new, unseen genomic regions based on their sequence content, making it more flexible and generalizable compared to methods that rely solely on predefined

peak sets. The model is trained using a binary cross-entropy loss function, optimizing the network to accurately predict accessibility status based on sequence input.

Limitations of current approaches

While these methods demonstrate the power of deep learning for scATAC-seq analysis, they also face important limitations that constrain their applicability. SCALE is designed to be retrained with each new dataset, limiting its ability to generalize across experiments and requiring substantial computational resources for each analysis. scBasset, while leveraging sequence information to enable predictions on novel genomic regions, does not directly model cell-to-cell variability in accessibility profiles, which is a fundamental characteristic of single-cell data. PeakVI addresses batch effects more explicitly than SCALE but still requires dataset-specific training. Moreover, all of these methods require substantial computational resources for both training and inference, which can be a barrier for widespread adoption. These limitations highlight the need for more flexible and efficient approaches to scATAC-seq analysis that can generalize across datasets while still leveraging the representational power of deep learning.

Large, pre-trained foundation models for genomic data

Foundation models in genomics

One of the most exciting developments in machine learning has been the emergence of large, pre-trained models that can be fine-tuned for specific tasks with relatively little additional data. These models, often referred to as "foundation models," have demonstrated remarkable performance across a wide range of applications, from image recognition to natural language understanding[42–49].

The emergence of foundation models in computer vision (CV) and NLP has inspired similar efforts in the field of genomics. Typically built on the transformer architecture[50], these models are pre-trained on large-scale genomic datasets and can be used in a variety of downstream tasks for zero-shot predictions, few-shot learning, and transfer learning. Examples include DNABERT[51]: bidirectional encoder for DNA sequences, Enformer: a model that predicts gene expression from DNA sequence[52], AlphaFold: a model for predicting protein structures from amino acid sequences[53–55], DNA Discrete Diffusion: a model for generating DNA sequences with desired properties[56], scGPT: a transformer-based model for single-cell RNA-seq data[57], and Geneformer: a transformer-encoder model for scRNA-seq data[58]. Overwhelmingly, these models will be pre-trained on large, diverse datasets and then leveraged for transfer learning to apply them to new, unseen datasets and tasks. This makes them particularly attractive as they can be fine-tuned with relatively little data and computational resources compared to training a model from scratch which is often infeasible for many researchers.

Foundation models for single-cell ATAC-seq data

While many models exist for DNA sequences, protein sequences, and scRNA-seq data, until very recently, there has been a lack of similar efforts focused on scATAC-seq data. This is likely due to the unique challenges posed by scATAC-seq data, including its high dimensionality, sparsity, and the complex relationship between chromatin accessibility and gene regulation. Moreover, scATAC-seq data is less ubiquitous than other types of genomic data, making it much more difficult to obtain large, diverse training datasets. Beyond the complexity and scarcity of data, however, lies an even more fundamental obstacle: *the lack of a universal 'vocabulary' for chromatin accessibility.*

A note on tokenization

Regardless of data modality (text, images, DNA sequences), tokenization is a critical first step for all foundation models, as it defines how raw, disparate inputs are mapped into a shared feature space that the model can operate on. In natural language processing, this is achieved through subword vocabularies that segment text into units like words or byte-pair encodings, creating a stable and universal representation of language[59–61]. In computer vision, images are typically partitioned into fixed-size patches that act as tokens, enabling transformers to process them as sequences.

Genomics has followed suit: proteomics and transcriptomics benefit from clear, biologically grounded vocabularies — amino acids for proteins, and nucleotides or k-mers for RNA. Supported by established gene ontologies, this makes it comparatively easy to build well-defined vocabularies and tokens for a model to process. By contrast, single-cell ATAC-seq presents a uniquely difficult tokenization problem: *the natural unit of information in ATAC-seq, a genomic interval, is not standardized across experiments or datasets, and "shared vocabularies" can vary dramatically depending on how peaks are called or which reference regions are chosen.* This lack of a canonical token set has delayed the emergence of foundation models for scATAC-seq, as the success of such models depends on a stable, unified representation of input features across diverse datasets. Despite this significant hurdle, the landscape has begun to change with the recent emergence of foundation models specifically designed to tackle this tokenization challenge.

The first scATAC-seq foundation models: EpiAgent and ChromFound

Foundation models for scATAC-seq data have been scarce. However, this landscape has begun to change with the recent emergence of foundation models specifically designed for chromatin accessibility data. Within the last nine months, two notable methods have been introduced: EpiAgent[62] and ChromFound[63]. These models represent the first serious attempts to develop large-scale, pre-trained foundation models for scATAC-seq analysis, potentially addressing many of the limitations faced by previous approaches while leveraging the power of modern deep learning architectures.

However, these models exhibit fundamental architectural limitations that constrain their flexibility and applicability. ChromFound explicitly relies on continuous accessibility embeddings rather than discrete tokens, sidestepping a crucial architectural innovation of transformers that enables interpretability and computational efficiency. EpiAgent, while using discrete cCRE tokens, employs a TF-IDF ranking scheme that requires access to the entire cell-by-peak matrix to determine token ordering for each individual cell. This matrix-level dependency prevents these models from operating on individual cells in isolation, precluding applications such as bulk ATAC-seq analysis, individual cell-level queries, or efficient vector database lookups for nearest-neighbor classification. These architectural constraints, combined with additional practical limitations, restrict the broader adoption and utility of current scATAC-seq foundation models.

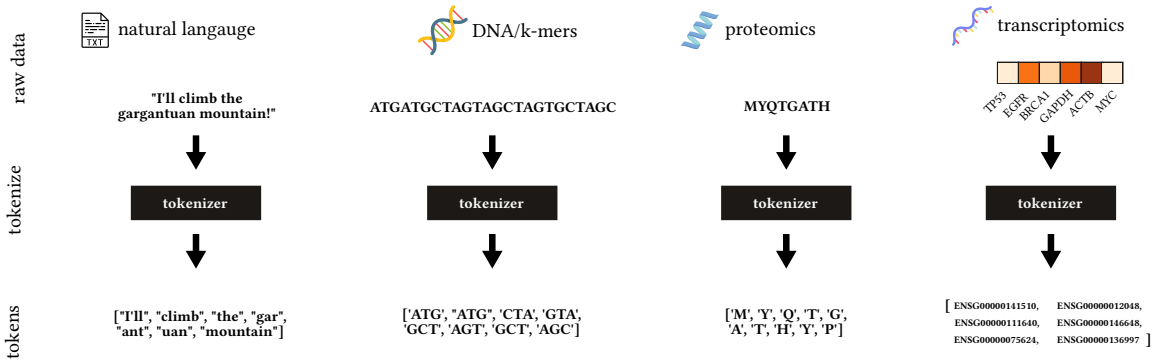


Figure 1.5: Overview of tokenization methods for various modalities.

Limitations of current scATAC-seq foundation models: EpiAgent and ChromFound

Overview of EpiAgent

EpiAgent is a transformer-based foundation model trained on 5 million single cells or 35 billion tokens from 31 tissues. The model has about 1.4 billion parameters, divided between an embedding module, an 18-layer bidirectional transformer, and a signal decoder. EpiAgent tokenizes cells by converting each cell's accessible cCREs into a "cell sentence," ranking cCREs by TF-IDF values and limiting input length to 8,192 tokens. EpiAgent uses two novel training tasks for pre-training: cell-cCRE alignment (classifying whether cCREs are accessible given the cell embedding) and signal reconstruction (rebuilding accessibility signals from embeddings). The model outputs contextualized cCRE embeddings and a `[CLS]` cell embedding that capture cellular heterogeneity and regulatory networks. These representations can be adapted for downstream tasks like unsupervised feature extraction, supervised annotation, data imputation, perturbation prediction, batch correction, query mapping, and even in-silico cCRE knockout analysis

Overview of ChromFound

Similarly, ChromFound is a foundation model specifically designed for scATAC-seq data. It was pretrained on 1.97 million cells spanning 30+ tissues and 6 disease conditions, using a massive 1.86 trillion training tokens. The architecture is hybrid, combining a Window Partition Self-Attention (WPSA) module to capture local enhancer–promoter dependencies within ±200 kb of transcription start sites and a Mamba block[64] for efficient long-range context processing. Its genome-aware tokenization encodes each open chromatin region with three components: chromosome identity, precise genomic coordinates, and continuous (non-binary) accessibility values. ChromFound is pre-trained using a masked reconstruction objective that predicts both zero and non-zero accessibility values, mitigating sparsity and preserving non-binary regulatory information. The model outputs contextualized OCR embeddings and low-dimensional cell embeddings that can be applied in zero-shot or fine-tuned settings for tasks such as cell clustering, batch correction, cell type annotation, cross-omics prediction (inferring gene expression from ATAC), and enhancer–gene link discovery.

Limitations

While both methods have shown promise and broken new ground in the field of scATAC-seq, they still suffer from substantial limitations. First, these methods rely on large, cumbersome models that exceed 1 billion parameters. This scale rivals even some large language models today, making them computationally intractable even with significant compute and GPU resources. Second, these models lack accessibility and reusability. ChromFound, to date, remains closed source, making it impossible to actually use or evaluate. EpiAgent, while open source, lacks a user-friendly system for accessing pre-trained models. The convention in CV and NLP is to share these models through platforms like Hugging Face, which neither EpiAgent nor ChromFound do, making fine-tuning on new datasets near impossible for most researchers.

Third, and critically, these models exhibit fundamental architectural constraints that limit their flexibility. ChromFound uses continuous accessibility values as token embeddings rather than discrete tokens, bypassing the interpretability and computational benefits that discrete tokenization provides in successful foundation models across other domains. While EpiAgent does employ discrete cCRE tokens, its TF-IDF ranking scheme requires the entire cell-by-peak matrix to determine which tokens represent each cell and in what order. This matrix-level dependency means the model cannot process individual cells in isolation. Consequently, these models cannot be applied to bulk ATAC-seq data, cannot perform efficient individual cell queries against vector databases for nearest-neighbor classification, and cannot analyze small numbers of cells independently without the full matrix

context. This fundamentally limits their utility for real-world applications where researchers may want to analyze a single new sample, compare individual cells, or integrate data incrementally.

Finally, both models produce dense, high-dimensional cell embeddings that lack interpretability and explainability. While these embeddings may capture biological variation, they do not provide clear insights into which specific regulatory elements or chromatin features drive cellular identity. In contrast, a foundation model that combines truly discrete tokenization with cell-level independence could achieve comparable performance while enabling direct interpretation of individual tokens, supporting flexible deployment across diverse experimental contexts, and dramatically reducing computational complexity.

Summary of computational methods for scATAC-seq analysis

Below is a summary of the key computational methods discussed in this chapters:

Table 1.2: Summary of computational approaches for scATAC-seq data

Method	Released	Description	Language	Type
chromVar	2017	Quantifies variability in chromatin accessibility across TF motifs/k-mers.	R	Tools
BROCKMAN	2018	Uses gapped k-mers + PCA to capture sequence determinants of accessibility.	R	Tools
scABC	2018	Clusters scATAC-seq by weighted k-means and identifies cell-type–specific peaks.	R	Tools
Cicero	2018	Infers gene activity scores and enhancer–promoter connections from scATAC-seq.	R	Tools
cisTopic	2019	Topic modeling (LDA) to uncover regulatory programs from accessibility data.	R, Python	Tools
ArchR	2021	End-to-end scalable scATAC-seq analysis toolkit (QC, clustering, peak calling).	R	End-to-end
Signac	2021	Seurat extension for multi-omic single-cell chromatin accessibility analysis.	R	End-to-end
SnapATAC	2021	Scalable clustering + integration framework for scATAC-seq profiles.	R	End-to-end
EpiScanpy	2021	Extends Scanpy for single-cell epigenomic assays (ATAC, DNA meth, etc.).	Python	End-to-end
SnapATAC2	2023	Rust-backed, optimized reimplementation of SnapATAC with high scalability.	Python	End-to-end
SCALE	2019	Variational autoencoder for scATAC-seq data dimensionality reduction.	Python	Deep learning
scBasset	2022	Sequence-based CNN for scATAC-seq data analysis and motif discovery.	Python	Deep learning
PeakVI	2022	Variational autoencoder with emphasis on batch correction and interpretability.	Python	Deep learning
EpiAgent	2024	Transformer foundation model for scATAC-seq analysis tasks.	Python	Foundation model
ChromFound	2025	Large-scale foundation model for multi-omic chromatin accessibility integration.	Python	Foundation model

Improving model sharing, efficiency, and flexibility of scATAC-seq foundation models

This dissertation advances the field of scATAC-seq analysis in three distinct ways:

First, it introduces a novel tokenization framework for training and running scATAC-seq models, along with a suite of high-performance, Rust-based infrastructure to support these methods, including vocabulary builders and fast tokenizers designed to enable highly efficient and scalable data processing.

Second, it presents scEmbed, a shallow neural network based on Word2Vec that follows the discrete token approach proposed in previous sections to first assess the feasibility of tokenized chromatin accessibility modeling. This provides a proof-of-concept for model sharing and the re-use of pre-trained genomic region embeddings, thereby streamlining downstream analysis and reducing computational overhead. We show that pre-trained region embeddings learned with scEmbed can be effectively utilized in various downstream tasks, improving performance and reducing the need for extensive retraining.

Finally, it develops Atacformer, a transformer-based foundation model for chromatin accessibility that incorporates our novel tokenization strategy and pushes the discrete token approach to its limit, analyzing its ability to extract meaningful biological insights at the single-region level. This design emphasizes flexibility and efficiency, achieving comparable performance with dramatically fewer parameters than existing approaches. Moreover, our new infrastructure facilitates easier model sharing and deployment and our discrete token framework enables biological insights and interpretability to discover regulatory mechanisms. Together, these contributions provide both methodological foundations and practical tools that move the field toward unified, generalizable models for single-cell epigenomic analysis.

Chapter 2: Background and related work in natural language processing (NLP)

Preface

This dissertation explores the intersection of natural language processing (NLP) and single-cell chromatin accessibility analysis, focusing on the development of transformer-based transfer learning approaches tailored for scATAC-seq data. To contextualize this interdisciplinary approach, it is essential to establish a foundational understanding of NLP's evolution and current state. The field of NLP has undergone remarkable transformation over the past decades, progressing from rule-based systems and statistical methods to the current era of deep learning and large-scale transformer architectures. This historical progression not only illuminates the technological advances that make modern NLP applications possible but also provides crucial context for understanding how established NLP methodologies can be adapted and applied to novel domains such as genomics. By grounding this work in NLP's developmental trajectory, we can better appreciate both the opportunities and challenges inherent in transferring language processing techniques to analysis of gene regulation.

The rise of natural language processing

Natural language processing (NLP) has long served as both a proving ground and a catalyst for innovations in machine learning. The field's central challenge — learning meaningful representations from sequential data — closely parallels problems found in domains like biology, where genomic sequences or chromatin accessibility profiles can be viewed as "languages." Over the past two decades, NLP has undergone a dramatic shift: from early statistical models based on word counts, to distributed word embeddings like Word2Vec, to recurrent neural networks and their gated variants, and finally to transformers, which have redefined what is possible in sequence modeling. Tracing this trajectory illuminates the conceptual breakthroughs that led to today's foundation models, while simultaneously providing the intellectual scaffolding for adapting these methods to biological data such as scATAC-seq.

This chapter provides a concise overview of key developments in NLP that have shaped modern machine learning as well as our own work on genomic interval analysis. We will, in turn, discuss the evolution of word embeddings, the advent of attention mechanisms and transformers, and the rise of large pre-trained models. By understanding these milestones, we can better appreciate how techniques originally designed for human language can be repurposed to decode the "language" of genome regulation.

Word2Vec and word embeddings

From words to numbers: the challenge of word representation

A core challenge in natural language processing is how to represent words in a way that a machine learning model can understand. Fundamentally, a neural network is only capable of processing numerical data. This means that before we can apply neural networks to text, we need a way to convert words into numbers. Specifically this means converting something like the sentence: `"The cat sat on the mat."` into a numerical format that a neural network can understand. This is not a trivial task, as words are discrete symbols that do not have an inherent numerical representation.

One-hot encoding

A simple and intuitive way to represent words numerically is through one-hot encoding. In this approach, we first create a vocabulary of all unique words in our dataset. Each word is then represented as a vector of zeros with a single one at the index corresponding to that word in the vocabulary. For example, if our vocabulary consists of the words `["the", "cat", "sat", "on", "mat"]`, the word `"cat"` would be represented as such (Figure 2.1A):

```
[0, 1, 0, 0, 0]
```

While one-hot encoding is straightforward, it has several limitations. First, it results in high-dimensional and sparse vectors, especially for large vocabularies. Second, it does not capture any semantic relationships between words; for instance, “cat” and “dog” would be represented as completely orthogonal vectors despite their semantic similarity. Interestingly, this limitation parallels challenges found in single-cell ATAC-seq data, where binary count matrices represent the accessibility of genomic regions in a similar one-hot fashion – each cell is represented by a sparse vector indicating which peaks are accessible (1) or inaccessible (0). Just as with words, this binary representation fails to capture relationships between functionally related genomic regions or cell types. The limitations of one-hot encoding motivated the development of more sophisticated representation techniques for both natural language and genomic data.

Distributed representations

To address the limitations of one-hot encoding, researchers developed distributed representations, also known as word embeddings. In this approach, words are represented as dense vectors in a continuous vector space, where semantically similar words are mapped to nearby points. This allows the model to capture relationships between words based on their contexts. For example, in a well-trained embedding space, the vectors for “king” and “queen” would be close to each other, and the

relationship between “king” and “queen” could be represented as a vector offset, such as “king” - “man” + “woman” = “queen” (Figure 2.1B).

The question then becomes: how do we learn these embeddings? One influential method for learning word embeddings is Word2Vec, which we will discuss next.

Word2vec

Perhaps one of the most influential methods for learning word embeddings is Word2Vec, introduced by Mikolov et al. in 2013[65]. Word2Vec is a shallow, two-layer neural network that is trained to predict the context words surrounding a target word in a sentence. There are two main architectures for Word2Vec: Continuous Bag of Words (CBOW) and Skip-Gram (SG). In the CBOW architecture, the model predicts a target word based on its surrounding context words (Figure 2.1C). Conversely, in the Skip-Gram architecture, the model predicts the context words given a target word (Figure 2.1D). Both architectures learn to represent words as dense vectors in a continuous vector space, where two semantically similar words will have similar vector representations (i.e. be close in the vector space).

Briefly, the training process involves sliding a window over a text corpus and using the words within that window to update the word vectors. The objective is to maximize the probability of predicting the context words given the target word (or vice versa, depending on the architecture). This is typically done using techniques like negative sampling or hierarchical softmax to efficiently approximate the softmax function over a large vocabulary[66,67].

The actual embeddings are obtained from the *weights* of the hidden layer after training. Importantly, words in the training process are fed to the model as one-hot encoded vectors, but this has the consequence of acting as a lookup table to retrieve the corresponding dense embedding vector from the hidden layer weights. For example, if we denote the one-hot encoded vector for a word as one-hot and the weight matrix of the hidden layer as W_h, the embedding for the word cat can be computed as:

$$\mathrm{e}_{\mathrm{cat}} = W_h^T \cdot \text{one-hot} \tag{1}$$

or,

$$\begin{pmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,\mathrm{V}} \\ w_{2,1} & w_{2,2} & \dots & w_{2,\mathrm{V}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{\mathrm{d},1} & w_{\mathrm{d},2} & \dots & w_{\mathrm{d},\mathrm{V}} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} w_{1,1} \cdot 0 + w_{1,2} \cdot 1 + \dots + w_{1,\mathrm{V}} \cdot 0 \\ w_{2,1} \cdot 0 + w_{2,2} \cdot 1 + \dots + w_{2,\mathrm{V}} \cdot 0 \\ \vdots \\ w_{\mathrm{d},1} \cdot 0 + w_{\mathrm{d},2} \cdot 1 + \dots + w_{\mathrm{d},\mathrm{V}} \cdot 0 \end{pmatrix} = \begin{pmatrix} w_{1,2} \\ w_{2,2} \\ \vdots \\ w_{\mathrm{d},2} \end{pmatrix} \tag{2}$$

Word2Vec embeddings have been highly successful in capturing semantic relationships between words and have been widely adopted in various NLP tasks. Extensions and improvements upon

Word2Vec, such as GloVe[68], FastText[69], and Doc2Vec[70] have further advanced the field of word embeddings. These methods have laid the groundwork for more complex models, including those based on transformers, which we will explore in subsequent sections. However, **the critical insight from Word2Vec — that words can be meaningfully represented in a continuous vector space based on their contexts — remains foundational to modern NLP and our own work on genomic interval analysis**. Each word gets its own vector in a high-dimensional space, and the relationships between these vectors capture semantic similarities and differences. This key concept has inspired our analogous approaches in genomics, where genomic regions or chromatin accessibility profiles can be embedded in a similar fashion to capture their functional relationships.

Indeed, many methods have been developed that adapt Word2Vec-style embeddings to biological sequence data, such as DNA, RNA, and proteins[71–73]. These approaches treat genes, proteins, or k-mers as "words" and biological sequences as "sentences," enabling the capture of functional and structural relationships in a continuous vector space. These methods were made possible by robust and agreed-upon vocabularies of biological sequences, such as the set of all known genes or proteins in a given organism.

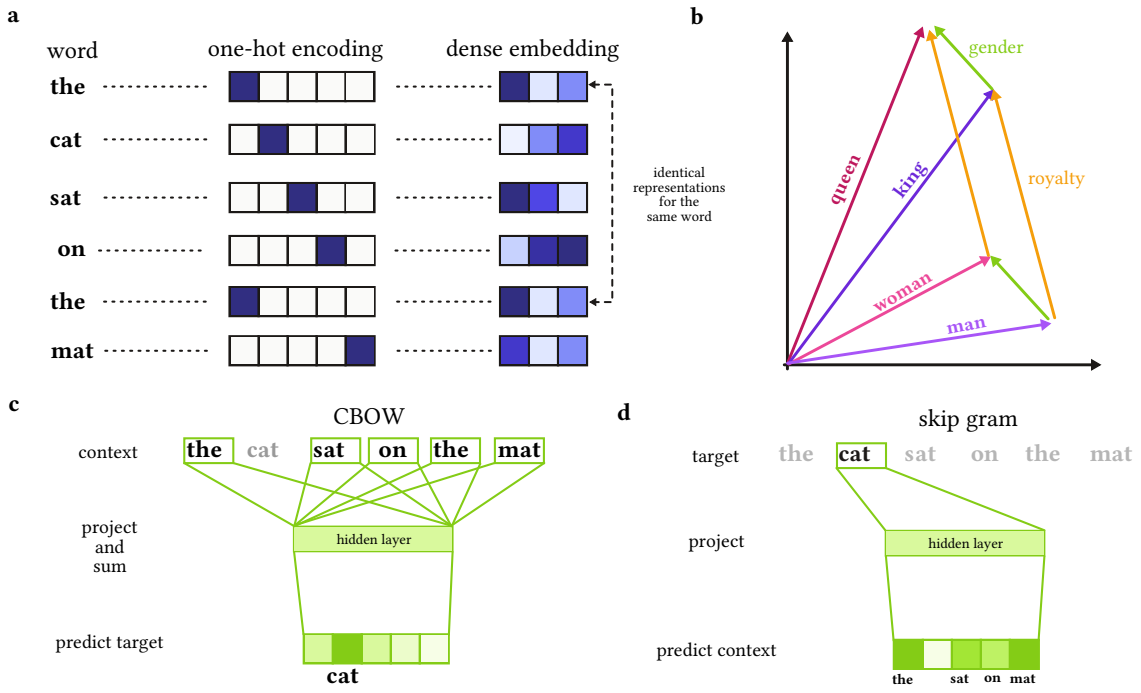


Figure 2.1: Different methods for representing words numerically: one-hot encoding vs. distributed representations.

a. Schematic of one-hot encoding versus distributed representations for individual words. One-hot encoding results in high-dimensional, sparse vectors, and orthogonal representations for different words. Distributed representations (embeddings) result in dense vectors where semantically similar words are close in the vector space. **b.** Example of semantic relationships captured by word embeddings, where vector arithmetic can reveal relationships such as "king" - "man" + "woman" ≈ "queen". **c.** Continuous Bag of Words (CBOW) architecture

of Word2Vec, where the model predicts a target word based on its surrounding context words. **d.** Skip-Gram (SG) architecture of Word2Vec, where the model predicts context words given a target word.

Recurrent Neural Networks and their gated variants

A river bank or a financial bank? Contextual embeddings and their limitations

One of the critical assumptions made with Word2Vec and other word embedding models is that word embeddings are context-*independent*; that is, each word has a single fixed representation regardless of its surrounding context. In other words, the embedding for a word like "bank" would be the same whether it appears in the context of a financial institution or the side of a river. The word is simply "looked up" in the embedding table and used as-is (Figure 2.2A). While this simplification has been effective for many NLP tasks, it limits the model's ability to capture the nuanced meanings of words that can change based on context. Instead the embedding for "bank" will be some average of the two meanings. This limitation motivated the development of contextual embeddings, which emerged first through recurrent neural networks and later evolved into the attention-based architectures that dominate today.

Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a class of neural networks designed for processing sequential data, making them particularly well-suited for tasks in natural language processing (NLP). Unlike traditional feedforward neural networks, RNNs have connections that loop back on themselves, allowing them to maintain a hidden state that captures and stores information about previous inputs in the sequence. This makes RNNs capable of modeling temporal dependencies and context, which are crucial for understanding language.

In a standard RNN architecture, words are still represented using embeddings, such as those learned by Word2Vec. However, instead of treating each word independently, the RNN processes the sequence of word embeddings one at a time, updating its hidden state at each step. The hidden state serves as a memory that captures information about the words that have been processed so far. This allows the RNN to generate context-aware representations of words based on their surrounding context in the sequence (Figure 2.2B).

Because of this, now if the word "bank" appears in a sentence, the RNN can use its hidden state to determine whether the context suggests a financial institution or the side of a river, and adjust its representation accordingly. This ability to capture context makes RNNs more powerful than static word embeddings for many NLP tasks. Still, simple RNNs often struggled with longer sequences due to unstable training dynamics, motivating a series of refinements which we describe next.

Long Short-Term Memory (LSTM) networks

While RNNs are capable of modeling sequential data, they suffer from a significant limitation known as the **vanishing gradient problem**. This issue arises when training RNNs on long sequences, where the gradients used for updating the model's weights can become very small, effectively preventing the model from learning long-range dependencies in the data. Intuitively, this means that the RNN may struggle to "remember" information from earlier in the sequence when making predictions about later words because the influence of those earlier words diminishes significantly over time. If we are discussing a financial bank, but the word "bank" appears several words later, the model may have difficulty connecting the two concepts.

To address this limitation, the NLP community turned to an architecture that had been introduced much earlier: the Long Short-Term Memory (LSTM) network (Hochreiter & Schmidhuber, 1997). While the original idea dates to the late 1990s, LSTMs gained prominence in the 2010s as sufficient computational power became available to train them effectively and the vanishing gradient problem proved a major bottleneck for simpler RNNs.

LSTMs are a specialized type of RNN that incorporate a more complex architecture designed to better capture long-range dependencies. The key innovation is the introduction of a memory cell and gating mechanisms (input, output, and forget gates) that regulate the flow of information. These gates allow the network to selectively remember or forget information over long periods, largely overcoming the vanishing gradient problem that plagued simple RNNs. Intuitively, this means an LSTM can hold on to the "financial" context long enough to correctly interpret the word "bank" when it appears much later in a sequence.

The ability to effectively model long-range dependencies unlocked the potential of recurrent models, leading to dramatic improvements in applications like machine translation and speech recognition. The success of LSTMs spurred further innovation. Simplified variants, such as the Gated Recurrent Unit (GRU), were proposed to retain much of the performance of LSTMs while reducing computational cost[74]. Building on these architectures, the sequence-to-sequence (seq2seq) framework demonstrated how paired encoder–decoder RNNs could perform end-to-end translation[75], and attention mechanisms were layered on top to further improve performance[76]. These developments marked the high-water mark of RNN-based NLP systems before the shift toward fully attention-based models.

The success of RNNs and LSTMs in natural language processing naturally extended beyond text to other sequential data types, particularly in computational biology. Researchers recognized that biological sequences — those like DNA, RNA, and protein sequences — share fundamental similarities

with natural language: they are composed of discrete symbols (nucleotides or amino acids) arranged in meaningful sequences where order and context matter significantly. This analogy proved fruitful, leading to the application of LSTM architectures for tasks such as gene expression prediction, protein function annotation, and DNA sequence classification[77,78].

Even still, LSTMs and GRUs have their own limitations. They can be computationally intensive to train, especially on very long sequences, and they may still struggle with extremely long-range dependencies. Additionally, while RNN-based architectures improved upon static embeddings, they were ultimately surpassed by transformer models, which dispense with recurrence entirely in favor of self-attention. We turn to those next.

Transformers and attention mechanisms

Transformers

Transformers are a type of neural network architecture that has gained significant popularity in recent years, particularly for natural language processing tasks. Introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017[50], transformers leverage a mechanism called self-attention to process input sequences *in parallel*, rather than *sequentially* as RNNs and LSTMs do. This enables two things: first, transformers can capture long-range dependencies in the data more effectively, as there is no more "distance" between words in a sequence; and second, they can be trained more efficiently on large datasets due to their parallelizable architecture. Specifically, they consume sequences as a whole instead of word-by-word, allowing for much faster training times (Figure 2.2C).

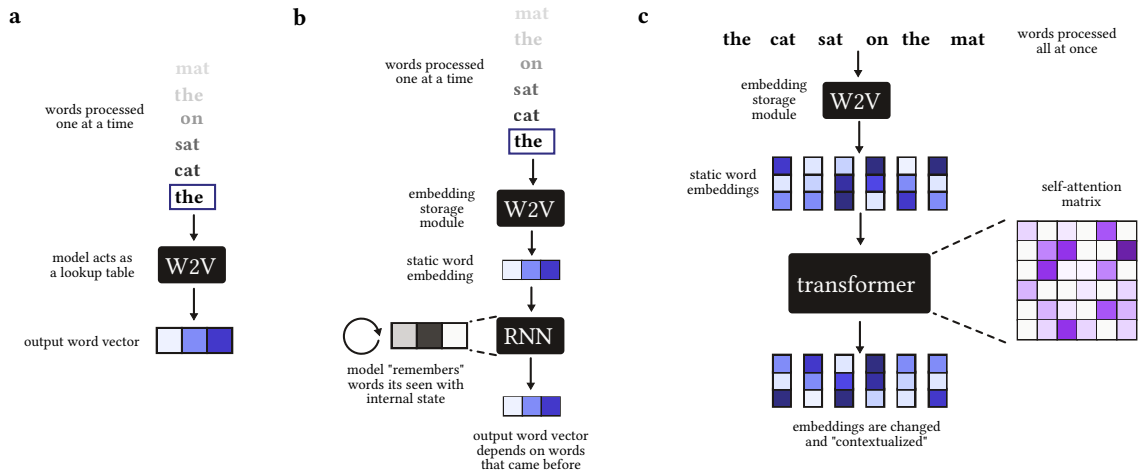


Figure 2.2: Evolution of word representation and sequence modeling techniques in NLP: from Word2Vec to RNNs to Transformers.

a. Schematic of Word2Vec. It learns word embeddings based on local context windows. The model acts as a lookup table for word vectors. **b.** Schematic of RNNs. They process sequences sequentially, maintaining a hidden state that captures information from previous time steps. **c.** Schematic of Transformers. They use

self-attention to process entire sequences in parallel, allowing for better capture of long-range dependencies. Transformers do away with recurrences entirely (i.e. processing one word at a time), instead relying on self-attention to relate different positions of the sequence.

Self attention

The core innovation of transformers is the self-attention mechanism, which allows the model to weigh the importance of different words in a sequence when updating their representations. In self-attention, each word in the input sequence is transformed into three vectors: a query vector, a key vector, and a value vector. The attention score between two words is computed as the dot product of the query vector of one word and the key vector of another word, followed by a softmax operation to obtain a probability distribution. This score determines how much attention one word should pay to another when updating its representation. The final output for each word is then computed as a weighted sum of the value vectors of all words in the sequence, where the weights are given by the attention scores.

Mathematically, the self-attention mechanism is expressed as follows:

$$\text{Attention}(Q, K, V) = \frac{\sigma(\text{QK}^T)}{\sqrt{d_k}} \cdot V \tag{3}$$

where Q, K, and V are the matrices of query, key, and value vectors for all words in the sequence, and d_k is the dimensionality of the key vectors. The division by $\sqrt{d_k}$ is a scaling factor that helps stabilize the gradients during training.

A key thing to note is that in self-attention, each word can attend to all other words in the sequence, allowing the model to capture complex dependencies and relationships between words regardless of their positions. This is in contrast to RNNs, where the influence of earlier words can diminish over time due to the sequential processing.

While powerful, the self-attention mechanism is computationally intensive, especially for long sequences, as it requires computing attention scores for all pairs of words. This leads to $O(n^2)$ complexity, where n is the length of the input sequence.

This complexity resulted in difficulties when scaling up transformers to very long sequences, such as entire documents or whole genomic region sets. To address this, various modifications and optimizations have been proposed, such as sparse attention mechanisms, which limit the number of words each word can attend to, and hierarchical transformers, which process sequences at multiple levels of granularity.

Approximations of self-attention

After the original transformer architecture was introduced, several variants and improvements have been proposed to enhance its performance and efficiency, particularly within the self-attention mechanism. We describe some notable examples in turn.

Low-rank & kernel-based approximations

A first class of methods reduces the quadratic cost of self-attention by approximating the attention matrix with low-rank or kernel-based decompositions. One example is Linformer, which projects the sequence dimension into a lower-rank space, yielding linear complexity while retaining competitive accuracy on many NLP tasks[79]. Similarly, Nyströmformer applies the Nyström method to approximate the softmax kernel, further reducing memory usage while scaling to longer contexts[80]. Finally, the Performer model takes a different approach, introducing FAVOR+ random feature maps to approximate the softmax kernel directly in linear time[81]. These methods sacrifice exactness but offer strong trade-offs between efficiency and fidelity.

Sparse attention mechanisms

Another approach is to approximate self-attention using sparse matrices, exploiting the intuition that not all pairwise interactions are necessary. Sparse Transformer introduced block-sparse patterns to limit computations without major accuracy loss[82]. One example is Longformer, which extended this idea with a sliding-window local attention pattern augmented by global tokens for tasks like question answering[83]. Another, BigBird, combined local, global, and random sparse connections, providing both empirical scalability and theoretical guarantees of universality[84]. These structured sparsity patterns allow transformers to process sequences with tens of thousands of tokens while retaining exact attention within the restricted subsets.

Algorithmic improvements

A final line of work focuses on re-engineering the exact attention operation itself to be more memory and bandwidth-efficient. One of the most well-known is FlashAttention. FlashAttention reformulates attention as a sequence of IO-aware matrix multiplications, using tiling and recomputation to eliminate the need to materialize the full attention matrix in GPU memory[85]. FlashAttention v2 further generalized these kernels for both training and inference, showing substantial speedups on modern accelerators[86]. Related libraries, such as xFormers, provide a general framework for fused attention kernels with minimal memory overhead. Unlike low-rank or sparse approaches, these methods compute attention exactly but achieve significant gains by exploiting hardware efficiency. FlashAttention has been widely adopted in large language model training pipelines due to its simplicity and effectiveness. Indeed, it is so efficient that it is now often the default attention implementation in many deep learning frameworks including PyTorch.

From transformers to large language models and beyond

Large language models and foundation models

The transformer architecture laid the groundwork for the development of large language models (LLMs), which are typically defined as transformer-based models with hundreds of millions to billions of parameters trained on massive text corpora. These models, such as OpenAI's GPT series[45] and Google's PaLM[87], have demonstrated remarkable capabilities in generating coherent text, answering questions, and performing various NLP tasks with minimal fine-tuning. The key to their success lies in their scale and the diversity of data they are trained on, which allows them to learn a wide range of linguistic patterns and knowledge.

Building on the success of LLMs, the concept of **foundation models** has emerged. Foundation models are large-scale models trained on broad data at scale and can be adapted to a wide range of downstream tasks. They serve as a base upon which more specialized models can be built through fine-tuning or prompt engineering. The term "foundation model" emphasizes the idea that these models provide a foundational understanding of language that can be leveraged for various applications across different domains[88].

The development of LLMs and foundation models has also spurred interest in multimodal models that can process and generate content across different modalities, such as text, images, and audio. Examples include OpenAI's CLIP[44] and DALL-E[89], which combine text and image understanding to perform tasks like image generation from textual descriptions.

The advancements in LLMs and foundation models have significant implications for various fields, including bioinformatics and genomics. By leveraging the principles of transfer learning and the ability to learn from large-scale data, researchers can develop models that understand biological sequences and structures, enabling new insights and applications in areas such as gene regulation, protein folding, and drug discovery. In the following chapters, we will explore how these concepts have influenced our approach to analyzing single-cell ATAC-seq data using transformer-based architectures.

Summary of NLP techniques

Below is a summary of models and methods from the NLP literature that have been discussed in this chapter:

Table 2.1: Summary of NLP techniques and models

Method	Year	Description
One-hot encoding	Early	Represents words as sparse binary vectors with single 1 at vocabulary index.
Word2Vec	2013	Shallow neural network learning dense word embeddings via context prediction.
GloVe	2014	Global vector representations combining matrix factorization with local context.
FastText	2017	Extension of Word2Vec incorporating subword information for rare words.
RNN	1980s	Recurrent networks processing sequences with hidden state memory.
LSTM	1997	Long Short-Term Memory networks addressing vanishing gradient problem.
GRU	2014	Gated Recurrent Units as simplified alternative to LSTMs.
Seq2seq	2014	Encoder-decoder framework for sequence-to-sequence tasks.
Transformer	2017	Self-attention based architecture processing sequences in parallel.
Linformer	2020	Low-rank approximation of attention for linear complexity.
Longformer	2020	Sparse attention with sliding window and global tokens.
FlashAttention	2022	IO-aware exact attention computation for memory efficiency.
GPT	2018-2023	Generative pre-trained transformer models for text generation.
CLIP	2021	Contrastive language-image pre-training for multimodal understanding.
PaLM	2022	Pathways Language Model demonstrating emergent capabilities at scale.

From words to genomic regions: adapting NLP techniques to gene regulation

Building machine learning models for genomic interval data using genomic tokens

Adapting NLP techniques to epigenomic data requires careful consideration of how to represent genomic intervals as discrete tokens. Unlike natural language, or even some biological datasets where words and tokens are well-defined units with semantic meaning, genomic *regions* are continuous sequences of nucleotides that do not have strong inherent boundaries or meanings. Therefore, to leverage NLP models for genomic data, we need to define a suitable vocabulary and tokenization strategy.

There is a useful analogy between words in language and regions in the genome: both are building blocks that models compose to express higher-order structure. However, the analogy breaks down in important ways. First, linguistic tokens arise from shared, largely agreed-upon ontologies (morphology, lexicons, grammars) and well-understood segmentation rules. Genomic intervals, by contrast, are fluid: regulatory elements, chromatin states, and functional domains overlap, vary in size, and lack a single universally accepted partitioning. Second, linguistic tokens have an inherent order to them. Genomic regions, while ordered along chromosomes, are rarely analyzed in a temporal or strictly sequential context. Instead, their function often depends on 3D conformation, epigenetic state, and cell type, complicating the notion of "context" that NLP models typically rely on.

Deep learning models are increasingly being developed for genomic interval data, but most treat intervals as continuous signals or coordinate-derived features rather than as discrete, reusable tokens. Treating genomic regions as tokens offers substantial advantages: compact, shared vocabularies that improve parameter efficiency and transfer learning; direct alignment between model components and biological entities for clearer mechanistic interpretations; and token-level attribution and attention analyses that make model decisions more transparent.

This thesis work formalizes our proposed framework of genomic interval tokens and adapts modern NLP architectures to epigenetic data, providing a framework and model suite that deliver strong predictive performance while substantially improving interpretability enabling biological insights and reducing computational complexity over prior approaches.

Chapter 3: Efficient computational tools for creating genomic interval vocabularies and tokenization frameworks for modern machine learning applications

Nathan J. LeRoy[1, 2], Julia Rymuza[1], Donald Campbell[1], Oleksandr Khoroshevskyi[1], Seth Stadick[3], Sang-Hoon Park[1], Edward Chen[4], Nathan C. Sheffield[1, 2]

[1]Department of Genome Sciences, School of Medicine, University of Virginia, 22908, Charlottesville VA
[2]Department of Biomedical Engineering, School of Medicine, University of Virginia, 22908, Charlottesville VA
[3]Life Sciences Group, Bio-Rad Laboratories, 1000 Alfred Nobel Dr, Hercules, 94547, California, USA
[4]Department of Computer Science, School of Engineering and Applied Sciences, University of Virginia, 22908, Charlottesville VA

Note: This chapter is adapted from the following publication and preprint:
(1) Rymuza *et al.*[90], which introduced a comprehensive framework for constructing and evaluating consensus genomic interval sets (or "universes") for machine learning applications; and
(2) LeRoy *et al.*[91], which presented the genomic tokenizers, a high-performance rust library for efficient tokenization of genomic interval data into these consensus vocabularies.

Introduction

Advancements in high-throughput sequencing technologies have generated vast and diverse epigenomic datasets from assays such as ChIP-seq, ATAC-seq, and Hi-C[92]. These experiments are frequently summarized as genomic intervals, which define regions on a genome. Summarized genomic interval data has grown rapidly over the past few years[93]. This proliferation of data provides a critical opportunity to uncover generalizable patterns, support predictive modeling, and enable transfer learning using large-scale machine learning (ML) methods. However, major barriers in applying modern ML methods to genomic interval data arise due to two fundamental gaps in available tooling: 1) the lack of systematic methods for creating consensus vocabularies of genomic intervals (vocabulary builders), and 2) the absence of efficient, flexible tools for mapping new datasets to these vocabularies (tokenizers).

First, genomic interval data is inherently variable and unstructured; each dataset defines its own regions of interest, making it difficult to compare or combine results across experiments. This is incompatible with ML methods, which generally require data to be described in a discrete, consistent vocabulary. For example, in natural language processing (NLP), models require well-defined vocabularies to process and integrate diverse sets of textual data. The process of mapping new, unseen datasets to a shared feature set is called tokenization and is a vital part of NLP research and development[59–61,94]. Without such a standardized basis, it is difficult or impossible to create feature-aligned representations suitable for ML. Similarly, for genomic intervals, it is necessary to

map new datasets to a shared vocabulary, or consensus set of genomic intervals[90,93]. This process is conceptually similar to tokenization in NLP and serves the same purpose: to enable consistent and scalable representation of variable input data. However, creating these consensus vocabularies remains largely *ad hoc* and manual. While methods exist for generating consensus peak sets or reference interval collections, they are typically designed for specific analyses rather than systematic vocabulary creation for ML applications. Most approaches rely on simple binning, intersection or merging strategies[95–97] without considering the downstream requirements of machine learning models, such as vocabulary size constraints, feature importance, or cross-dataset generalizability. This gap leaves researchers to develop custom, often suboptimal solutions for each project.

Second, even when consensus vocabularies exist, the tools for efficiently mapping genomic intervals to these shared feature sets are inadequate for modern ML workflows. Specifically, while tools exist for genomic interval comparison[98–102], they are limited in several ways. They are typically only accessible in a single environment, such as R, or as command-line tools, and are not optimized for fast, in-memory processing. This limitation poses a significant pain point for machine learning pipelines in Python, which require high-throughput, efficient data handling. Additionally, they generally lack flexible APIs that integrate seamlessly in the Python-based machine learning ecosystem, particularly with libraries like PyTorch, TensorFlow, or huggingface/transformers. As a result, ML applications in genomics often suffer from ad hoc preprocessing steps, pipeline bottlenecks, and limited scalability.

To solve these problems, we created `gtars`, a comprehensive library designed specifically for genomic machine learning that addresses both gaps, and `geniml`, a toolkit for generating consensus peak sets for genomic machine learning models. For vocabulary creation, we provide `gtars-uniwig` (referred to as `uniwig`), a pre-processing tool for systematic consensus set generation that serves as the first necessary and compute-intensive preprocessing step for our downstream vocabulary builders. For tokenization, we developed `gtars-tokenizers`, which provides four main improvements: First, its Rust core makes it faster than many existing tools, and in many cases, as fast as the fastest available implementations. Second, it is designed for convenience for ML, exposing a direct bridge into modern ML infrastructure such as HuggingFace and PyTorch, so genomic intervals can be tokenized and passed into models without ad hoc preprocessing. Third, unlike prior utilities, it treats genomic intervals in a way that mirrors the conceptualization of words in NLP, enabling consistent, vocabulary-based representations that scale across datasets. Finally, it offers a unified engine with bindings for Python, R, Rust, command line, and web applications so the same foundation can serve diverse users and workflows.

These tools are essential components for establishing robust and modern ML workflows for genomic interval data. The standardization and efficiency they provide are critical for scaling genomic machine learning beyond individual experiments to large-scale, multi-dataset analyses. By addressing both vocabulary creation and tokenization systematically, `gtars` enables researchers to focus on model development and biological insights rather than wrestling with data preprocessing challenges. In the following sections, we describe these tools in turn, demonstrating how they work together to bridge the gap between genomic data and machine learning infrastructure.

Results

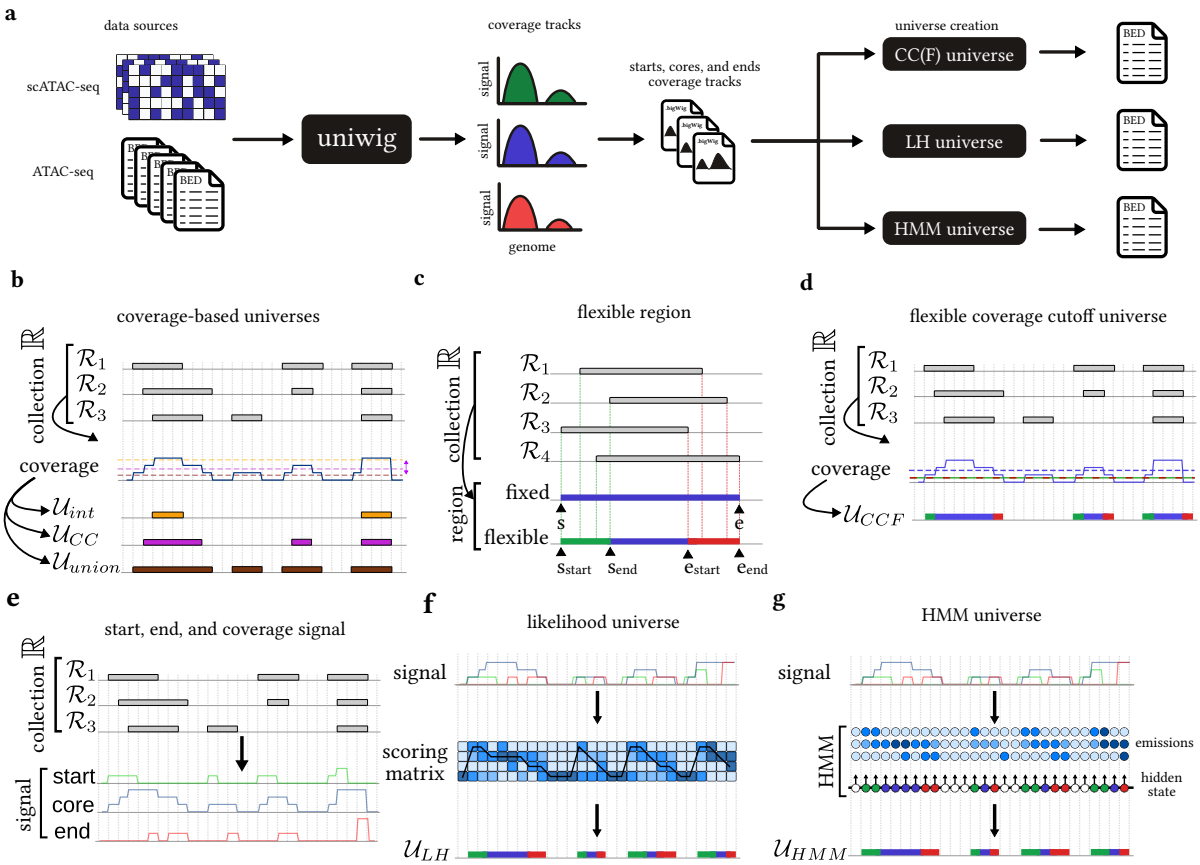


Figure 3.1: Overview of uniwig and the universe creation tools.

a. Schemtic overview of the uniwig process. The tool takes as input a set of BED files and will output coverage tracks for the cores starts and ends. **b.** Coverage-based universes are derived from the genome coverage of a collection of region sets. Examples include intersection U_i, coverage cutoff U_{CC}, and union universe U_{union}. **c.** A flexible region in contrast to fixed region can represent boundaries of many variable regions. **d.** The flexible coverage cutoff (CCF) universe is based on coverage of the genome by a collection. It uses two cutoff values: the lower defines flexible boundaries and the upper defines the region core. **e.** A collection of genomic region sets is aggregated, and region starts, core (overlap), and ends are counted, creating signal tracks. **.f** Maximum likelihood universe is derived from three signal tracks. Using a likelihood model, we build a scoring matrix that assesses the probability of each position being a given part of a flexible region. Next, we find the most likely path, which represents the maximum likelihood universe. **g.** The HMM universe treats signal tracks representing genome coverage by different parts of a region as emissions of hidden states that correspond to different parts of flexible regions.

Creating a principled vocabulary for genomic intervals

The creation of a principled vocabulary for genomic intervals requires a systematic two-step process. First, input genomic interval datasets (typically BED files) are processed using `uniwig` to generate standardized coverage tracks representing interval starts, cores, and ends across the genome. Second, these coverage tracks are then processed through one of four universe construction methods—coverage cutoff (CC), flexible coverage cutoff (CCF), maximum likelihood (LH), or Hidden Markov Model (HMM)—to generate the final consensus genomic interval set that serves as the vocabulary for machine learning applications (Figure 3.1A, See methods).

Overview of uniwig: a pre-processing tool for consensus genomic interval set construction

The first step when building a machine learning model for genomic intervals is to define a universe of intervals that will serve as the vocabulary for the model. This universe should comprehensively cover the genomic regions relevant to the biological context of interest, such as regulatory elements in a specific cell type or tissue. The universe is typically represented as a BED file containing genomic coordinates. Once the universe is defined, new datasets can be tokenized by mapping their intervals to the nearest or overlapping regions in the universe, converting them into a format suitable for machine learning models.

To facilitate this process, we developed `uniwig`, a high-performance tool in rust that computes the coverage of genomic intervals across multiple datasets and generates a unified representation. `uniwig` takes as input a set of BED files representing genomic intervals from different experiments and computes three single coverage `.bigWig` files, one for the starts, the cores, and the ends of the intervals. This representation captures the distribution of intervals across the genome and can be used to identify regions that are consistently covered across datasets. The output `.bigWig` files can then be used as input for downstream analysis or as part of the vocabulary creation process (Figure 3.1A, See methods).

Simple coverage-based universe construction

Once the coverage files are generated, they can be used to create a consensus set of genomic intervals that will serve as the vocabulary for machine learning models. The simplest method for constructing a consensus genomic interval set is to use coverage information directly. This approach involves computing the coverage of genomic intervals across all input datasets and applying a threshold to define which regions should be included in the final universe. One can apply either a fixed coverage cutoff (CC), an intersection, or a union of all input datasets to define the universe (Figure 3.1B). However, one limitation of this approach is that it treats each interval as a binary event (present or

absent), which may not capture the full complexity of the data, especially when intervals vary in length or when there are differences in the number of intervals across datasets. For this, we consider methods that consider flexible regions – that is, regions with a start, core, and end – to better capture the underlying biological signals (Figure 3.1C).

Novel methods for constructing consensus genomic interval sets

To improve upon simple coverage-based approaches, several advanced methods have been developed within our group, with the definitive framework and evaluation published by Rymuza *et al.* (2024). In summary, we provide three methods for generating these consensus sets, each with its own advantages and trade-offs: 1) coverage cutoff (CC) universe, 2) maximum likelihood (LH) universe, and 3) the HMM universe. These methods were developed by our group and others[90], and each is suited to different scenarios depending on the characteristics of the input data and the desired properties of the resulting universe. We briefly summarize each method below (See methods for more details).

Coverage Cutoff Flexible (CCF) Universe

This method treats a collection of flexible genomic region sets as a single coverage signal track across the genome. It then applies a coverage threshold, or cutoff, to this track. Any genomic position where the coverage is greater than or equal to the chosen cutoff is included in the final universe. This approach is straightforward and allows for easy control over the size of the universe by adjusting the cutoff value. However, it may not capture all relevant regions, especially if the input datasets are highly variable or if some regions are only present in a few datasets (Figure 3.1D, Figure 3.1E). This approach is a hybrid between simply taking the union of all regions (a cutoff of 1) and taking the intersection (a cutoff equal to the total number of region sets). The method provides a tunable parameter that can be adjusted based on the needs of the analysis.

Maximum Likelihood (LH) Universe

The Maximum Likelihood (LH) universe was designed to better preserve the boundaries of individual regions and avoid merging adjacent regions, which can be an issue with simple coverage-based methods. Instead of just using coverage, this method calculates three separate signal tracks at base-pair resolution: one for region starts, one for region ends, and one for coverage (core). It then uses a likelihood model to calculate the probability of each genomic position being a start, core, or end (Figure 3.1F). The final universe is determined by finding the most likely path of these states (start, core, end) across the genome.

Hidden Markov Model (HMM) Universe

Lastly, the Hidden Markov Model (HMM) approach is a more tunable and sophisticated version of the likelihood model. It models the genome as a sequence of hidden states (e.g., region start, core, end, or background). The three signal tracks (starts, ends, coverage) are treated as emissions, or

observations, generated by these hidden states. The key advantage is that a user can adjust the model's transition probabilities (the likelihood of moving from one state to another) and emission probabilities to fine-tune the resulting universe, for example, to prevent unnecessary fragmentation of regions (Figure 3.1G).

This framework established by Rymuza *et al.*[90] provides a systematic approach to constructing consensus genomic interval sets that can be tailored to the specific needs of different machine learning applications. By leveraging the coverage information from multiple datasets, these methods enable the creation of robust and biologically meaningful vocabularies that facilitate the application of machine learning techniques to genomic interval data.

Building on this comprehensive evaluation framework, Rymuza *et al.*[90] conducted systematic benchmarking across multiple genomic datasets to assess the performance of these universe construction methods in downstream machine learning applications. Their analysis revealed that the HMM universe consistently outperformed simpler coverage-based approaches and the maximum likelihood method across various evaluation metrics, including model accuracy, feature interpretability, and biological relevance of the resulting vocabularies (Supplementary Figure A.1).

Having established methods for creating consensus vocabularies, the next challenge is efficiently mapping new genomic datasets to these standardized feature spaces. This process — analogous to tokenization in natural language processing — requires specialized tools optimized for the scale and performance demands of modern machine learning workflows.

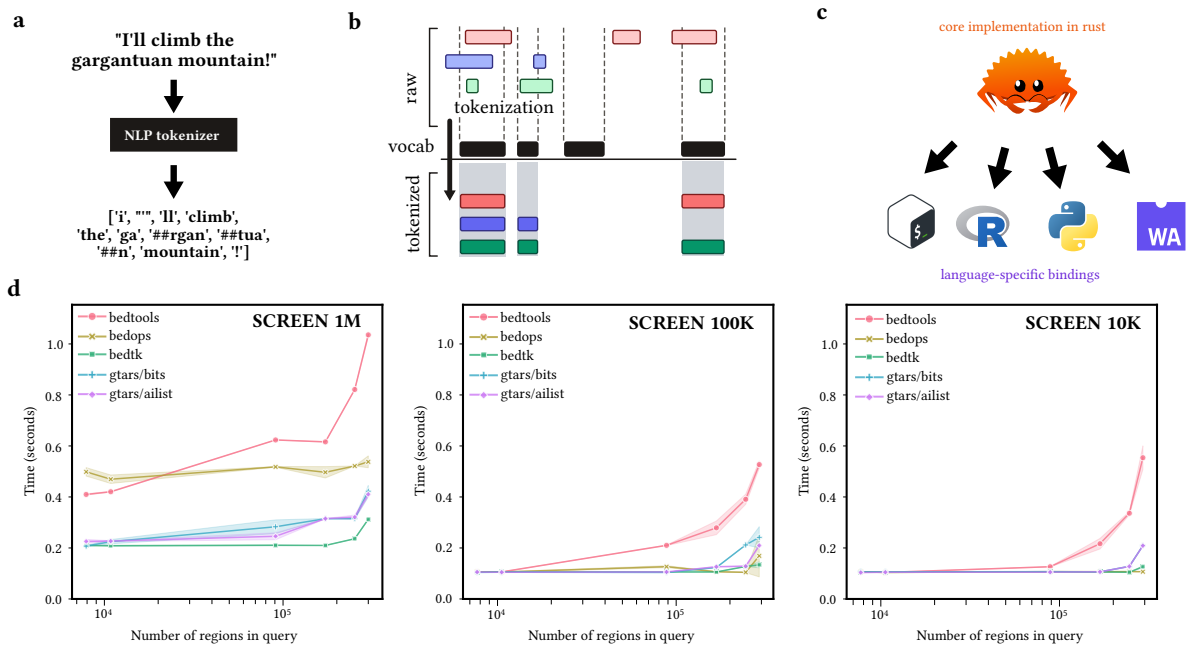


Figure 3.2: Overview and benchmarking of gtokenizers, a Rust-based library for genomic interval tokenization.

a, Schematic of natural language tokenization. NLP tokenizers typically break sentences up into words or word-pieces. **b**, Schematic illustrating gtokenizers applied to regulatory elements (e.g., cCREs) for standardized interval representation. **c**, Architecture of gtokenizers, with a core implementation in Rust and support for multiple language bindings (e.g., CLI, R, Python, WebAssembly). **d**, Runtime benchmarking across three query sizes (1M, 100K, 10K regions) against existing tools (bedtools, bedops, bedtk) and Rust-based implementations (`gtars/bits`, `gtars/alist`), demonstrating scalability and performance.

Overview of genomic interval tokenizers

Modern deep-learning workflows in natural language processing require tokenizers to convert new text into the model's fixed vocabulary, enabling consistent inputs for downstream processing. Tokens in language models correspond to discrete words or subword units (Figure 3.2A). In the genomic domain, a comparable process is necessary: machine learning models that treat genomic intervals as discrete units, like words in a sentence, must map each dataset to a common set of regions, or a vocabulary for genomic intervals[90,103–106] This vocabulary ensures data across experiments are represented in a standardized, comparable way (Figure 3.2B). Different datasets can thus be interpreted with the same model architecture and feature space, just as diverse text inputs are aligned via tokenization in NLP.

We implemented two overlap methods in `gtars-tokenizers`: `gtars/bits`, which uses binary interval tree search (BITS)[107], and `gtars/alist`, which uses an Augmented Interval List (AIList)[101] (see methods). Both methods are implemented in Rust for performance and memory efficiency. To maximize flexibility and usability, we provide bindings for `gtars/tokenizers` in Python, R, and WebAssembly, as well as a command-line interface (CLI) (Figure 3.2C). This allows users to integrate genomic interval tokenization into their existing workflows, whether they are using Python-based machine learning libraries like TensorFlow or PyTorch, R-based bioinformatics tools, or require a web-based solution for use in a browser (see methods).

Gtars tokenizers are highly performant

To highlight the performance of `gtars/tokenizers`, we benchmarked it against existing tools for genomic interval tokenization (see methods). We compare `gtars/tokenizers` to `bedtools`, `bedops`, and `bedtk`[98–100]. These tools focus on general-purpose genomic interval arithmetic and are not optimized for machine learning applications. We found `gtars/tokenizers` to be consistently as fast or faster than existing tools (Figure 3.2D). For large universes with >1 million intervals (like those used in genomic interval machine learning), `gtars-tokenizers` is around 2- 3x faster than `bedtools` and `bedops`, while being comparable to `bedtk`. This pattern holds across different query sizes (1M, 100K, and 10K regions), demonstrating the scalability and performance of `gtars/tokenizers`.

Gtars tokenizers work seamlessly with modern machine learning infrastructure

The gtars-tokenizer implementation is compatible with the Hugging Face tokenizers API, enabling seamless integration with the broader Hugging Face ecosystem. The `gtars` tokenizers are near-drop-in replacements for existing Hugging Face tokenizers, meaning users can pass them to the HuggingFace transformers package functions and classes using the same ergonomics as a standard NLP workflow. The consistent interface makes it easy for ML engineers to adapt to training models on genomic interval data. It also means that the downstream outputs of the training process will seamlessly integrate with popular downstream frameworks and tools that rely on the Hugging Face tokenizers standard, such as PyTorch Lightning, AllenNLP, and evaluation libraries like Evaluate, PEFT, and Weights & Biases. To highlight this, we provide a brief example of how someone can use our tokenizers to preprocess data for a simple neural network built with PyTorch. The snippet first creates a new tokenizer from a BED-file, and then uses it to preprocess data for a neural network.

```
import torch
import gtars.tokenizers as Tokenizer

tokenizer = Tokenizer.from_bed("path/to/bed/file.bed")
network = torch.nn.Embedding(tokenizer.vocab_size, 64)

query_intervals = [("chr1", 1000, 2000), ("chr2", 3000, 4000)]
tokens = tokenizer.tokenize(query_intervals)["input_ids"]
out = network(torch.tensor(tokens))
```

Gtars tokenizers are available in a wide array of computing environments.

To maximize usability, we expose the Rust core of gtars-tokenizers as a Rust library crate, as a command-line tool, with R bindings, Python bindings, and for WebAssembly (WASM). This broad set of interfaces ensures that the same high-performance engine can serve diverse communities – from machine learning researchers to bioinformaticians and end-users in web tools – without duplicating functionality or compromising performance. It also reduces maintenance requirements for the community because a single fast interface can be deployed in many situations.

Discussion

The `gtars` and `geniml` project provides a suite of high-performance tools designed to bridge the gap between raw genomic interval data and modern machine learning workflows. We address two of the most significant bottlenecks in the field: the ad hoc nature of vocabulary creation and the inefficiency of tokenization.

First, by providing `uniwig` and companion universe creation methods, we establish a systematic and reproducible foundation for building consensus genomic interval sets. This high-performance preprocessing engine enables the practical application of sophisticated universe construction methods,

such as the LH and HMM approaches. This moves the field beyond simplistic merging or binning strategies and toward the creation of robust, biologically meaningful vocabularies that are tailored for machine learning applications.

Second, the `gtars-tokenizers` library provides the critical link between these vocabularies and the ML ecosystem. Its fast, unified interface makes it easy to integrate into popular ML packages like Hugging Face and PyTorch, and it is also highly useful for traditional applications of interval overlap arithmetic. By providing a fast, ML-aware abstraction, `gtars-tokenizers` helps move the field beyond tool-specific pipelines toward interoperable, general-purpose models of genome function.

Together, these components create a synergistic workflow. Principled vocabularies are of little use without an efficient tool to map to them, and a high-performance tokenizer is ineffective without a well-defined vocabulary. New tools like `gtars` that address the full pipeline will be an important part of the evolving ecosystem that promotes fast, reproducible analysis on genomic regions. Future work that extends this entire approach—from universe creation to tokenization—to other data types like fragments, AnnData objects, bulk ATAC-seq, or even SNPs could reshape how we represent and analyze the genome.

Chapter 4: Fast clustering and annotation of scATAC-seq data using pretrained region embeddings

Nathan J. LeRoy[1,2], Jason P. Smith[1,3,4], Guangtao Zheng[5], Julia Rymuza[1], Erfaneh Gharavi[1,6], Donald E. Brown[6,7], Aidong Zhang[2,5,6], Nathan C. Sheffield[1,2,3,4,5,6,8]

[1]Department of Genome Sciences, School of Medicine, University of Virginia, Charlottesville, VA 22908, USA
[2]Department of Biomedical Engineering, School of Medicine, University of Virginia, Charlottesville, VA 22904, USA
[3]Department of Biochemistry and Molecular Genetics, School of Medicine, University of Virginia, Charlottesville, VA 22908, USA
[4]Child Health Research Center, School of Medicine, University of Virginia, Charlottesville, VA 22908, USA
[5]Department of Computer Science, School of Engineering, University of Virginia, Charlottesville, VA 22908, USA
[6]School of Data Science, University of Virginia, Charlottesville, VA 22904, USA
[7]Department of Systems and Information Engineering, University of Virginia, Charlottesville, VA 22908, USA
[8]Department of Public Health Sciences, School of Medicine, University of Virginia, Charlottesville, VA 22908, USA

Note: This chapter is adapted from the following publication:
LeRoy *et al.*[106], which introduced scEmbed, a method for fast clustering and annotation of scATAC-seq data using pretrained region embeddings.

Introduction to scEmbed

Data from the single-cell assay for transposase-accessible chromatin using sequencing (scATAC-seq) is now widely available. This data is used to uncover complex regulatory networks at the single-cell level, elucidating the cellular mechanisms that drive cell-to-cell heterogeneity. The power of scATAC-seq data has motivated the development of new computational approaches for its analysis[27,30,31,34,35,39–41,108–111]. Despite these advances, scATAC-seq analysis continues to face two key challenges: the 1) high dimensionality and 2) inherent sparsity of the data[112,113].

scATAC-seq analysis often includes two critical tasks: 1) dimensionality reduction followed by clustering and 2) cell-type annotation of cell clusters. For the dimensionality reduction task, numerous methods have been developed, such as SCALE and scBasset, which use variational autoencoders and convolutional neural networks to learn low-dimensional representations of single cells for downstream clustering tasks[39,41]. Other methods include ChromVAR, cisTopic, SnapATAC, and ArchR, which leverage latent semantic analysis (LSI) and topic modeling to cluster individual cells[27,30,35]. These methods usually require complex processing pipelines and large compute power. The second task, cell-type annotation, is less well-served, with most current methods simply repurposing cell-type annotation tools from scRNA-seq[114]. Methods developed for scATAC are few and suffer notable limitations. First, they mainly take a cross-modality approach, integrating data from reference scRNA-seq sets, so they are limited in relying on a secondary data modality. Finding an appropriate secondary dataset to complement the unlabeled set can be difficult[115]. Second, many supervised methods –

while powerful – require model training to predict cell types from a fixed output. This can make the discovery of novel cell types a challenge[111]. Finally, these methods are notoriously compute-intensive[116], a limitation that has grown more problematic as atlas-level datasets have emerged. This problem is further compounded by the fact that many researchers seek to integrate multiple scATAC-seq datasets, which can be computationally prohibitive with current methods.

Here, we address these challenges with an alternative approach to scATAC-seq dimensionality reduction and cell-type annotation using pre-trained embedding models. As discussed in previous chapters, the development of foundation models for scATAC-seq has been hindered by the lack of standardized vocabularies and the challenge of creating meaningful token representations for genomic regions. A natural first step toward building such foundation models is to investigate whether genomic regions can be effectively represented as discrete token embeddings, analogous to word embeddings in natural language processing. scEmbed explores this foundational question by learning fixed, low-dimensional representations of genomic regions that can be shared across datasets, validating the discrete tokenization approach and establishing baseline performance for transfer learning in chromatin accessibility analysis.

Our method improves both dimensionality reduction and cell-type annotation by significantly reducing the computational time and complexity of these workflows with the added benefit of leveraging information from high-quality reference datasets. Instead of analyzing datasets end to end, we use unsupervised learning to model the patterns of regulatory region co-occurrence in high-quality reference datasets, and then transfer this knowledge to new, unseen data. We implemented this method in scEmbed, an unsupervised machine-learning method that learns low-dimensional representations of genomic regions from scATAC-seq datasets. We first show that scEmbed performs as well as established methods for dimensionality reduction and clustering while maintaining robustness to data loss. Moreover, by leveraging models pre-trained on reference data, scEmbed drastically reduces the time and complexity of scATAC-seq analysis with little to no loss in clustering performance. Finally, we build a novel cell-type annotation system by exploiting the learned embeddings produced by pre-trained embedding models without needing any external data modalities. Our system can accurately annotate unseen data in seconds using pre-trained reference models. scEmbed takes a new approach to scATAC-seq analysis by focusing on ATAC-seq data alone, while building high-quality embeddings of genomic regions en route to single-cells, which offers flexibility and speed for a wide range of downstream tasks.

Results

Overview of the scEmbed architecture

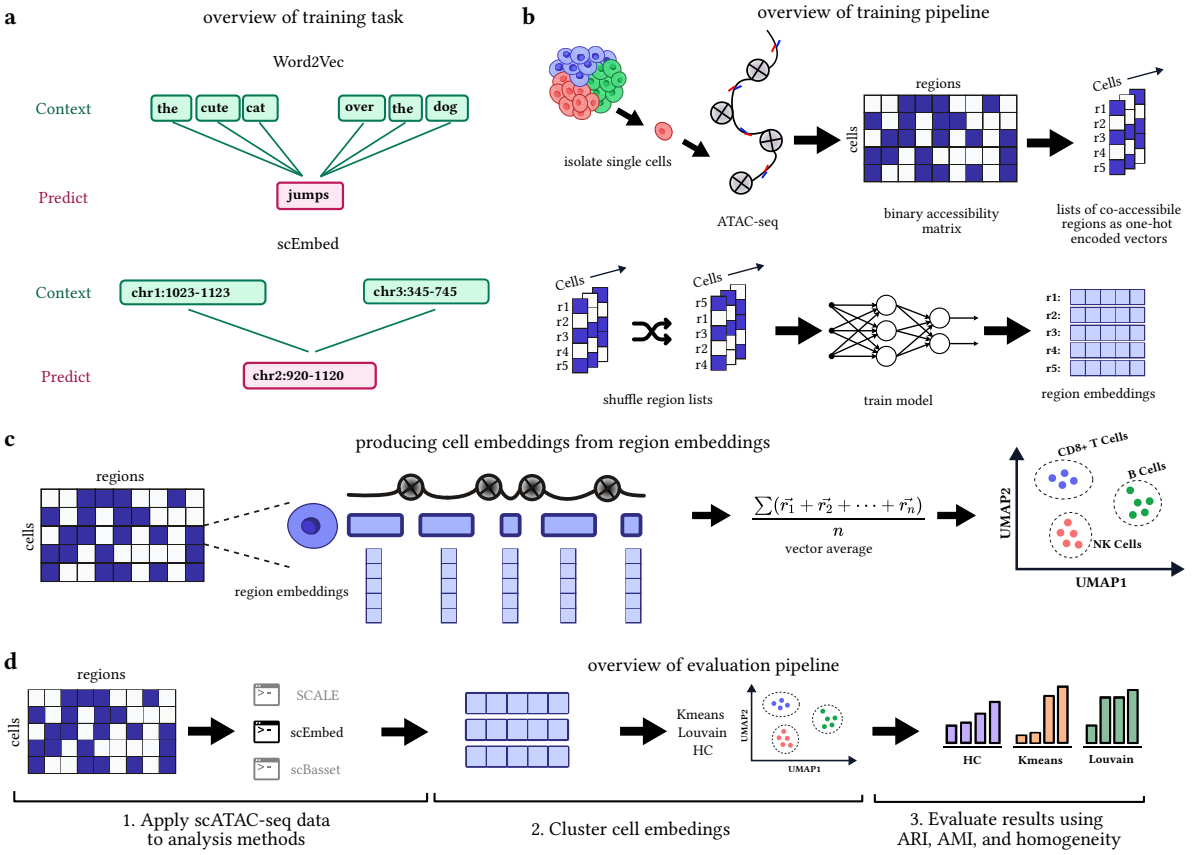


Figure 4.1: An overview of the scEmbed architecture and training procedure.

a. scEmbed leverages Word2Vec as its core model. Word2Vec learns to predict words given a semantic context. Similarly, scEmbed learns to predict genomic regions, given a genomic context. This is unsupervised, and uses the patterns of genomic region co-occurrence to learn representations of individual regions. **b.** Overview of the scEmbed learning process, starting with scATAC-seq data. **c.** Once region embeddings are learned, they can be used to construct cell embeddings by averaging the embeddings of regions accessible in each cell. We use cell embeddings for downstream tasks of clustering and cell-type prediction. **d.** Diagram showing three steps of the benchmarking process

To build a novel neural network that can learn dense, low-dimensional vectors of genomic regions, we designed scEmbed. scEmbed adapts our previous work, Region2Vec[103], to single cells. The model is a modified unsupervised Word2Vec[65] model that learns to predict genomic region co-accessibility similarly to Word2Vec (Figure 4.1A). scEmbed consists of a single embedding layer followed by a context prediction layer for training. Briefly, scEmbed treats each cell as a document and its accessible regions as words. Context is simulated by repeatedly shuffling these regions (Figure 4.1B). We experiment with both skip-gram (SG; predicting context regions given a target region) and continuous bag-of-words (CBOW; predicting a target region given its context) training objectives. After training, we are left with a set of region embeddings; one for each genomic region in the model vocabulary. To

construct cell embeddings, we average region embeddings for each cell, which are then used for tasks like clustering, analysis, or transfer learning (Figure 4.1C).

To validate scEmbed, we followed an earlier approach[112] by Chen *et. al.* to benchmark the model on clustering tasks using published reference scATAC data. We experiment with a diverse set of both real and simulated scATAC-seq datasets with known and unknown cell-type labels. These embeddings are clustered using three clustering methods: K-means, hierarchical clustering (HC), and Louvain clustering. The clusters are then subjected to evaluation using established metrics (Figure 4.1D) – namely adjusted mutual information (AMI), adjusted rand index (ARI), and homogeneity (see Methods). Finally, we explore the potential of scEmbed for transfer learning tasks like cell clustering and cell-type annotation by repurposing existing models.

scEmbed is competitive with existing scATAC-seq methods

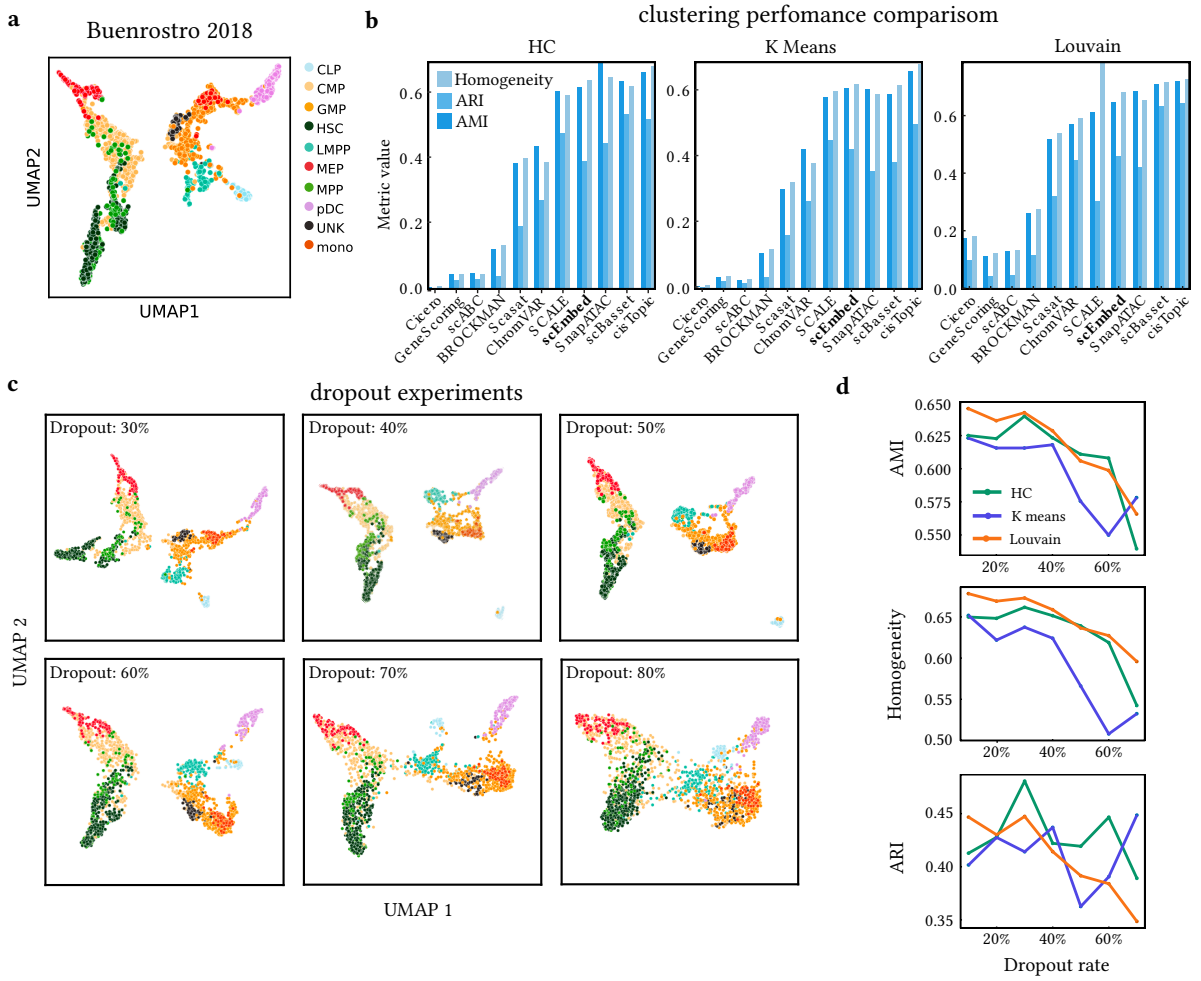


Figure 4.2: Benchmarking shows that scEmbed is competitive with existing approaches.

a. UMAP plot of the Buenrostro2018 dataset cell-embeddings produced by scEmbed. **b.** Results of the benchmarking pipeline. scEmbed is competitive with the top methods. We tested three clustering methods: Hierarchical clustering (HC), K means, and Louvain. The clustering results were evaluated using three metrics: Adjusted mutual information (AMI), adjusted rand index (ARI), and homogeneity. **c.** UMAP plots

visually showing the resultant clusters of cell embeddings produced by scEmbed following data loss. **d.** Line plots showing the change in three clustering metrics (ARI, AMI, and Homogeneity) as a function of dropout rate. Plots show that scEmbed retains its ability to accurately cluster single cells up to nearly 80% data loss.

When benchmarked against *Buenrostro2018*[117], scEmbed visually clusters cells of the same type. (Figure 4.2A). scEmbed performed similarly to the best-performing scATAC-seq methods, including SCALE, scBasset, cisTopic and SnapATAC (Figure 4.2B). This performance was achieved with minimal preprocessing of the data and a completely unsupervised learning workflow. In addition to the *Buenrostro2018* dataset, we also benchmarked scEmbed on another, more recent and comprehensive scATAC-seq dataset from Luecken et al.[118]. Again, comparing clusters with ground truth labels, scEmbed performs well (Supplementary Figure A.2).

scEmbed is robust to data loss

Next we wondered if we could leverage scEmbed for transfer learning tasks, which can result in a loss of information. As such, we sought to evaluate its ability to cluster data with increasing levels of information loss. To test scEmbed's robustness to missing data, we trained the model on datasets of increasing sparsity. Starting with the *Buenrostro2018* dataset (2.8% non-zero entries)[117], we randomly dropped non-zero values in the binary accessibility matrix until approximately 80% of the initial non-zero data was lost. A dropout rate of 80% resulted in a matrix that was 0.5% non-zero. Even at a drop-out rate of 80%, scEmbed was able to visually cluster cells of the same type (Figure 4.2C). To quantify this, we computed the same three clustering scores for each dropout dataset: 1) Adjusted Rand Index (ARI), 2) Adjusted Mutual Information (AMI), and 3) Homogeneity scores. We found that scEmbed retained clustering accuracy comparable to other scATAC-seq analysis methods[112] even when faced with 80% data loss (Figure 4.2D). These findings confirm that scEmbed can learn rich biological knowledge, even for the most sparse datasets. We hypothesize that this robustness arises from the model's ability to capture redundant regulatory information across genomic regions. Even with substantial data loss, the remaining accessible regions likely contain sufficient co-accessibility patterns to maintain the learned relationships between functionally related regulatory elements. The ability to handle sparseness is a critical characteristic of scATAC-seq analysis, and particularly so for scEmbed, which can be used to transfer information from existing models, as we describe next.

Using scEmbed to transfer knowledge of genomic region co-occurrence to unseen datasets

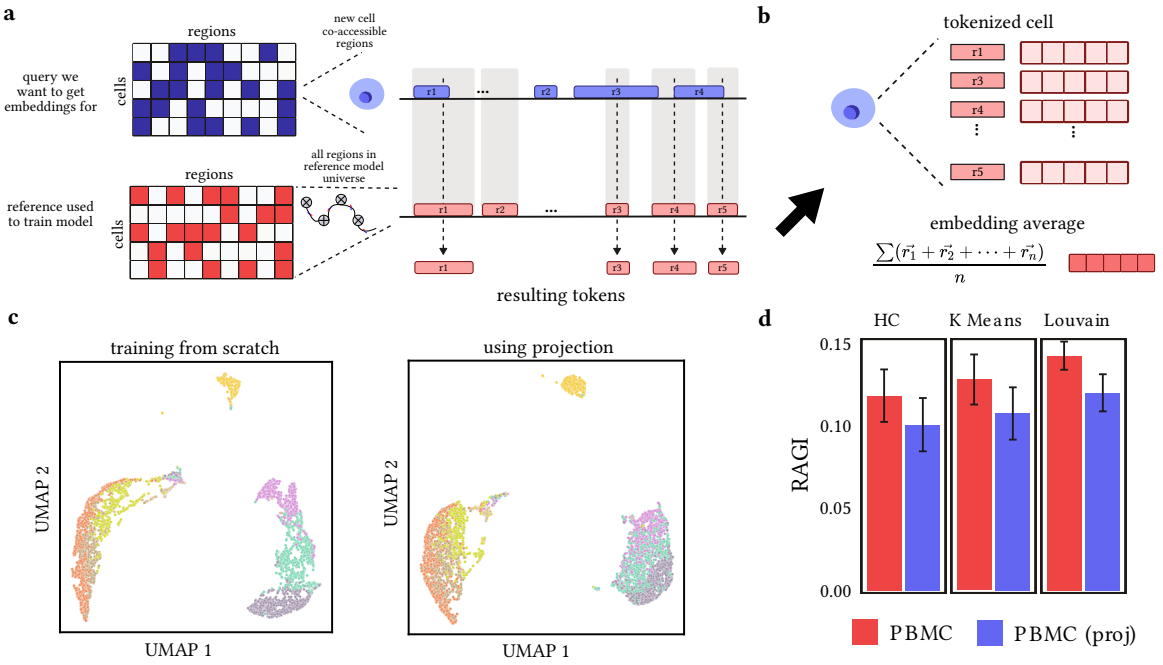


Figure 4.3: scEmbed enables knowledge transfer to unseen datasets.

Transfer learning with scEmbed occurs in three steps. **a.** Diagram of overlap analysis depicting how a new cell from a new dataset (blue) is tokenized into the feature space of the data used for the pre-trained model (red). **b** Diagram showing the computation of embeddings for new, unseen data. This is achieved using average pooling of region embeddings. **c.** UMAP plots of both projected (right) and unprojected (left) datasets. The plots show nearly identical clustering of embeddings learned from the original dataset versus projection. **d.** RAGI score plots for both the original dataset embeddings and projected cell embeddings. RAGI scores are computed for three clustering methods: Hierarchical clustering, K-means, and Louvain.

A key innovation in scEmbed is its two-step training process, rather than the common single-step approach. In the first step, scEmbed learns embeddings of genomic regions rather than cells. In the second step, these region embeddings are aggregated to construct cell embeddings. This decoupled architecture enables a critical capability: the learned region embeddings can be reused to generate embeddings for entirely new datasets that the model has never seen, allowing scEmbed to leverage knowledge from pre-trained reference models (Supplementary Figure A.3). We call this process of using a pre-trained model to generate embeddings of new data "projection".

We next sought to assess this projection process by asking whether scEmbed could cluster a new dataset based entirely on a pre-trained model. Put another way, we wanted to ask: *how well can scEmbed cluster cells when we re-use a model we just trained for brand new dataset?* To assess this, first, we trained a model on the original Buenrostro2018 dataset[117]; second, we took a new dataset, 10X genomics 5k PBMCs from a healthy donor, and tokenized each cell's regions into the original models vocabulary (Figure 4.3A), followed by region pooling via average pooling (Figure 4.3B). Tokenization

is a critical step in this process and involves mapping the genomic regions of the new dataset to those in the vocabulary of the pre-trained model (see Methods).

We used these single-cell embeddings directly for UMAP visualization and clustering analysis. To assess the quality of the projection, we assumed that 8 distinct cell populations existed and took advantage of marker gene analysis to assign labels to each cell. We use the Residual Average Gini Index (RAGI) score to evaluate the clustering ability of scEmbed[112] which enables us to assess clustering performance when ground truth labels are unknown (see Methods). We found that the projected cell analysis showed no marked differences in clustering proficiency when compared with the embeddings produced by conventional model training. The UMAP plots were visually similar (Figure 4.3C), indicating similar clustering performance.

To further explore the difference, we next evaluated clustering performance using a repeated subsampling strategy that consisted of four steps: (i) train a new model on the PBMC data alone; (ii) repeatedly subsample 1000 cells and compute their embeddings using the new model and the Buenrostro2018 model using projection; (iii) cluster the cells using three strategies (HC, K-means and Louvain); and (iv) compute the RAGI score with these 1000 subsampled cells. The scores were then averaged across all subsamples. Our results showed that the RAGI score between the original and projected datasets did not differ significantly, indicating similar clustering performance (Figure 4.3D, Supplementary Figure A.4). Taken together, these results suggest that scEmbed can effectively and consistently transfer knowledge from a pre-trained model to new datasets with minimal loss in clustering performance.

Pre-trained models from reference datasets can be used to annotate cell clusters

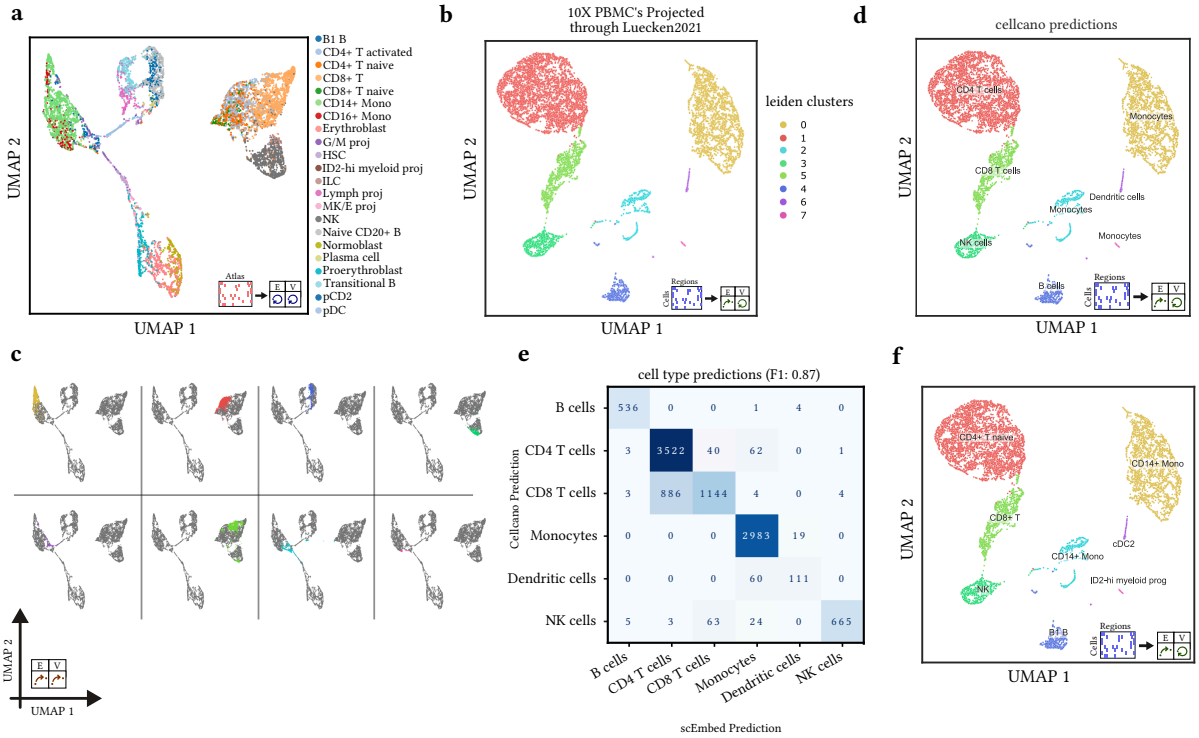


Figure 4.4: Pre-trained embedding models can be exploited for cell-type annotation tasks.

a. Diagram showing scEmbed's three projection paths. **b.** Overview of the standard "no-projection" data flow. **c.** Overview of three data flows for new data. EV-projection places the new data in the same latent space as the reference data. **d.** UMAP plot of the reference data embeddings built using the standard workflow. **e.** UMAP plot of the new PBMC data with E-projection. **f.** Plots showing the EV-projection data flow applied to the new PBMC dataset. Grey cells represent the reference topology; colored cells are projected new PBMC data. Separate plots depict individual clusters for visual clarity. **g.** Confusion matrix of scEmbed classification results compared to Cellcano. **h.** UMAP plots showing the cell labels assigned by Cellcano (left) and the cell type labels assigned by scEmbed (right).

To clarify the three projection methods used in scEmbed, we define each approach and its specific use case. The "no projection" method represents the standard workflow where a model is trained directly on the query dataset, and embeddings are generated by passing the same data through the trained model. "E-projection" involves training a model on reference data and then generating embeddings ("E") for new, unseen query data using the pre-trained region embeddings. This approach enables knowledge transfer without retraining. "EV-projection" extends E-projection by additionally training a UMAP model on the reference dataset embeddings, allowing new query data to be both embedded using the pre-trained model and visualized ("V") within the reference dataset's UMAP topology. This unique capability enables researchers to directly compare query cells to reference populations in a shared embedding space, facilitating intuitive cell-type annotation and biological interpretation (Supplementary Figure A.3).

Now, convinced that pre-trained models could be used to visualize an unseen query dataset, we next asked whether this approach could be used to annotate cell types without training a new model specifically for cell-type annotation. Specifically, we sought to leverage reference mapping and vector similarity to transfer labels from a reference set to a query set. We first built a reference model using the Luecken2021 multi-omic dataset[118], a first-of-its-kind multimodal benchmark dataset of 120,000 single cells from the human bone marrow of 10 diverse donors measured with two commercially available multimodal technologies. Using scEmbed and the 'no projection' data flow (see methods), we trained a model on the dataset and clustered the resulting learned embeddings (Figure 4.4A). This model served as the reference for all downstream experiments with a new PBMC dataset from 10X genomics. Using E-projection, scEmbed creates visually distinct clusters of single cells (Figure 4.4B). To visualize these embeddings in the context of the original embedding topology, we employ EV-projection where we first train a UMAP model on the original Luecken2021 embeddings and then project the new PBMC data into this UMAP space using the pre-trained Luecken2021 model. We found that each identified cluster from E-projection aggregates to a distinct location in the original UMAP embedding topology of the Luecken2021 model (Figure 4.4C). This suggests that the new PBMC data is being meaningfully integrated into the reference topology, allowing for direct comparison between the new and reference datasets.

Confident our pre-trained Luecken2021 embedding model was distinctly clustering the new PBMC dataset, we sought to assign cell-type labels to each cluster. We used Cellcano, a new scATAC-seq cell annotation method, to assign ground-truth labels to each cluster of the E-projected PBMC embeddings for evaluation of our method[111] (Figure 4.4D). Our cell-type annotation system was limited by the cell types annotated in Luecken2021; as such, we mapped each scEmbed prediction class to a corresponding Cellcano class for comparison (Supplementary Table A.1) after following their annotation procedure (see the scEmbed materials and methods; Supplementary Figure A.5). Using a simple k-nearest-neighbor (KNN) classification algorithm, scEmbed was highly consistent with the Cellcano labels (F1 = 0.87, Figure 4.4E). However, without class mapping, scEmbed offers higher specificity of cluster identity and even identifies a cluster of ID2-hi myeloid progenitor cells not found with Cellcano (Figure 4.4F). Moreover, this workflow enables researchers to quickly try new models trained on many different cell types and rapidly discover cell types in their data. Using our projection system, researchers can avoid training a new model each time they want to use a new reference dataset, which is a common approach in many modern cell-type annotation systems[40,111,119]. The entire process of dimensionality reduction, clustering and annotation took $<$10 min on a laptop,

and we observed similar time savings across several models. Thus, we conclude that EV-projection is a promising approach for fast visualization and annotation of new data.

Discussion and future work

In this work, we demonstrate the robustness and versatility of scEmbed, a new tool for the analysis of scATAC-seq data. scEmbed differs from existing methods in that instead of learning embeddings of individual cells directly, it first learns embeddings of genomic regulatory regions and then uses these to compute cell embeddings. We demonstrate how this approach allows scEmbed to use pre-trained genomic region embedding models to effectively cluster data not seen by the model. Our evaluation of scEmbed against existing scATAC-seq methodologies demonstrates its efficacy and competitiveness, even with a relatively simple network architecture. scEmbed performs well, even when faced with severe data loss. The standout feature of scEmbed is its capability to repurpose learned region embeddings for downstream analysis tasks.

This approach provides flexibility and efficiency, setting it apart from other currently available tools. By exploiting region overlaps and applying previously learned region embeddings, we have formulated a novel method for representing unseen scATAC-seq data within the latent space of the original training data. This process, termed 'projection', yielded superb clustering of cells, showing no significant decrease in performance compared with models trained entirely on the new dataset. This performance underscores the potential of scEmbed in the context of ATAC-seq transfer learning tasks and opens up exciting possibilities for future research. Moreover, we emphasize the novelty of scEmbed in its ability to engage in transfer learning without the need for another data modality like scRNA-seq, which overwhelmingly required by current methods. Future studies will explore the ability of our model to learn and extract overarching regulatory patterns from publicly available data. This learning, coupled with the inherent transferability of scEmbed, will empower researchers to fine-tune the models for specific downstream tasks, enabling gains in performance, efficiency, and flexibility.

Finally, we leveraged embeddings computed by scEmbed and its pre-trained models to build a novel cell-type annotation system. Our method is consistent with current scATAC-seq cell-type annotation implementations, with the added advantage of requiring no external data modalities. Furthermore, by exploiting pre-trained models and pre-computed cell embeddings from reference datasets, the scEmbed annotation system can easily scale to millions of cells and uses only a fraction of the compute time. This is because utilization of a pre-trained model requires only interval overlap analysis to map the new data into the feature space on which the model was trained. We have made the pre-trained

models used in this study available for download and use on huggingface. To facilitate model sharing and usability even further[93], we have built software packages to easily download and use these models within Python. Moreover, these same packages can be used to train new models or fine-tune public ones on custom datasets. We hope that these resources will enable researchers to leverage the power of scEmbed for their own research.

The integration of unsupervised learning with transfer learning may offer new directions for other bioinformatics tasks that are similarly burdened by the challenges of high dimensionality and data sparsity. Furthermore, the deployment of pre-trained models for reference datasets may inspire novel methodologies for efficient and accurate cell-type annotation systems across different data modalities. In the future, the pre-training approach of scEmbed could be adapted for use with cross-modality methods that span data types[104]. In addition, more advanced deep learning architectures and training strategies like transformers could be integrated to further enhance the model's performance and applicability. In conclusion, scEmbed's ability to distill meaningful representations from vast, complex scATAC-seq datasets, and repurpose this knowledge for rapid and accurate analysis of new datasets, has great potential. This work is a step towards developing more efficient, scalable, and flexible tools for genomic data analysis. The opportunities unlocked by scEmbed for research and clinical application promise exciting advancements in the comprehension of cellular heterogeneity and the intricate regulatory networks that drive it.

Chapter 5: Atacformer: A transformer-based foundation model for analysis and interpretation of ATAC-seq data

Nathan J. LeRoy[1,2], Guangtao Zheng[3], Oleksandr Khoroshevskyi[1], Donald Campbell[1], Aidong Zhang[3], Nathan C. Sheffield[1,2]

[1]Department of Genome Sciences, School of Medicine, University of Virginia, 22908, Charlottesville VA
[2]Department of Biomedical Engineering, School of Medicine, University of Virginia, 22908, Charlottesville VA
[3]Department of Computer Science, School of Engineering and Applied Sciences, University of Virginia, 22908, Charlottesville VA

Note: This chapter is adapted from the following preprint:
LeRoy *et al.*[120], which introduced Atacformer, a transformer-based foundation model for analysis and interpretation of ATAC-seq data.

Introduction to Atacformer

The exponential growth of publicly available genomic data has spurred the development of powerful pre-trained foundation models across diverse genomic modalities, including DNA sequences (e.g. Enformer[52] and DNA Discrete Diffusion[56]); gene expression (e.g. Geneformer[58] and scGPT[57]); and protein folding (e.g. AlphaFold[53–55]). By leveraging transfer learning, these models have dramatically enhanced data integration capabilities, enabling researchers to apply knowledge learned from large-scale datasets to new, often smaller-scale tasks.

Despite significant progress across genomics, transcriptomics, and proteomics, the development of foundation models explicitly tailored for epigenomic data remains comparatively underexplored. The Assay for Transposase-Accessible Chromatin sequencing (ATAC-seq) and its single-cell variant (scATAC-seq)[18,24] have emerged as key methods for interrogating the regulatory landscape of the genome, offering insights into disease mechanisms, cellular heterogeneity, and tissue development. However, these assays present computational challenges due to their high dimensionality and inherent data sparsity[92,121]. Consequently, a wide array of computational tools have been developed to manage these complexities and simplify analysis pipelines for researchers. Comprehensive platforms, including ArchR[35] and SnapATAC[31,37], provide scalable, end-to-end solutions that support many standard workflows, such as quality control, dimensionality reduction, and trajectory analysis. More advanced, specialized tools like cisTopic provide a probabilistic topic modeling framework for discovering co-accessible enhancers and regulatory patterns, while deep learning approaches like SCALE[39] and scBasset[41] seek to address fundamental challenges of data sparsity and learning relationships between DNA sequence and chromatin accessibility.

These tools have pushed forward the analysis of scATAC-seq data. However, current methods for scATAC-seq are specialized, task-specific, and limited to the immediate dataset under investigation. They fail to exploit the shared biological knowledge encoded in the many publicly available datasets. No method has focused on producing a foundation model, trained on vast and diverse datasets, designed to be adapted to both new models through downstream fine-tuning and to new datasets through transfer learning. Recognizing this limitation, new models for scATAC-seq are emerging. This includes ChromFound[63] and EpiAgent[62], which are the first foundation models explicitly tailored for scATAC-seq, utilizing large-scale pre-training on millions of cells. Despite their advances, these models have several limitations. First, they are confined to single-cell scATAC-seq data and do not handle bulk ATAC-seq region-sets, which are widely used in many contexts. Second, they do not model genomic regions as discrete tokens, opting instead for continuous representations of cells that limit interpretability. Third, they rely on very large architectures – for example, EpiAgent uses over 1 billion parameters across its embedding and transformer modules, which requires substantial GPU resources for inference and adoption. Fourth, they do not explore multimodal contrastive integration with scRNA-seq, which restricts cell-type transfer and cross-modal inference capabilities[122].

As demonstrated in the previous chapter, scEmbed validated the feasibility of treating genomic regions as discrete tokens and showed that fixed region embeddings could effectively support transfer learning for scATAC-seq analysis. However, scEmbed also revealed a fundamental limitation of static embeddings: they encode a single, universal representation for each genomic region that remains constant regardless of cellular context. This approach fails to capture the dynamic, cell-state-specific nature of gene regulation, where the functional role of a regulatory element depends heavily on the surrounding chromatin landscape and the other regions with which it interacts. A transcription factor binding site near an active promoter in a stem cell may have a vastly different regulatory impact than the same site in a differentiated neuron. To address this limitation and move toward a true foundation model for scATAC-seq, we require contextualized embeddings that adapt based on the regulatory context in which each region appears – precisely the capability that transformer architectures with self-attention mechanisms provide.

To address these limitations, we present Atacformer, a transformer-based foundation model. Atacformer leverages large-scale pre-training on single-cell ATAC-seq data. Furthermore, unlike existing approaches, we model and place emphasis on genomic intervals as discrete tokens – the fundamental "words" of the regulatory genome. This token-based representation aligns with biological intuition and allows Atacformer to exploit the strengths of transformer architectures, which are particularly well-suited for learning from sequences of discrete inputs, as demonstrated

in Natural Language Processing. Beyond the Atacformer model, we also introduce a dual-encoder contrastive learning approach called Contrastive RNA-ATAC-Fine-Tuning (CRAFT), which supports cross-modal alignment between scATAC-seq and scRNA-seq data. We benchmarked the performance of Atacformer and CRAFT models across four key applications. First, we assessed zero-shot cell clustering on multiple new, unseen PBMC datasets after fine-tuning the model on a cell-type clustering task. Next, we tested direct fragment file processing to measure speed and biological concordance without conventional preprocessing. Third, we applied Atacformer to bulk region-set data to evaluate embedding quality and metadata prediction. Finally, we leveraged the token-level embeddings to investigate putative weak regulatory interactions. Our results demonstrate that Atacformer and CRAFT can achieve state-of-the-art performance with best-in-class runtime for scATAC-seq analysis while maintain powerful, zero-shot clustering performance on new, unseen datasets.

Results

Atacformer is a new transformer-based foundation model for ATAC-seq data

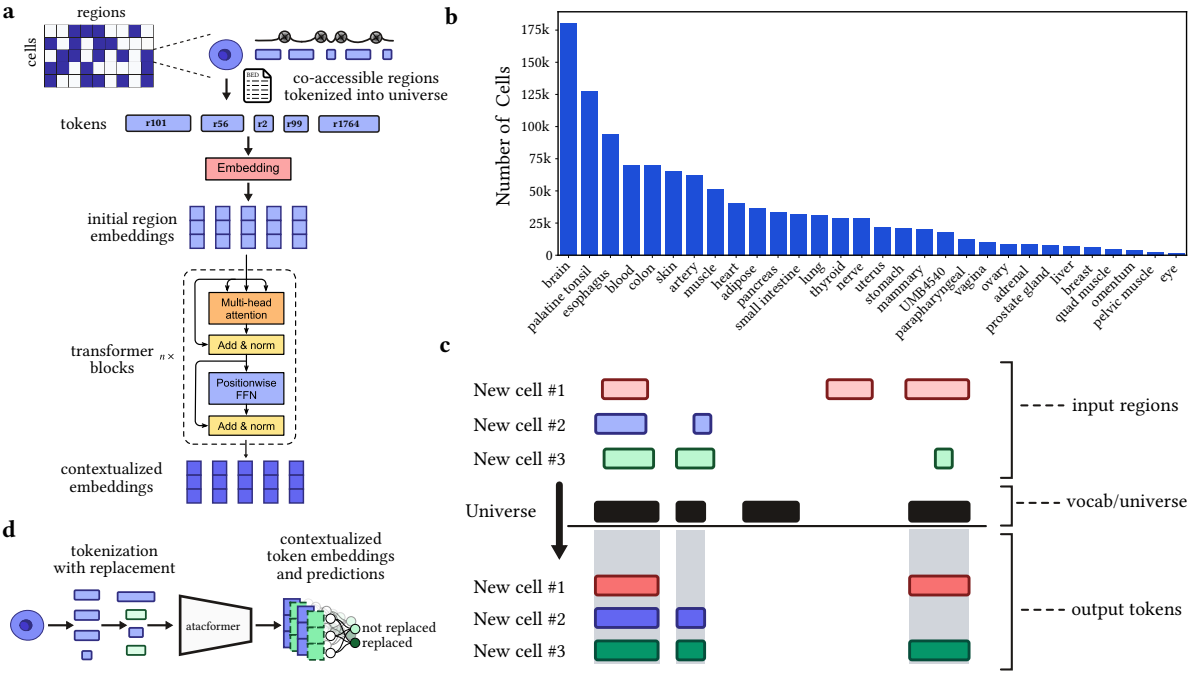


Figure 5.1: An overview of the Atacformer architecture and training procedure.

a. Model architecture and pretraining schematic for Atacformer. Individual cells are tokenized into the model universe, followed by random token replacement. These tokens are then passed to the embedding module, followed by n transformer blocks to generate contextualized embeddings. **b.** Tissue distribution and representation in the scATAC atlas used in Atacformers pretraining. **c.** The Atacformer tokenization strategy. New cells are tokenized into the model vocabulary using interval overlap analysis.

Atacformer is a transformer-based[50] foundation model that produces contextualized embeddings of genomic regions. Unlike existing models that are large, rigid, and task-specific, Atacformer is general-

purpose and lightweight: users need only provide chromosome coordinates (chromosome, start, end) to begin analysis. By minimizing assumptions about the data and streamlining inputs, Atacformer serves as a fast, efficient backbone that can be adapted to virtually any genomic interval task. Atacformer consists of two main components: a genomic region embedding module and a stacked transformer encoder layer with multi-head attention (Figure 5.1A). To train Atacformer, we curated a single-cell ATAC-seq atlas consisting of 1.2 million cells and $>$ 10 billion tokens from 30 tissues (Figure 5.1B; Supplementary Table A.2)[123–128]. We uniformly processed all raw datasets using a standardized pre-processing pipeline to ensure data integrity and compatibility (see Methods). We used these uniformly processed results to create a unified consensus vocabulary based on our earlier work[90] consisting of 890,704 distinct genomic regions (see Methods).

To tokenize a single cell into a set of dense, low-dimensional embeddings, we first map each accessible region in the cell to a corresponding region in the model's vocabulary through simple interval intersection (Figure 5.1C). This process transforms noisy, unstandardized genomic intervals into fixed tokens while preserving the biological significance of co-accessibility patterns. To enable extremely fast, in-memory tokenization that supports modern machine learning workflows, we developed a set of Rust-based tokenizers to be used in conjunction with Atacformer[91].

Atacformer is trained using an ELECTRA-style pre-training objective[129] (Figure 5.1D) in which Atacformer receives tokenized region sets in which a random subset of tokens are replaced with others sampled from the vocabulary. The model is then tasked with predicting which tokens were replaced. Unless noted, we pre-trained Atacformer using a 45% token replacement rate and a context window of 8,192, as this captures the majority of co-accessible regions in all single-cells in our corpus (Supplementary Figure A.7). We refer to this model as atacformer-base. We evaluated atacformer-base to establish a performance baseline, and then fine-tuned it to achieve better performance and enable more flexible downstream analyses.

Atacformer can be paired with Geneformer for powerful multiomics analysis

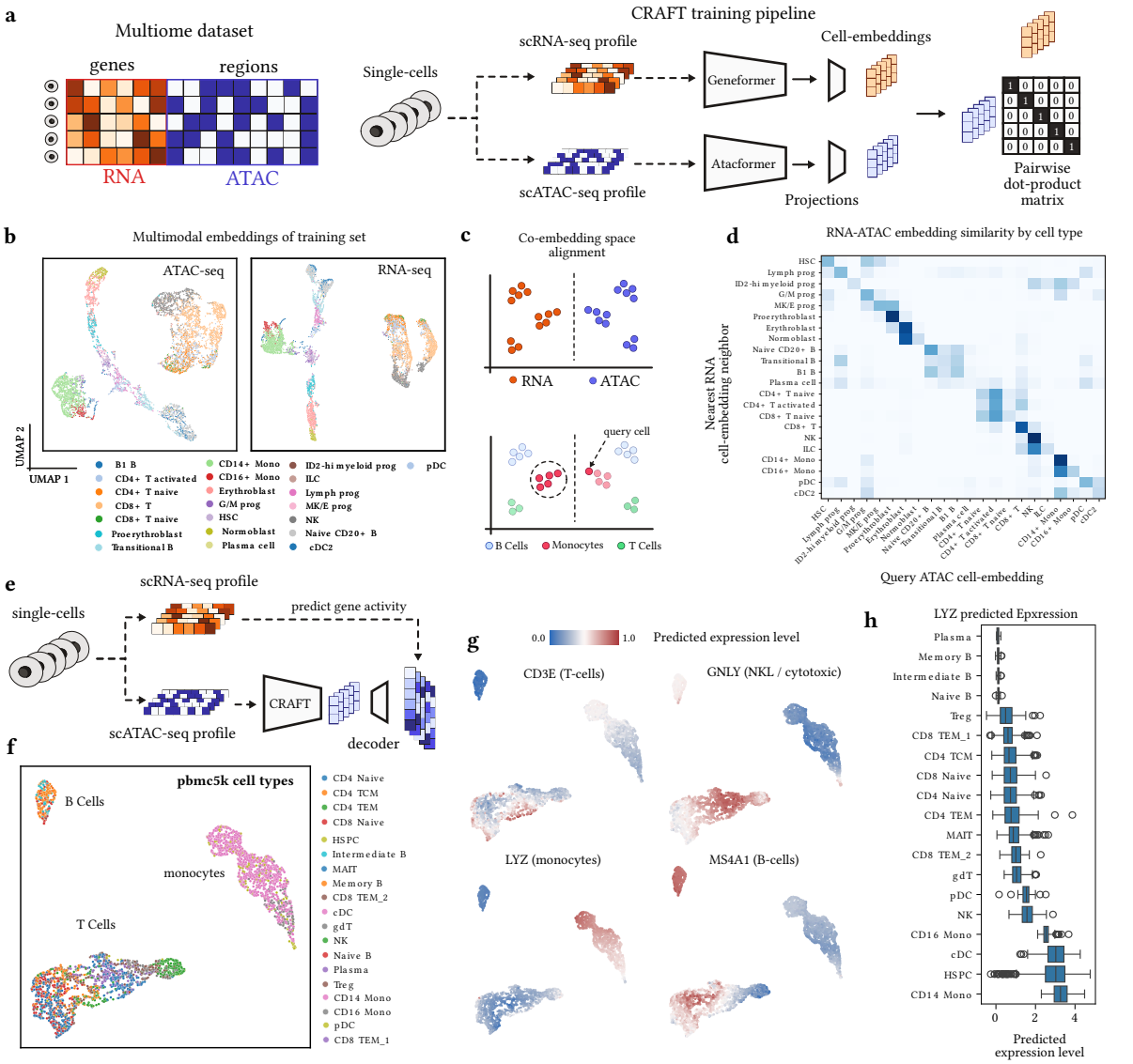


Figure 5.2: CRAFT is a powerful dual-encoder, multimodal single-cell embedding model.

a. Schematic of the CRAFT training procedure. (Left) Schematic of a multiomic single-cell dataset showing ATAC and RNA signal jointly profiled in a single cell. (Right) Training step for a single mini-batch of cells. **b.** UMAP visualizations of the single-cell embeddings generated using the ATAC-encoder (left) and the RNA encoder (right). **c.** Schematic of the learned co-embedding space after training. **d.** Heatmap of nearest RNA-embedding neighbors to a single ATAC-embedding cell, organized by cell type. **e.** Schematic of the RNA-decoder training procedure. **f.** PBMC5k dataset clustered using ATAC-embeddings; colored by cell-type. **g.** Heatmaps of PBMC5k dataset, colored by predicted RNA-expression profile for four different marker genes. **h.** Distribution of predicted LYZ expression levels across cell types.

We sought to fine-tune Atacformer on a multimodal task. Recent breakthroughs, such as CLIP[44], demonstrate that aligning fundamentally different data types in a shared latent space enables zero-shot transfer and cross-modal retrieval. Inspired by this, we investigated whether Atacformer would benefit from being paired with an encoder of another modality, such as scRNA-seq. To investigate, we developed Contrastive RNA-ATAC Fine-Tuning (CRAFT), a multi-modal model that

combines Atacformer with Geneformer, a transcriptome encoder[58]. CRAFT uses contrastive training on multiomic data to create a shared latent space for scATAC-seq and scRNA-seq data. We initialized the model with pre-trained Atacformer and Geneformer models. We then trained CRAFT using a multiomic dataset containing over 106,000 cells profiled with scATAC and scRNA simultaneously (Figure 5.2A; see Methods).

UMAP visualizations of ATAC-seq and RNA-seq embeddings were topologically similar, highlighting modality alignment (Figure 5.2B). Embeddings maintained biologically meaningful relationships; for example, the distinct groups of monocytes and CD8+ T cells are maintained in both modalities, as is a clear linear trajectory of the stages of red blood cell development from proerythroblast to erythroblast to normoblast . When projected into a single two-dimensional UMAP, modalities separated visually (Supplementary Figure A.8); but despite this, nearest neighbors in higher-dimensional spaces preserved biological similarities, allowing accurate cross-modal cell-type alignment (Figure 5.2C). To assess biological coherence, we projected ATAC embeddings into the multimodal CRAFT latent space and queried their nearest RNA neighbors. Consistently, the local RNA neighborhoods aligned with the same cell type as the corresponding ATAC embedding. (Figure 5.2D).

Next, we assessed whether the aligned ATAC embeddings encoded transcriptomic information by training a small decoder to predict RNA-seq profiles from ATAC embeddings (Methods, Figure 5.2E). This should allow us to leverage the joint CRAFT embeddings to predict scRNA-seq outputs from datasets where only scATAC-seq was assayed, or vice versa. We applied this decoder to an unseen scATAC-seq dataset with known labels, labeled with scVI (see methods). UMAP projections of the ATAC embeddings visually separated cells into clusters for T Cell, B Cell, and Monocyte lineages (Figure 5.2F), showing that these out-of-sample cells are distinguished by the model. Next, we applied the RNA-seq decoder to predict scRNA-seq profiles. The predicted RNA-seq profiles accurately recapitulated known gene expression differences among these lineages, including cell-type-specific markers. For example, the imputed expression of monocyte marker LYZ was elevated in monocytes; B cell marker MS4A1 was elevated in B cells; T cell marker CD3E was elevated in T cells; cytotoxic cell marker GNLY was elevated in cytotoxic cells (Figure 5.2G). Finally, we examined the predicted normalized expressed across cell-types quantitatively. As an example, we show that the monocyte marker LYZ is disproportionately up-regulated in Monocyute-like cells (Figure 5.2H). Collectively, these results demonstrate that RNA-seq patterns have been incorporated in the ATAC-seq embeddings of the multi-modal CRAFT model.

In total, these results demonstrate that CRAFT effectively integrates Atacformer embeddings with complementary single-cell models, enabling robust multimodal analysis and cross-modal biological inference. This dual-encoder framework accurately aligns chromatin accessibility and transcriptomic data within a unified latent space, facilitating precise cell-type identification and enhancing the interpretability of single-cell chromatin profiles.

Fine-tuned Atacformer models and CRAFT enable fast and accurate zero-shot cell-clustering

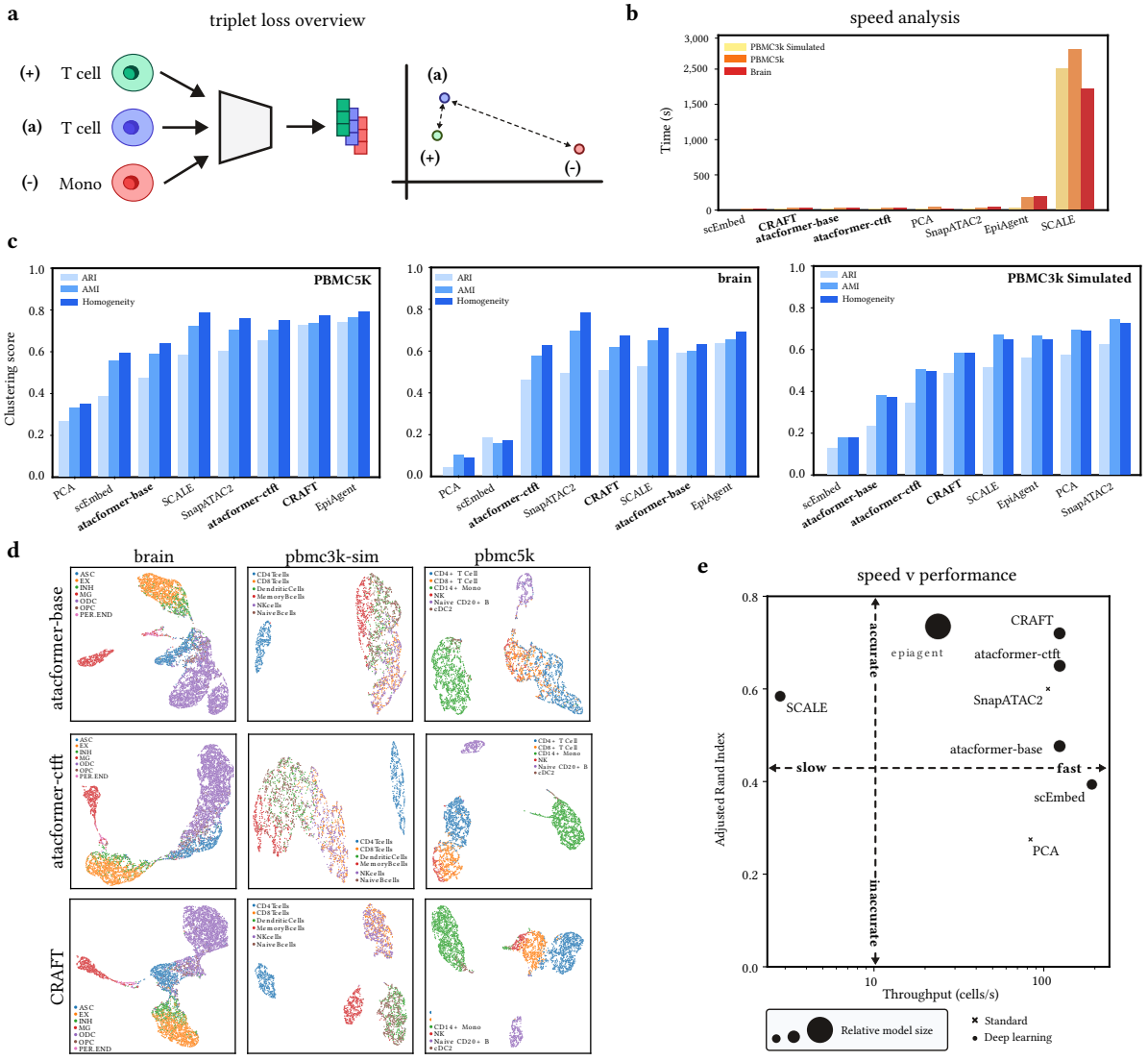


Figure 5.3: Atacformer clusters new scATAC data accurately in a zero-shot approach.

a. Schematic of the supervised cell-type fine-tuning task using triplet loss. **b.** Runtime comparisons between Atacformer and other popular methods for clustering scATAC-seq data. **c.** Clustering performance of Atacformer-base and Atacformer-ctft on three separate PBMC datasets compared with other popular methods for clustering scATAC-seq data. **d.** UMAP plots highlight the latent space change when fine-tuning Atacformer on cell types. **e.** Runtime versus ARI chart for Atacformer and other popular methods.

The first bottleneck in scATAC-seq analysis is cell clustering. Most current tools either retrain on every dataset or take minutes per thousand cells. We sought to improve cell-clustering without sacrificing

speed or requiring any additional processing. A pre-trained Atacformer model could possibly cluster unseen data very quickly; however, we reasoned that a model trained on an unsupervised token replacement prediction task would perform sub-optimally for a cell-type clustering task. To that end, we sought to improve the single-cell embeddings in two ways: First, inspired by the Sentence-BERT (SBERT) family of text embedding models[130], we designed a novel fine-tuning approach for cell-type clustering. Starting with the base model (`atacformer-base`), we trained a **cell-type fine-tuned** version, `atacformer-ctft` using triplet loss to position cells of the same type together in the latent space (Figure 5.3A; see Methods). We used the Luecken2021 multi-omics PBMC dataset for fine-tuning, which provides high-confidence cell-type labels across matched modalities[118]. Second, we utilized our previously described CRAFT model. Our reasoning was that the resultant single-cell embeddings from CRAFT would be much higher quality through mutual refinement of ATAC-seq and RNA-seq signal – a phenomenon seen in other contexts[131,132].

To assess both the base and fine-tuned models, we collected three separate datasets for evaluation: 1) A third-party, pre-annotated brain dataset, 2) a simulated PBMC dataset from bulk ATAC-seq data, and 3) a pre-annotated PBMC dataset. We generated cell embeddings for all cells in each dataset, and then used them to benchmark `atacformer-base`, `atacformer-ctft` and CRAFT against Principal Component Analysis (PCA) and several popular methods for scATAC-seq clustering[37,39,62,106]. To establish Atacformer's capabilities for clustering data it's never seen before (i.e. zero-shot clustering), all datasets were explicitly excluded from the training data for both `atacformer-base`, `atacformer-ctft`, and CRAFT.

We found Atacformer to be one of the fastest methods evaluated; the only faster method being our previous Word2Vec-based method scEmbed[106] (Figure 5.3B). The slowest methods were those that required the use of a very large model (EpiAgent, 1.5B parameters[62]), or required training from scratch on a combined dataset with both the original and new data (SCALE).

To evaluate clustering accuracy, we clustered the embeddings obtained from each method to obtain cluster labels. These labels were then compared to the ground-truth cell-type labels using three metrics: Adjusted Rand Index (ARI), Adjusted Mutual Information (AMI), and Homogeneity score (see Methods). As expected, the atacformer-base model underperformed other methods for PBMC-based datasets. The cell-type fine-tuned model increased clustering performance according to ARI by $\sim$ 15% across the PBMC datasets (Figure 5.3C). These gains were also reflected in the training dataset for the fine-tuning procedure (Supplementary Figure A.10). Notably, `atacformer-base` performed exceptionally well on the brain dataset, surpassing the cell-type fine-tuned model. We reasoned that

this was because the cell-type fine-tuned atacformer model was fine-tuned only on blood data, hurting its performance. Finally, we found CRAFT to perform very well across the datasets, highlighting the power of contrastive learning for embedding quality improvement. To further explore how the fine-tuning step improved the base model for this task, we compared the embeddings of the two atacformer models visually using UMAP. The clustering score gains are clearly reflected in the UMAPs. For example, the cell-type fine-tuning approach significantly improves Atacformers ability to cluster the NK-cells along with differentiating between CD4+ and CD8+ T cells (Figure 5.3D).

In practice, it is common for individual cells to yield more fragments than the model's maximum context window, resulting in more tokens than can be processed in a single forward pass (Supplementary Figure A.9A). We addressed this with two strategies: (1) truncating after the first C tokens (Supplementary Figure A.9B), and (2) randomly sampling C tokens per cell (Supplementary Figure A.9C). We adopted random sampling, reasoning that it better preserves the underlying biological heterogeneity and avoids systematic positional bias. To quantify the effect of context window truncation, we systematically varied the number of tokens sampled per cell and assessed the impact on clustering performance (Supplementary Figure A.9). Remarkably, we observed that Atacformer maintains strong clustering accuracy even with a substantial reduction in context window size (Supplementary Figure A.9D). The performance remains robust until severe truncation, after which clustering metrics rapidly decline (Supplementary Figure A.9E). This result suggests that the model is resilient to moderate input reduction, enabling efficient analysis without significant loss in biological resolution.

We highlight CRAFT's striking performance when considering both speed and clustering ability together. For the PBMC5k dataset, CRAFT not only exhibited the best clustering performance, it also generated those clusters in the smallest amount of time (Figure 5.3E), yielding the highest cells-per-second throughput of all methods. In addition, we performed a preliminary batch-correction analysis across multiple PBMC datasets. Visually, Atacformer embeddings integrated as well as EpiAgent's (Supplementary Figure A.11), despite the latter's much larger parameter count. This suggests that Atacformer's zero-shot capabilities hold promise for rapid, multi-dataset integration without retraining, a key requirement for scalable single-cell analysis. Finally, we omitted ChromFound[63], a newly published scATAC-seq foundation model, from these panels since the code is not open-source and ARI/runtime on these exact PBMC benchmarks aren't reported. In their work, they report ARI=0.48 on a similar PBMC dataset, while maintaining a throughput of 4 cells per second. This would make it the second slowest method benchmarked.

Atacformer learns global regulatory structure in bulk region set data

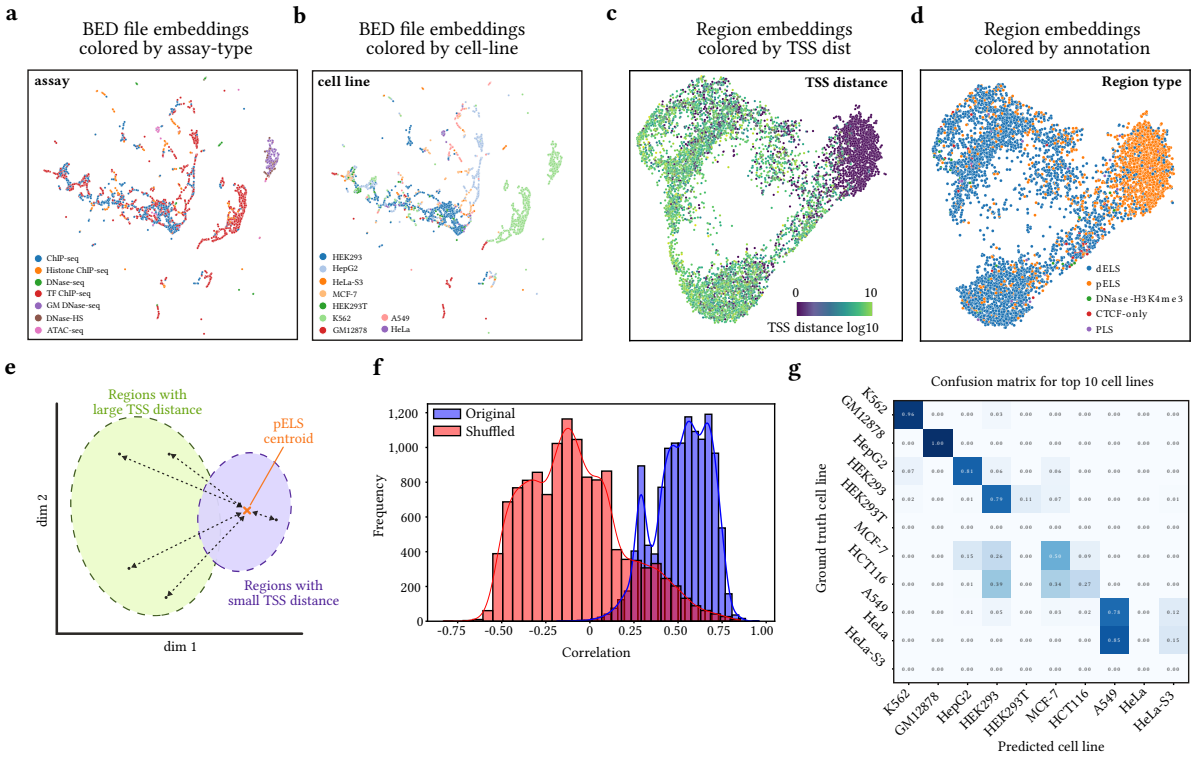


Figure 5.4: Atacformer generalizes to bulk regulatory datasets.

a. UMAP projection of ∼10,000 BED file embeddings from BED files on BEDbase. Points are coloured by assay, revealing discrete clusters of assay types. **b.** The same UMAP projection colored by cell line shows distinct groupings for common cell types, indicating that Atacformer encodes both experimental and biological context without supervision. **c.** UMAP visualization of a set of contextualized region embeddings from an example BED file, colored by TSS distance. **d.** The same UMAP visualization, instead with each region embedding colored by its annotation from the ENCODE SCREEN project. **e.** Schematic of the global TSS distance analysis. The distance of each contextualized region embedding to the pELS centroid is found. In general, regions with a high TSS distance will have a larger distance. **f.** Distribution of Pearson correlation coefficients when correlating a regions annotated TSS-distance to its distance to the pELS centroid. **g.** Confusion matrix for our cell line classifier, limited to the top ten cell lines in the bulk dataset.

Single-cell assays are redefining chromatin biology, yet aggregate profiles from bulk ATAC-seq, ChIP-seq, and DNase-seq are still far more common, and will likely continue to be produced in the future due to dramatically lower cost, as they provide useful information for homogeneous samples or when cell-to-cell heterogeneity is not important. In recent years the number of BED files uploaded to the Gene Expression Omnibus have increased by more than 10,000 files per year, most of them from bulk experiments of many varieties[93,133]. To help manage and interpret this growing archive, we recently developed BEDbase – a unified platform for aggregating, analyzing, and serving genomic region data. We therefore asked whether Atacformer, trained on single-cell data, could generalize to the more heterogenous and voluminous profiles posted to GEO, and thereby provide practical utility for BEDbase, such as by imputing missing key metadata or enhancing region annotations.

To assess this, we fine-tuned Atacformer on BED files annotated with hg38 on BEDbase. We downloaded and tokenized over 35,000 BED files and continued pretraining atacformer-base using ELECTRA-style token replacement prediction for 10 epochs, yielding `atacformer-bb`. Using `atacformer-bb`, we generated embeddings for BED files from the top 10 cell lines and assays in our bulk data training set. The embeddings clustered by both assay type (Figure 5.4A) and cell line (Figure 5.4B), indicating the model learned general patterns of regulatory biology. As shown in Figure 5.4A, assays such as ChIP-seq, TF ChIP-seq, DNase-seq, and ATAC-seq formed distinct groups, suggesting that the model captured assay-specific signal features. Meanwhile, the (Figure 5.4B) shows that the same embeddings also separated cleanly by biological source, with individual clusters corresponding to cell lines such as HEK293, K562, GM12878, and HeLa.

To investigate whether token-level embeddings also reflect learned biology, we annotated each of the model's 890,704 regions with its distance to the closest transcription start site (TSS) and its detailed region classification according to the ENCODE registry of cCREs (see Methods). We reasoned that the contextualized region embeddings produced by the transformer would enable rich information to be encoded in the embeddings. When visualizing an example BED file with UMAP, we found that contextualized embeddings clustered by TSS distance (Figure 5.4C). To confirm that the model is distinguishing promoters from enhancers, we also colored the contextualized region embeddings by their region annotations (Figure 5.4D). We found that regions clustered broadly into groups of proximal enhancer-like sequences (pELS) and distal enhancer-like sequences (dELS).

We next sought to interrogate whether this trend held broadly for all BED files in the training dataset. For this, we developed a novel assessment score inspired by our previous work on region embedding evaluation[105] called the TSS distribution score. Briefly, the score is computed for each region using three steps. First, we identify the pELS centroid in embedding space. Second, each regions embedding distance to this centroid is computed. Finally, we correlate a each regions TSS-distance to its distance to the pELS centroid. In general, we hypothesized that regions close to the centroid would also have corresponding small TSS distances. Conversely, regions far away from the centroid would have corresponding large TSS distances.

We found that the contextualized embeddings for BED files consistently has TSS distribution scores greater than zero, reinforcing the fact that the contexualized embeddings are capturing learned regulatory biology across cell lines and assay types. As a control, we also computed the TSS distribution score for shuffled labels, which yielded a distribution of scores centered at zero (Figure 5.4F).

Finally, we sought to use Atacformer to help annotate data on BEDbase. Many files in the database are missing critical metadata – particularly cell line annotations (Supplementary Figure A.12). We therefore hypothesized that Atacformer could be used to impute this missing data and built a proof of concept pipeline to annotate missing cell line information for all BED files. Specifically, we hypothesized that the BED-file embeddings could be used as input to a classifier to predict the missing annotation.

To build the pipeline, we split our dataset into two groups: 1) BED-files with annotated cell lines and 2) BED-files with missing cell line annotations, but they can be inferred from the file description (Supplementary Figure A.13; see Methods). We use the BED-files with annotated cell lines to train an XGBoost decision tree model[134], and then subsequently use the trained model to predict the cell lines for the BED-files with missing annotations, using the inferred annotations as ground truth labels for evaluation. The decision tree model was effective at annotating the missing cell lines. The model achieved an F1 score of 0.85 and an accuracy of 86% across over 275 cell lines (Figure 5.4G). Taken together, these results highlight how Atacformer is able to generate powerful embedding representations of bulk genomic interval data.

Direct raw-fragment processing with atacformer accelerates scATAC analysis

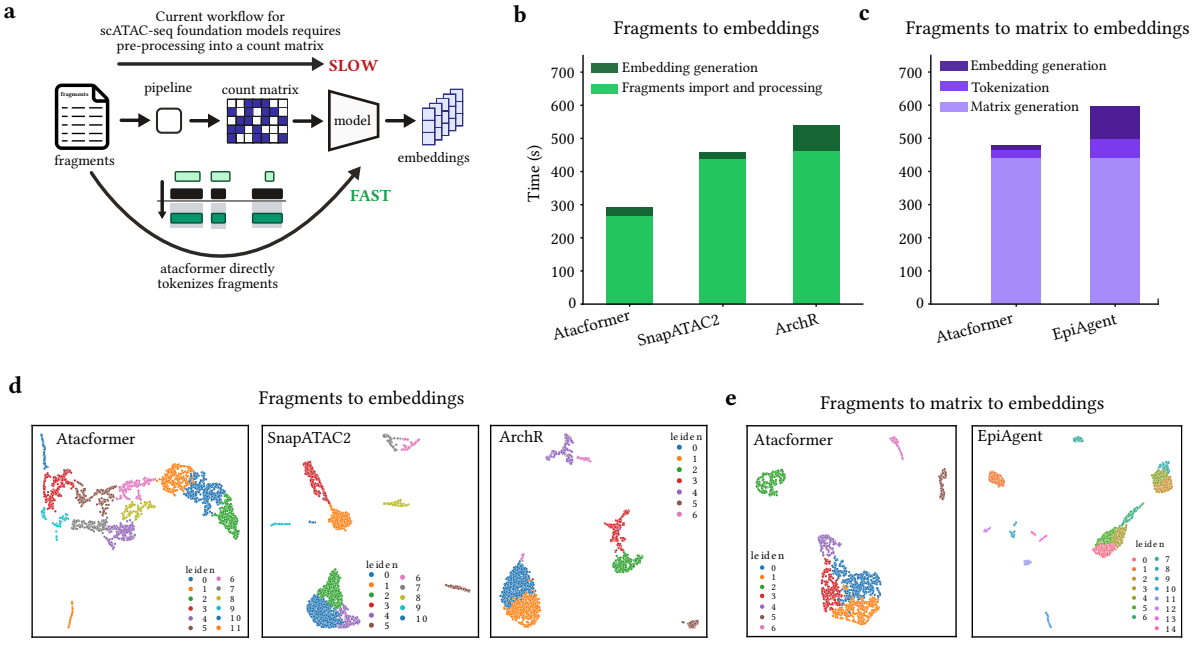


Figure 5.5: Atacformer is the only method that operates on sequence fragments directly.

a. Schematic showing the difference between Atacformers ability to generate embeddings for fragments files and other current scATAC-seq foundation models. **b.** Speed comparison of processing times for Atacformer with SnapATAC2 and ArchR when processing direct fragments files. Fragments file processing is grouped into two main steps: 1) fragments file importing and 2) cell embedding generation. **c.** Speed comparisons between Atacformer and EpiAgent count matrix processing. Count matrix processing is grouped into three main steps: 1) matrix generation, 2) tokenization, and 3) embedding generation. **d.** UMAP visualizations for single-cell

embeddings starting with fragments files for ArchR, Atacformer, and SnapATAC2. **e.** UMAP vsualizations of single-cell embeddings starting with count matrices.

One of the most common starting file-formats for scATAC-seq analysis is the fragments file from 10X genomics. This is a BED-like file that contains aligned reads for single-cells, organized by barcode. Therefore, it is desirable for scATAC-seq workflows to take fragments files as input for cell clustering. However, existing scATAC-seq deep learning models such as EpiAgent, ChromFound, and SCALE require count matrices as input. This necessitates a pre-processing step using a tool like ArchR or SnapATAC2[35,37] to generate the necessary input format when working with fragments files. In contrast, because of its tokenization procedure, Atacformer is uniquely positioned to generate single-cell embeddings immediately from fragments files. By leveraging interval-overlap-based tokenization, Atacformer bypasses the need for tedious preprocessing steps and produces embeddings directly from fragments, enabling more flexible pipelines and zero-shot analyses. This can improve both speed and flexibility in analysis pipelines (Figure 5.5A).

To examine this, we obtained a new 3,000 cell dataset from flash-frozen human healthy brain tissue from 10X genomics. We call this brain3k. We then conducted two separate analyses: 1) fragments to cell embeddings, and 2) count matrices to cell embeddings. For the first analysis, starting from the fragments file, we processed the data in three ways: 1) using Atacformer and its native tokenizers; 2) using SnapATAC2; and 3) using ArchR. We organized this processing into two main steps: 1) fragments import and filtering, and 2) embedding generation. We measured the time it took to conduct each step for each method (see Methods). We then used the resultant embeddings to generate UMAP visualizations. For the second analysis, we took the processed count matrices and fed them to Atacformer and EpiAgent, generating embeddings and corresponding UMAP visualizations, similarly measuring the time it took to conduct each step.

We first used the Atacformer tokenizers to import and tokenize the fragments, filtering by barcodes. We then used these tokenized fragments to generate cell-level embeddings for each cell. In timed comparisons, the end-to-end wall time for Atacformer – comprising fragments import/filtering and embedding generation – was substantially lower than either SnapATAC2 or ArchR (Figure 5.5B). Most of the runtime in the baseline pipelines was spent on I/O-heavy import and matrix construction, whereas Atacformer's Rust-backed interval-overlap tokenization minimizes this overhead by operating directly on the fragments file[91].

Then we next sought to use the processed count matrices as input into Atacformer and compare the performance against EpiAgent, a recently published scATAC-seq foundation model. We found that the majority of the processing time for both methods was spent on pre-processing the count matrix

into a suitable input format. However, once this step was complete, Atacformer generated embeddings substantially faster than EpiAgent. In particular, the Atacformer tokenization and embedding generation steps were approximately 2.5x faster than EpiAgent's equivalent steps (Figure 5.5C).

Using the resulting embeddings from the fragments and count matrices, we visualized cells with UMAP and performed Leiden clustering. For the fragments-based analysis, Atacformer produced coherent, well-separated clusters that were comparable to or better resolved than those obtained from matrices built with SnapATAC2 or ArchR (Figure 5.5D). Notably, Atacformer achieves this in a zero-shot manner – without peak calling or an intermediate peak-by-cell matrix - by leveraging fragment co-occurrence structure within cells. For the count-matrix-based analysis, Atacformer again produced well-separated clusters that were comparable to or better resolved than those from EpiAgent (Figure 5.5E).

Together, these results demonstrate that Atacformer enables practical, faster, and more flexible scATAC-seq workflows: embeddings can be generated directly from fragments, eliminating prerequisite matrix construction and supporting downstream analyses (e.g., clustering) with minimal preprocessing.

Contextualized region embeddings from scATAC-seq data infers cryptic TSSs

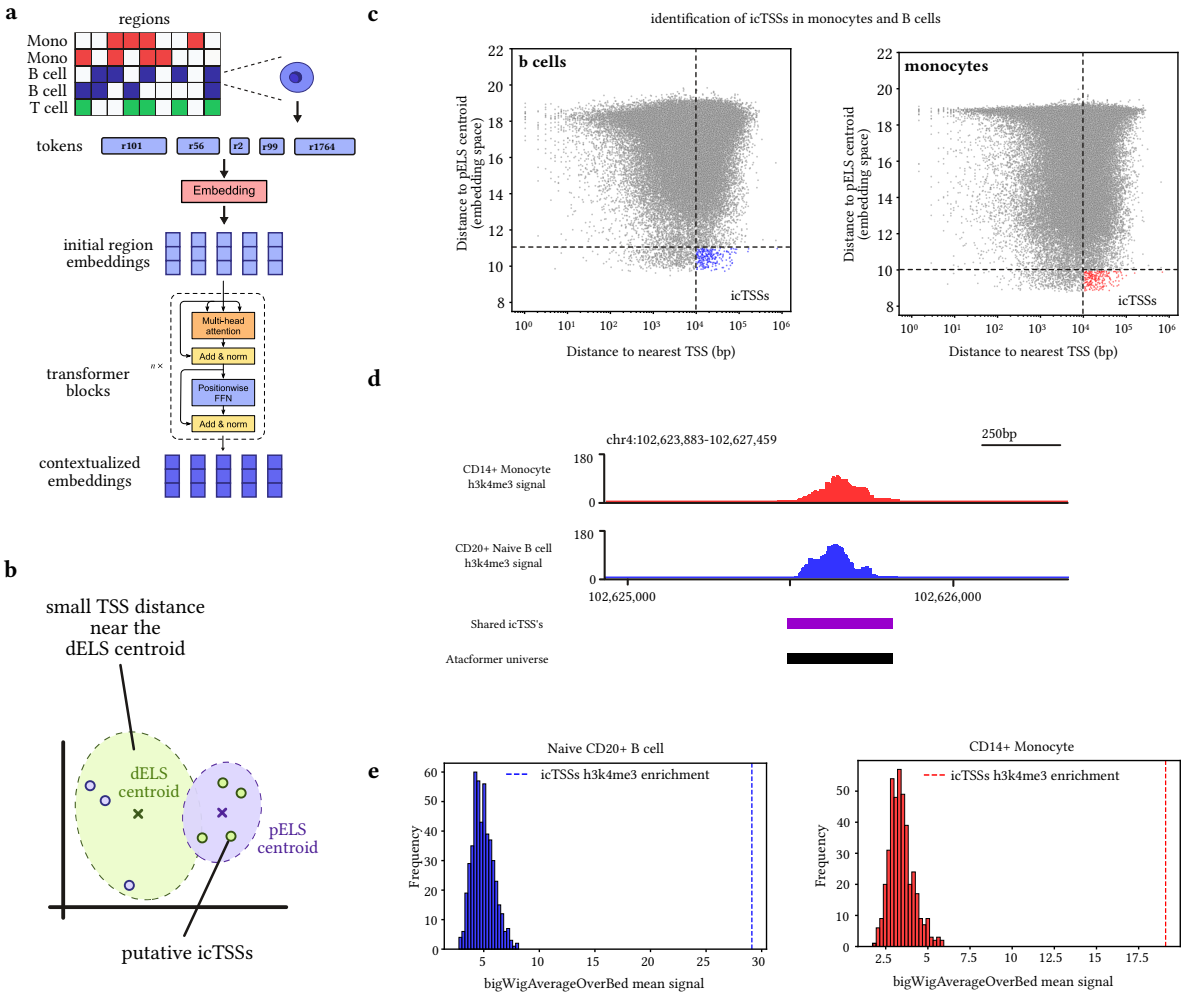


Figure 5.6: Atacformer uncovers weak promoters using scATAC-seq data alone.

a. Schematic of generating contextualized region embeddings from a single cell. **b.** Schematic of the canonical contextualized latent space for co-accessible regions in a single-cell. In general, distal enhancer like sequences cluster together while proximal enhancer like sequences cluster together. **c.** Plot of annotated TSS distance versus pELS centroid distance. We highlight paradoxical regions which are both annotated as very far from the TSS and found very close to the pELS centroid. **d.** Example of an icTSS with strong H3K4me3 signal despite no annotated promoter nearby. **e.** Histograms of H3K4me3 enrichment in a randomyly generated background distribution compared to the icTSSs in both CD14+ monocytes and CD20+ Naive B cells.

A primary strength of the Atacformer architecture is its ability to generate token-level embeddings for individual cis-regulatory elements (cCREs). Unlike other foundation models that produce cell-level representations, Atacformer enables direct investigation of relationships between discrete, well-annotated, genomic regions within a single cell. Building on our initial exploration with bulk ATAC-seq data (Figure 5.4C), we applied this approach to single-cell data, capturing contextualized embeddings for each region prior to aggregation into cell-level representations (Figure 5.6A).

As with bulk data, single-cell region embeddings were broadly structured by annotation, clustering according to TSS distance and their classification as proximal (pELS) or distal (dELS) enhancer-like sequences (Figure 5.6B). Within this structure, however, we identified a discordant subset of regions. These elements were annotated as highly distal from any TSS, yet their embeddings clustered tightly with the pELS centroid – a location dominated by promoter-proximal regions. We hypothesized that this discrepancy reflects functionally important sites, such as weak promoter regions, which we term inferred cryptic TSSs (icTSSs; pronounced "iced-teas").

Cryptic Transcription Start Sites (TSSs) are genomic locations that initiate transcription but are not the primary annotated start sites for known genes. Often found in intronic or intergenic regions, they can be activated in specific cellular contexts or by genetic perturbations, producing truncated proteins or novel non-coding RNAs that may alter cellular function[135].

To investigate this, we subsampled 10,000 single-cells from the Luecken2021 dataset for CD14+ monocytes and naïve CD20+ B cells. For each cell, contextualized co-accessible region embeddings were generated, and each region's distance to the nearest TSS was plotted against its distance to the pELS centroid. Both cell types exhibited a modest but positive correlation between these metrics (Spearman's $\rho = 0.30$ and 0.29, respectively). We then defined icTSSs as regions annotated as highly distal from a TSS yet paradoxicaly embedded very close to the pELS centroid (Figure 5.6C). To validate that icTSSs represent latent promoter regions, we obtained H3K4me3 ChIP-seq data for both cell types from the ENCODE consortium[136]. Briefly, H3K4me3 is a histone modification in which three methyl groups are added to the fourth lysine of histone H3. It is a canonical promoter mark, strongly enriched at transcription start sites where it stabilizes the transcriptional machinery. Thus, enrichment of H3K4me3 provides strong evidence that a region harbors promoter activity.

We found that icTSSs were strongly enriched for H3K4me3 (Figure 5.6D; Supplementary Figure A.14). To assess statistical significance, we constructed null distributions by repeatedly sampling random region sets from the Atacformer vocabulary and comparing their signal overlap with H3K4me3 (see Methods). Against this background, monocyte-specific icTSSs showed a 6.33-fold enrichment for monocyte H3K4me3 peaks, while B-cell icTSSs showed 6-fold enrichment for B-cell peaks (empirical $p < 0.001$ in both cases) (Figure 5.6E).

Together, these results provide strong evidence that Atacformer's contextualized embeddings can identify bona fide weak promoters directly from ATAC-seq data, and that this signal is cell-type specific.

Discussion

Atacformer introduces a general-purpose transformer-based foundation model for chromatin accessibility data, demonstrating strong performance across a diverse set of tasks, including cell-type clustering, fragment file processing, bulk ATAC-seq embedding, and multimodal integration with RNA-seq. Unlike prior models, Atacformer explicitly tokenizes genomic intervals as discrete units and discards reliance on positional encodings, instead encouraging the model to learn contextual biological relationships directly from data. This aspect of Atacformer directly builds on our previous work with learning genomic region embeddings[103,104].

One of Atacformer's most impactful contributions is its ability to operate directly on raw fragment files, bypassing the need for intermediate matrix generation. This greatly reduces computational overhead and processing time, making it especially valuable for large-scale or time-sensitive analyses. Compared to existing scATAC pipelines like SnapATAC2 and ArchR, Atacformer achieves comparable biological fidelity while accelerating analysis by 80% in our benchmarks. This positions Atacformer not just as a new method, but as a fundamentally streamlined alternative to conventional workflows.

Our results show that Atacformer's pre-training strategy, based on ELECTRA-style replaced token detection, enables generalization across both single-cell and bulk data. Fine-tuning on BEDbase bulk datasets reveals that Atacformer embeddings encode biological information such as assay type and cell line identity even in the absence of labels. Token-level embeddings are organized by promoter-enhancer distance without ever having seen explicit TSS annotations during training, underscoring the model's ability to infer global regulatory structure from local context alone.

By integrating Atacformer with Geneformer, we further demonstrate Atacformer's extensibility to multimodal contexts. The CRAFT framework highlights that chromatin-accessibility embeddings can align with transcriptomic signals in a shared latent space, enabling cross-modal retrieval and RNA imputation from ATAC alone. This not only expands Atacformer's applicability to joint profiling datasets, but also opens the door for future extensions such as natural language and epigenome integration or DNA methylation-RNA alignment using similar dual-encoder strategies.

While Atacformer achieves strong results with fewer parameters than some contemporary models, several limitations remain. First, the lack of positional encoding–while intentional–may hinder tasks requiring spatial resolution, such as enhancer-promoter linking across large genomic distances. Second, our approach to fragment tokenization is sensitive to the predefined vocabulary and resolution, which could affect generalization across genome builds or non-human species. Finally, our

evaluations–particularly in multimodal alignment–were limited to well-annotated datasets; broader assessments in low-quality or noisy settings are needed.

Looking ahead, several avenues for extending Atacformer's capabilities are promising. These include:

- Longer context windows or streaming architectures for encoding ultra-complex single-cell profiles.
- Generative pre-training approaches (e.g. diffusion or masked span prediction) to enable more flexible inference tasks.
- Application to clinical and diagnostic datasets, especially in cancer, where chromatin structure is often perturbed.

In conclusion, Atacformer serves as both a performant model and a software framework for ATAC-seq analysis. Its general-purpose embeddings, fragment-level input pipeline, and compatibility with other models make it a powerful tool for epigenomic research. By bridging bulk and single-cell assays, integrating modalities, and enabling fast and interpretable analysis, Atacformer contributes to the growing ecosystem of foundation models in biology and offers a blueprint for future advances in genomic machine learning.

Chapter 6: Conclusions and future work

Overview and Summary of Contributions

The goal of this work was to develop a unified framework for analyzing genomic interval data – particularly single-cell ATAC-seq (scATAC-seq) – through modern machine learning and transfer learning. This was achieved through three key contributions that together form a scalable and interpretable foundation and framework for chromatin accessibility analysis through democratized deep learning models. At its core, this dissertation conceptualizes genomic regions as discrete, linguistic tokens, adapting natural language processing (NLP) concepts and modeling strategies to epigenomics. This, in turn, develops infrastructure and subsequent foundation models that learn generalizable regulatory representations transferable across datasets and biological contexts.

First, we developed high-performance infrastructure for data preprocessing and standardization called the `gtars` toolkit. Written in Rust with Python bindings, gtars provides efficient handling of large-scale genomic interval data and utilities for two key tasks: first, defining region universes from large collections of peak calls, and second, tokenizing individual datasets against these universes. This infrastructure facilitates consistent data representation by ensuring that all datasets share a common vocabulary of genomic regions, which is critical for enabling transfer learning in downstream models. Furthermore, gtars' performance and usability lower the barrier to entry for researchers seeking to apply machine learning to chromatin accessibility data, as well as ensuring that these tools fit into the broader machine learning ecosystem.

Second, we introduced scEmbed, a Word2Vec-inspired model demonstrating that pre-trained region embeddings can be reused across datasets. Trained on large chromatin accessibility corpora, scEmbed learns fixed embeddings for genomic regions that encode shared regulatory context. These embeddings enable rapid analysis and transferability across studies without retraining, establishing the feasibility and importance of stable, shared token vocabularies in genomic modeling. However, scEmbed's static embeddings revealed limitations in small, static representations that failed to generalize to cell-state-specific regulatory dynamics, motivating the need for larger, more flexible models capable of contextual understanding.

Finally, to address these limitations, we developed Atacformer, the central contribution that realizes the core vision of this work: a transformer-based foundation model that generates contextualized region embeddings by incorporating cell-level chromatin accessibility context. Atacformer captures complex regulatory dependencies and generalizes across biological conditions, performing well

in zero-shot settings for clustering, annotation, and batch correction. In parallel, we introduced the CRAFT model, a dual-encoder architecture linking Atacformer with Geneformer to jointly model chromatin accessibility and gene expression, illustrating the potential for unified multiomic embeddings that was envisioned from the beginning of this research program. We finally explored the biological insights enabled by these models, demonstrating that they capture meaningful regulatory relationships, identify cryptic promoters, and reflect functional relatedness among genomic regions, highlighting the models interpretability and biological relevance.

Together, these components form a continuous pipeline from disparate, raw genomic intervals to contextual representations, which is anchored in shared vocabulary, interpretability, and scalability.

Technical limitations and challenges

Through the development of gtars, scEmbed, and Atacformer, we have gained critical insights into the capabilities and constraints of our current approaches. The challenges encountered throughout this research have revealed important boundaries of existing methods and highlighted opportunities for future innovation. Key technical limitations identified through this work include:

1. **Data imbalance and diversity:** Training data remain biased toward well-characterized tissues, cell types, and experimental conditions, limiting generalization to rare or novel biological contexts.
2. **Tokenization strategies:** Current approaches employ relatively simple, fixed-region tokenization that makes critical assumptions about regulatory element boundaries and completely omits order and spacing information between regions.
3. **Interpretability:** While models demonstrate strong performance on cell-level tasks, interpretation of contextualized region embeddings remains underdeveloped, limiting our ability to extract mechanistic insights about regulatory grammar and interactions.
4. **Computational scalability:** Computational demands present dual challenges—scaling down to enable accessible inference for resource-constrained researchers, and scaling up to extend context length for processing comprehensive genomic datasets like bulk ATAC-seq and multiomic experiments.

We expand on these limitations in the next section, outlining specific next steps to address each challenge and advance the state of the art in computational epigenomics.

Future Directions: Improving generalization, efficiency, and interpretability of regulatory genomics models

Future aim 1: Scaling the training Atlas

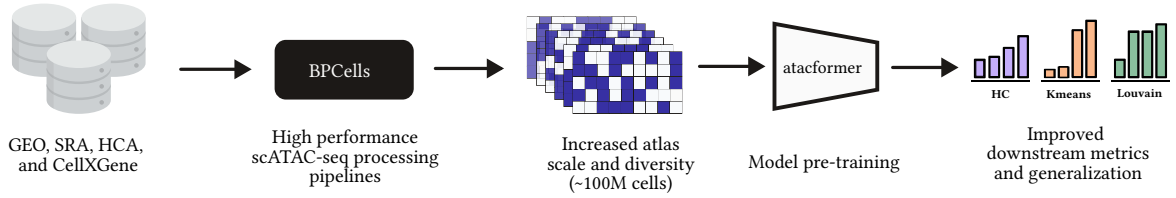


Figure 6.1: Overview of the expanded training atlas and its components.

Motivation

Atacformer was trained on a substantial corpus of single-cell ATAC-seq data (~1.2 million cells); however, this training set remains biased toward well-characterized tissues, cell types, and experimental conditions. Specifically, the dataset is currently dominated by brain, immune, and tonsil cells (Figure 5.1B). This bias is reflected in model performance, with reduced performance when processing datasets originating from biological contexts outside the models training distribution (kidney, liver, rare cell types; Supplementary Figure A.15). In contrast, foundation models in natural language processing (NLP) have demonstrated that massive, diverse training corpora are critical for achieving strong generalization and reducing bias[45]. To achieve similar benefits in regulatory genomics, scaling the training atlas to encompass tens or hundreds of millions of cells spanning all major human tissues, developmental stages, and disease states would be a logical next step. Indeed, foundation models for transcriptomics are already being trained on datasets of this scale with an updated Geneformer model being trained on over 100 million cells[137] and the release of Tahoe-x1, a new transcriptomics foundation model trained on 100 million cells[138]. Expanding the training corpus for chromatin accessibility models like Atacformer could improve generalization and reduce bias in downstream analyses.

Proposed approach

Several recent technological developments now make it feasible to scale training corpora to 100 million cells or more. First, leveraging the computational scalability of tools like BPCells[139], which offer improved memory efficiency for processing large-scale datasets compared to our initial preprocessing pipeline (SnapATAC2[37]), would facilitate such expansion. Second, Oxbow[140] — a unification layer built on Apache Arrow with Rust-based I/O—enables efficient handling of next-generation sequencing data as in-memory or larger-than-memory data frames across Python and R, streamlining cross-platform

data integration. Together, these advances address the storage, I/O, and processing bottlenecks that previously limited large-scale data curation.

Building on this infrastructure, one could systematically collect single-cell ATAC-seq datasets from public repositories including GEO, SRA, the Human Cell Atlas, and CellXGene, targeting a training corpus exceeding 100 million cells that spans all major human tissues, developmental stages, and disease states. Such an effort could prioritize underrepresented biological systems—diverse neuron subtypes, rare immune populations, and organoid models—to mitigate the tissue and cell-type biases observed in the current Atacformer training set. A robust quality control pipeline would harmonize data from heterogeneous sources, accounting for differences in sequencing depth, peak calling strategies, and experimental protocols. Stratified sampling strategies would ensure balanced representation across tissues and cell types during model training, reducing the risk of overfitting to well-studied biological contexts.

Expected impact and evaluation

Scaling the training atlas to 100 million cells would be expected to yield models with substantially improved generalization across diverse biological contexts. Such models could demonstrate better zero-shot performance on previously unseen tissues and cell types, reduced bias in cell-type annotation and clustering, and more robust representations for rare cell populations currently underrepresented in existing training data. These improvements would establish a stronger foundation for clinical applications that require accurate modeling of diverse disease contexts and developmental stages.

To quantitatively assess these improvements, one could evaluate retrained models using the same cell-clustering benchmarks established in the original Atacformer work. Specifically, measuring clustering quality metrics (silhouette scores, adjusted Rand index) across held-out datasets spanning diverse tissues would allow comparison of performance between models trained on the original corpus versus an expanded 100-million-cell atlas. Particular attention could be paid to performance gains on underrepresented tissues (kidney, liver, rare immune populations) where the current model shows reduced accuracy. Additionally, assessing the model's ability to generalize to completely novel datasets not represented in either training corpus would provide a rigorous test of improved generalization. Finally, the curated training corpus could be released as a community resource to support reproducibility and enable other researchers to build upon this foundation.

Future aim 2: Improved tokenization strategies

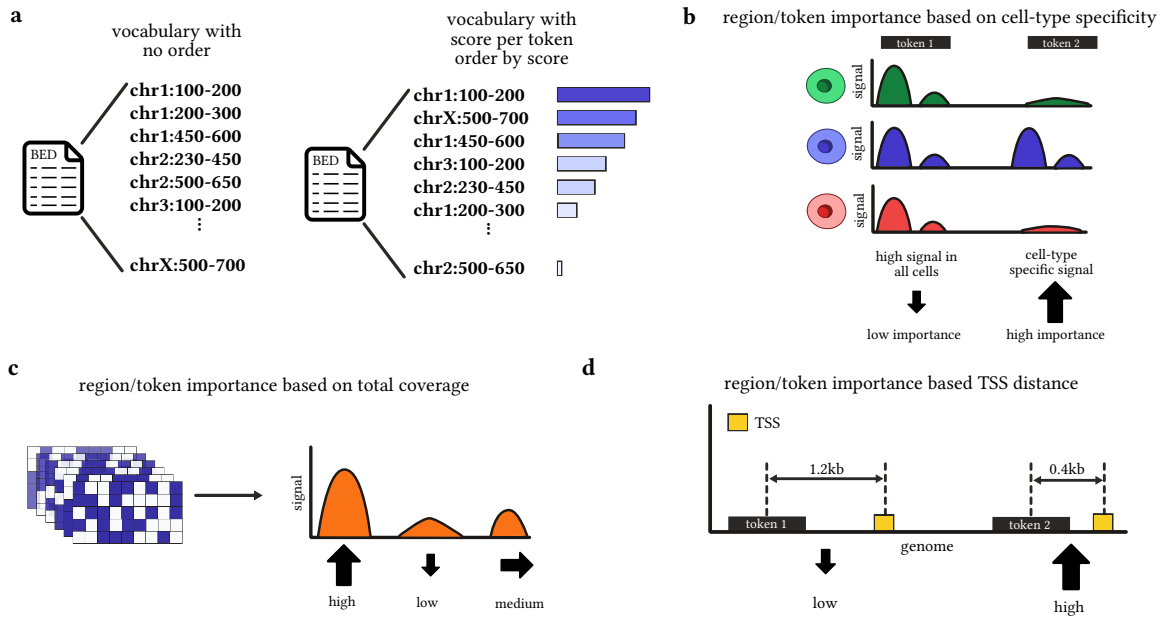


Figure 6.2: Overview of the improved tokenization strategies and their components.

a. Schematic of a model vocabulary/universe that is ordered by an importance score. **b.** Overview of determining importance via cell-type specificity. **c.** Overview of determining importance by total corpus coverage. **d.** Overview of determining importance by TSS distance.

Motivation

The current tokenization strategies for our epigenomic foundation models face two interconnected challenges. First, they indiscriminately include all accessible regions detected in a single cell. Often, several thousand cCREs are co-accessible in any given cell. This results in very large context windows required to capture the full regulatory environment (Supplementary Figure A.7). This approach is computationally expensive and inefficient, as many of these regions are housekeeping elements or low-information peaks analogous to stop words in natural language processing. Second, current tokenization treats all regions equally and imposes no inherent order, representing each cell as an unordered "bag of peaks" that discards potentially important spatial relationships between regulatory elements.

These two problems could be addressed simultaneously through importance scoring at tokenization time, an approach successfully employed by other similar models (EpiAgent and Geneformer)[58,62,137]. By assigning an importance score to each region in the vocabulary, one could retain only the top K most critical regions, effectively reducing context window size while focusing model attention on functionally relevant elements. Additionally, sorting regions by importance establishes a natural ordering that prioritizes regulatory elements most likely to influence cellular state. This dual benefit

— reduced context and imposed order — addresses both computational efficiency and biological interpretability (Figure 6.2A).

Finally and importantly, the choice of tokenization strategy has cascading effects on pretraining objectives. In developing Atacformer, we had to pivot from masked language modeling (MLM)[42] to ELECTRA-style[129] discriminative pretraining because MLM becomes computationally prohibitive for large vocabularies and context windows. Additionally, standard MLM assumes a fixed token order for masking, which is incompatible with our unordered bag-of-peaks representation. ELECTRA sidesteps both issues by using a discriminative objective that does not require predicting exact tokens from a large vocabulary. However, with improved tokenization that reduces context size and establishes inherent order, one could revisit MLM and explore alternative pretraining objectives that were previously infeasible, potentially unlocking better learned representations.

Proposed approach: Importance scoring

The first component of improved tokenization involves developing scoring functions to rank genomic regions by regulatory importance. Several complementary strategies could be employed:

1. **Cell-type specificity metrics** would identify regions with high variance across cell types, as these likely encode cell-type-defining regulatory programs rather than housekeeping functions (Figure 6.2B).

2. **Signal overlap quality** could assess whether a region shows weak or strong accessibility signal, prioritizing high-confidence peaks with robust coverage (Figure 6.2C).

3. **Distance to transcription start sites** could prioritize regions proximal to gene promoters, as regulatory elements closer to TSSs often have stronger and more direct effects on gene expression. This could be implemented by assigning higher importance scores to regions within a certain distance threshold of annotated TSSs, with scores decaying as a function of genomic distance (Figure 6.2D).

These metrics could be combined into a composite importance score, either through manual weighting or by training a machine learning classifier to predict regulatory impact based on downstream gene expression effects.

Proposed approach: Ordering strategies

The second component involves establishing an explicit ordering over regions, moving beyond the current "bag-of-peaks" representation. One straightforward approach would be to order regions by their importance scores, creating a ranked sequence where the most critical regulatory elements appear first in the token sequence. Alternatively, one could order regions by genomic distance,

preserving spatial relationships between nearby regulatory elements that may form functional modules. This could be implemented through learned positional embeddings that encode genomic coordinates, allowing the model to capture distance-dependent interactions. A more sophisticated approach, inspired by ChromFound's complex genomic positional embeddings, would combine learnable chromosome embeddings in addition to other spatial features like distance along the genome. Balancing these ordering strategies with computational tractability remains a key challenge, as distance-aware attention mechanisms can introduce additional overhead.

Expected impact

Improved tokenization strategies would be expected to yield two major benefits. First, by incorporating spatial relationships and regulatory importance into token representations, such approaches could improve performance across downstream tasks including cell-type clustering, region annotation, and cellular state prediction. Ordering regions by importance or genomic position would allow attention mechanisms to capture biologically meaningful relationships that are currently obscured by the bag-of-peaks representation. This richer contextualization should translate directly to more accurate embeddings and better generalization across diverse biological contexts.

Second, and equally important, importance scoring would enable sustained performance with substantially reduced context window sizes. By retaining only the top K most critical regions per cell, models could maintain or even improve accuracy while processing far fewer tokens. This would yield a more efficient and practically usable model—reducing training time, lowering inference costs, and enabling deployment on less powerful hardware. The ability to achieve comparable or superior performance with smaller context windows would democratize access to these foundation models and accelerate iteration during methods development.

Future aim 3: Token-level interpretability and fine-tuning

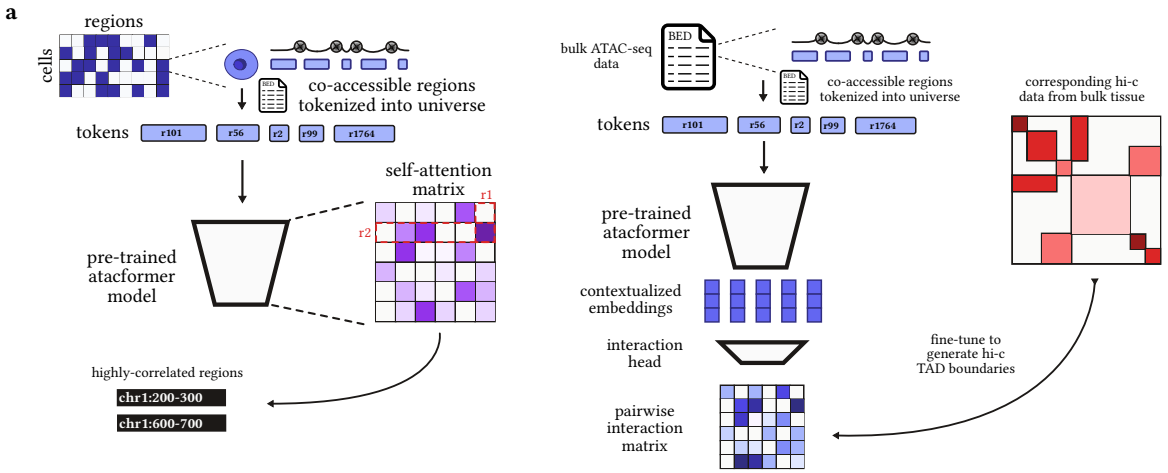


Figure 6.3: Overview of the further directions for token-level interpretability and fine-tuning.

a. We can investigate region-pairs through the attention matrix. Strong attention scores between two regions might indicate biological relationships. **b.** Using bulk ATAC-seq data, we can fine-tune foundation models to predict chromatin contacts between regions.

Motivation

While Atacformer demonstrates strong performance on cell-level tasks such as clustering and annotation, interpretation and analysis of contextualized region embeddings remains underdeveloped. Our initial exploration of token-level embeddings revealed promising signals. For instance, we identified putative latent transcription start sites within weak promoter regions by examining how these regions cluster in embedding space to reveal paradoxic patterns: regions annotated as being far from transcription start sites while clustering with known TSSs (Figure 5.6). However, this only scratches the surface of what token-level analysis could reveal about regulatory grammar and chromatin organization. Two complementary directions could substantially advance understanding of these models: 1) attention-based relationship discovery and 2) token-level fine-tuning for region annotation tasks.

Proposed approach: Attention matrix analysis

Transformer attention matrices encode pairwise relationships between tokens, and attention weight analysis has proven valuable in natural language processing for uncovering syntactic dependencies and semantic relationships[141,142]. In genomics, attention patterns could reveal functional regulatory connections that are not apparent from sequence features alone. Systematic extraction and analysis of attention weights between region pairs across large cohorts of cells could be performed. Consistently high-attention pairs across diverse cell types would suggest stable regulatory interactions, such as housekeeping promoter-enhancer links, while cell-type-specific attention patterns could identify dynamic, context-dependent regulatory wiring. These attention-derived relationships could be used to predict promoter-enhancer pairs, infer transcription factor co-binding networks, and identify compensatory or redundant regulatory elements (Figure 6.3A). Validation against orthogonal data sources like Hi-C contact maps, ChIA-PET loops, and expression quantitative trait loci (eQTLs) would assess whether attention weights capture biologically meaningful three-dimensional chromatin interactions and regulatory logic.

Proposed approach: Token-level fine-tuning

In our current work, we explored fine-tuning strategies exclusively at the cell level, training models to predict cellular phenotypes or cluster assignments. However, token-level fine-tuning—analogous to named entity recognition (NER) in natural language processing—remains unexplored and could unlock new capabilities[42,143]. One compelling application would involve training models to predict chromatin contact pairs using small Hi-C datasets as supervision. This would create a generative model capable of predicting topologically associating domains (TADs) and long-range interactions

directly from scATAC-seq data, effectively learning to infer three-dimensional genome organization from accessibility patterns alone. Another promising direction involves region annotation tasks, where models could learn to classify individual regions (e.g., as active enhancers, silencers, or insulators) based on the broader regulatory context of the cell. This context-aware annotation would differ fundamentally from static chromatin state annotations like ChromHMM, as it would account for the dynamic regulatory environment captured by the full set of accessible regions in each cell (Figure 6.3B). Both applications would leverage the rich contextual information encoded in Atacformer's learned representations, extending the model's utility beyond cell-level predictions to region-level biological insights.

Expected impact

These advances would yield two major benefits. First, token-level analysis would provide mechanistic insights into regulatory grammar that are currently inaccessible—revealing novel promoter-enhancer wiring, identifying context-dependent regulatory switches, and generating testable hypotheses about three-dimensional genome organization. Second, fine-tuned models for contact prediction and region annotation could be released as community resources, enabling researchers to leverage foundation model representations for specific biological questions without requiring computational expertise or large-scale datasets. By extending capabilities from cell-level clustering to region-level insights, these directions would democratize sophisticated regulatory genomics analysis.

Future aim 4: Context window optimization — exploring the extremes

Motivation

Throughout Atacformer development, we employed a fixed context window of 8,192 tokens, chosen to accommodate the majority of single cells while remaining computationally tractable. However, this one-size-fits-all approach leaves substantial room for optimization in *both* directions. Preliminary analysis revealed that while many cells benefit from large context windows, a significant proportion may be adequately represented with far fewer tokens (Supplementary Figure A.9), particularly when coupled with improved tokenization strategies that prioritize the most informative regions. Conversely, our ability to analyze bulk ATAC-seq data remains severely limited, as bulk datasets often contain 50,000–200,000 accessible regions per sample, far exceeding our current capacity. Exploring both extremes—smaller context windows for efficiency and larger windows for comprehensive genomic coverage—would unlock new capabilities and broaden the applicability of foundation models in regulatory genomics.

Proposed approach: Scaling down with efficient architectures

The first direction involves determining how small context windows can be while maintaining performance. This could be achieved by combining tokenization strategies with importance scoring to retain only the most informative regions. Systematic experimentation with reduced context sizes (e.g., 512, 1,024, 2,048 tokens) could identify the optimal trade-off between biological coverage and computational efficiency. This could be paired with advanced self-attention implementations such as state space models (Mamba, S4)[64,85] that offer subquadratic scaling with sequence length, dramatically reducing training and inference time. Additional efficiency gains could come from model compression techniques like quantization to create lightweight variants suitable for CPU-based inference. The goal would be to produce models analogous to the sentence-transformer family in natural language processing[130]: fast, efficient, and deployable in local environments without specialized hardware, enabling researchers with limited computational resources to leverage foundation model representations.

Proposed approach: Scaling up for bulk data integration

The second direction involves extending context windows to accommodate comprehensive bulk ATAC-seq datasets, which contain orders of magnitude more accessible regions than single-cell data. Our initial bulk data analysis was constrained to BED files with only a few thousand regions to fit within the 8,192-token limit, severely restricting the diversity and complexity of bulk datasets that could be analyzed (Figure 5.4). To address this, one could explore cutting-edge long-context architectures such as sparse attention mechanisms (Longformer[83], BigBird[84]), memory-efficient implementations (Flash Attention[85]), or hybrid architectures inspired by ChromFound[63] that combine local and global attention patterns. Target context lengths of 50,000–100,000 tokens would enable processing of full bulk ATAC-seq samples, allowing models to capture long-range regulatory interactions and global chromatin architecture. While these approaches may require high-performance computing environments, the ability to integrate bulk data alongside single-cell data would create a unified framework for analyzing chromatin accessibility across experimental modalities. This capability would be particularly valuable for platforms like BEDbase, which curates large collections of bulk genomic datasets.

Expected impact

Optimizing context windows in both directions would produce complementary benefits. Smaller, efficient models would democratize access to foundation model capabilities, enabling local deployment, faster iteration during analysis, and broader adoption in resource-limited settings. This would parallel the success of sentence-transformers in making NLP embeddings accessible to any researcher with a laptop. Conversely, extended context models would unlock comprehensive analysis

of bulk ATAC-seq data, enabling integration of diverse experimental modalities and providing insights into genome-wide regulatory architecture that are currently inaccessible. Together, these advances would create a flexible ecosystem of models tailored to different computational environments and biological questions—from lightweight tools for rapid exploration to heavyweight models for in-depth analysis of complex regulatory landscapes.

Broader Impact and Closing Remarks

Modern NLP and computer vision have matured immensely through standardized tokenization, evaluation, and model-sharing ecosystems. Genomics has lacked such a unifying framework. Fragmented tools, inconsistent formats, and limited reproducibility have hindered progress particularly for machine learning applications.

This dissertation establishes a foundation for such a framework. By defining genomic regions as transferable tokens, it provides the conceptual and infrastructural basis for foundation models in regulatory genomics. The progression from gtars (infrastructure) to scEmbed (embedding prototype) to Atacformer (contextual foundation model) parallels the evolution from bag-of-words to transformers in NLP—transforming static, handcrafted features into dynamic, learnable representations of biological meaning.

Ultimately, this work demonstrates that gene regulation, like language, is compositional and contextual. Treating chromatin accessibility as a structured language enables not only better models but also deeper biological insight—revealing the syntax of regulation and setting the stage for a new generation of data-driven discovery in epigenomics.

References

1. Alberts, B. *et al. Molecular Biology of the Cell.* (Garland Science, 2002).
2. Levine, M. & Tjian, R. Transcription regulation and animal diversity. *Nature* **424,** 147–151 (2003).
3. Ong, C.-T. & Corces, V. G. Enhancer function: new insights into the regulation of tissue-specific gene expression. *Nature Reviews Genetics* **12,** 283–293 (2011).
4. Lander, E. S. *et al.* Initial sequencing and analysis of the human genome. *Nature* **409,** 860–921 (2001).
5. Farh, K. K.-H. *et al.* Genetic and epigenetic fine mapping of causal autoimmune disease variants. *Nature* **518,** 337–343 (2015).
6. French, J. D. & Edwards, S. L. The Role of Noncoding Variants in Heritable Disease. *Trends in Genetics* **36,** 880–891 (2020).
7. Jones, P. A. Functions of DNA methylation: islands, start sites, gene bodies and beyond. *Nature Reviews Genetics* **13,** 484–492 (2012).
8. Smith, Z. D. & Meissner, A. DNA methylation: roles in mammalian development. *Nature Reviews Genetics* **14,** 204–220 (2013).
9. Frommer, M. *et al.* A genomic sequencing protocol that yields a positive display of 5-methylcytosine residues in individual DNA strands. *Proceedings of the National Academy of Sciences of the United States of America* **89,** 1827–1831 (1992).
10. Li, Y. & Tollefsbol, T. O. DNA Methylation Detection: Bisulfite Genomic Sequencing Analysis. *Epigenetics Protocols* 11–21 (2011) doi:10.1007/978-1-61779-316-5_2.
11. Barski, A. *et al.* High-Resolution Profiling of Histone Methylations in the Human Genome. *Cell* **129,** 823–837 (2007).
12. Johnson, D. S., Mortazavi, A., Myers, R. M. & Wold, B. Genome-Wide Mapping of in Vivo Protein-DNA Interactions. *Science* **316,** 1497–1502 (2007).
13. Robertson, G. *et al.* Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nature Methods* **4,** 651–657 (2007).
14. Kaya-Okur, H. S. *et al.* CUT&Tag for efficient epigenomic profiling of small samples and single cells. *Nature Communications* **10,** 1930 (2019).
15. Thurman, R. E. *et al.* The accessible chromatin landscape of the human genome. *Nature* **489,** 75–82 (2012).
16. Boyle, A. P. *et al.* High-Resolution Mapping and~Characterization of Open Chromatin across the Genome. *Cell* **132,** 311–322 (2008).
17. Giresi, P. G., Kim, J., McDaniell, R. M., Iyer, V. R. & Lieb, J. D. FAIRE (Formaldehyde-Assisted Isolation of Regulatory Elements) isolates active regulatory elements from human chromatin. *Genome Research* **17,** 877–885 (2007).
18. Buenrostro, J. D., Giresi, P. G., Zaba, L. C., Chang, H. Y. & Greenleaf, W. J. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. *Nature Methods* **10,** 1213–1218 (2013).
19. Lieberman-Aiden, E. *et al.* Comprehensive Mapping of Long-Range Interactions Reveals Folding Principles of the Human Genome. *Science* **326,** 289–293 (2009).
20. Dixon, J. R. *et al.* Topological Domains in Mammalian Genomes Identified by Analysis of Chromatin Interactions. *Nature* **485,** 376–380 (2012).
21. Belton, J.-M. *et al.* Hi-C: A comprehensive technique to capture the conformation of genomes. *Methods (San Diego, Calif.)* **58,** 10.1016/j.ymeth.2012.05.001 (2012).
22. Rotem, A. *et al.* Single-cell ChIP-seq reveals cell subpopulations defined by chromatin state. *Nature biotechnology* **33,** 1165–1172 (2015).
23. Smallwood, S. A. *et al.* Single-cell genome-wide bisulfite sequencing for assessing epigenetic heterogeneity. *Nature Methods* **11,** 817–820 (2014).
24. Buenrostro, J. D. *et al.* Single-cell chromatin accessibility reveals principles of regulatory variation. *Nature* **523,** 486–490 (2015).

25. Zhang, Y. *et al.* Model-based Analysis of ChIP-Seq (MACS). *Genome Biology* **9,** R137 (2008).

26. Cusanovich, D. A. *et al.* Multiplex single cell profiling of chromatin accessibility by combinatorial cellular indexing. *Science (New York, N.Y.)* **348,** 910–914 (2015).

27. Schep, A. N., Wu, B., Buenrostro, J. D. & Greenleaf, W. J. chromVAR: inferring transcription-factor-associated accessibility from single-cell epigenomic data. *Nature Methods* **14,** 975–978 (2017).

28. de Boer, C. G. & Regev, A. BROCKMAN: deciphering variance in epigenomic regulators by k-mer factorization. *BMC Bioinformatics* **19,** 253 (2018).

29. Pliner, H. A. *et al.* Cicero Predicts cis-Regulatory DNA Interactions from Single-Cell Chromatin Accessibility Data. *Molecular Cell* **71,** 858–871.e8 (2018).

30. Bravo González-Blas, C. *et al.* cisTopic: cis-regulatory topic modeling on single-cell ATAC-seq data. *Nature Methods* **16,** 397–400 (2019).

31. Fang, R. *et al.* Comprehensive analysis of single cell ATAC-seq data with SnapATAC. *Nature Communications* **12,** 1337 (2021).

32. Ji, Z., Zhou, W. & Ji, H. Single-cell regulome data analysis by SCRAT. *Bioinformatics* **33,** 2930–2932 (2017).

33. Danese, A. *et al.* EpiScanpy: integrated single-cell epigenomic analysis. *Nature Communications* **12,** 5228 (2021).

34. Cusanovich, D. A. *et al.* A Single-Cell Atlas of In~Vivo Mammalian Chromatin Accessibility. *Cell* **174,** 1309–1324.e18 (2018).

35. Granja, J. M. *et al.* ArchR is a scalable software package for integrative single-cell chromatin accessibility analysis. *Nature Genetics* **53,** 403–411 (2021).

36. Stuart, T., Srivastava, A., Madad, S., Lareau, C. A. & Satija, R. Single-cell chromatin state analysis with Signac. *Nature Methods* **18,** 1333–1341 (2021).

37. Zhang, K., Zemke, N. R., Armand, E. J. & Ren, B. A fast, scalable and versatile tool for analysis of single-cell omics data. *Nature Methods* **21,** 217–227 (2024).

38. Gayoso, A. *et al.* A Python library for probabilistic analysis of single-cell omics data. *Nature Biotechnology* **40,** 163–166 (2022).

39. Xiong, L. *et al.* SCALE method for single-cell ATAC-seq analysis via latent feature extraction. *Nature Communications* **10,** 1–10 (2019).

40. Ashuach, T., Reidenbach, D. A., Gayoso, A. & Yosef, N. PeakVI: A deep generative model for single-cell chromatin accessibility analysis. *Cell Reports Methods* **2,** 100182 (2022).

41. Yuan, H. & Kelley, D. R. scBasset: sequence-based modeling of single-cell ATAC-seq using convolutional neural networks. *Nature Methods* **19,** 1088–1096 (2022).

42. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019) doi:10.48550/arXiv.1810.04805.

43. Radford, A., Narasimhan, Salimans, T. & Sutskever, I. Improving Language Understanding by Gnerative Pre-Training. (2018).

44. Radford, A. *et al.* Learning Transferable Visual Models From Natural Language Supervision. (2021) doi:10.48550/arXiv.2103.00020.

45. Brown, T. B. *et al.* Language Models are Few-Shot Learners. (2020) doi:10.48550/arXiv.2005.14165.

46. Raffel, C. *et al.* Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. (2023) doi:10.48550/arXiv.1910.10683.

47. Dosovitskiy, A. *et al.* An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. (2021) doi:10.48550/arXiv.2010.11929.

48. Touvron, H. *et al.* LLaMA: Open and Efficient Foundation Language Models. (2023) doi:10.48550/arXiv.2302.13971.

49. He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. (2015) doi:10.48550/arXiv.1512.03385.

50. Vaswani, A. *et al.* Attention Is All You Need. (2017) doi:10.48550/arXiv.1706.03762.

51. Ji, Y., Zhou, Z., Liu, H. & Davuluri, R. V. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics* **37,** 2112–2120 (2021).

52. Avsec, Ž. *et al.* Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods* **18,** 1196–1203 (2021).

53. Abramson, J. *et al.* Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature* **630,** 493–500 (2024).

54. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589 (2021).

55. Senior, A. W. *et al.* Improved protein structure prediction using potentials from deep learning. *Nature* **577,** 706–710 (2020).

56. Sarkar, A. *et al.* Designing DNA With Tunable Regulatory Activity Using Score-Entropy Discrete Diffusion. *bioRxiv* 2024.05.23.595630 (2025) doi:10.1101/2024.05.23.595630.

57. Cui, H. *et al.* scGPT: toward building a foundation model for single-cell multi-omics using generative AI. *Nature Methods* 1–11 (2024) doi:10.1038/s41592-024-02201-0.

58. Theodoris, C. V. *et al.* Transfer learning enables predictions in network biology. *Nature* **618,** 616–624 (2023).

59. Wu, Y. *et al.* Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. (2016) doi:10.48550/arXiv.1609.08144.

60. Kudo, T. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. (2018) doi:10.48550/arXiv.1804.10959.

61. Kudo, T. & Richardson, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. (2018) doi:10.48550/arXiv.1808.06226.

62. Chen, X. *et al.* EpiAgent: foundation model for single-cell epigenomics. *Nature Methods* 1–12 (2025) doi:10.1038/s41592-025-02822-z.

63. Jiao, Y. *et al.* ChromFound: Towards A Universal Foundation Model for Single-Cell Chromatin Accessibility Data. (2025) doi:10.48550/arXiv.2505.12638.

64. Gu, A. & Dao, T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. (2024) doi:10.48550/arXiv.2312.00752.

65. Mikolov, T., Chen, K., Corrado, G. & Dean, J. Efficient Estimation of Word Representations in Vector Space. (2013) doi:10.48550/arXiv.1301.3781.

66. Morin, F. & Bengio, Y. Hierarchical Probabilistic Neural Network Language Model. in *International Workshop on Artificial Intelligence and Statistics* 246–252 (PMLR, 2005).

67. Gutmann, M. & Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* 297–304 (JMLR Workshop and Conference Proceedings, 2010).

68. Pennington, J., Socher, R. & Manning, C. GloVe: Global Vectors for Word Representation. in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 1532–1543 (Association for Computational Linguistics, Doha, Qatar, 2014). doi:10.3115/v1/D14-1162.

69. Bojanowski, P., Grave, E., Joulin, A. & Mikolov, T. Enriching Word Vectors with Subword Information. *arXiv:1607.04606 [cs]* (2017).

70. Le, Q. & Mikolov, T. Distributed Representations of Sentences and Documents. in *Proceedings of the 31st International Conference on Machine Learning* 1188–1196 (PMLR, 2014).

71. Asgari, E. & Mofrad, M. R. K. Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. *PLoS ONE* **10,** e141287 (2015).

72. Ng, P. dna2vec: Consistent vector representations of variable-length k-mers. *arXiv:1701.06279 [cs, q-bio, stat]* (2017).

73. Du, J. *et al.* Gene2vec: distributed representation of genes based on co-expression. *BMC Genomics* **20,** 82 (2019).

74. Cho, K. *et al.* Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. (2014) doi:10.48550/arXiv.1406.1078.

75. Sutskever, I., Vinyals, O. & Le, Q. V. Sequence to Sequence Learning with Neural Networks. (2014) doi:10.48550/arXiv.1409.3215.

76. Bahdanau, D., Cho, K. & Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. (2016) doi:10.48550/arXiv.1409.0473.

77. Min, X., Zeng, W., Chen, N., Chen, T. & Jiang, R. Chromatin accessibility prediction via convolutional long short-term memory networks with k-mer embedding. *Bioinformatics* **33,** i92–i101 (2017).

78. Wang, H. *et al.* A new LSTM-based gene expression prediction model: L-GEPM. *Journal of Bioinformatics and Computational Biology* **17,** 1950022 (2019).

79. Wang, S., Li, B. Z., Khabsa, M., Fang, H. & Ma, H. Linformer: Self-Attention with Linear Complexity. (2020) doi:10.48550/arXiv.2006.04768.

80. Xiong, Y. *et al.* Nyströmformer: A Nyström-Based Algorithm for Approximating Self-Attention. (2021) doi:10.48550/arXiv.2102.03902.

81. Choromanski, K. *et al.* Rethinking Attention with Performers. (2021) doi:10.48550/arXiv.2009.14794.

82. Child, R., Gray, S., Radford, A. & Sutskever, I. Generating Long Sequences with Sparse Transformers. (2019) doi:10.48550/arXiv.1904.10509.

83. Beltagy, I., Peters, M. E. & Cohan, A. Longformer: The Long-Document Transformer. (2020) doi:10.48550/arXiv.2004.05150.

84. Zaheer, M. *et al.* Big Bird: Transformers for Longer Sequences. (2021) doi:10.48550/arXiv.2007.14062.

85. Dao, T. *et al.* Hungry Hungry Hippos: Towards Language Modeling with State Space Models. (2022) doi:10.48550/arXiv.2212.14052.

86. Dao, T. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. (2023) doi:10.48550/arXiv.2307.08691.

87. Chowdhery, A. *et al.* PaLM: Scaling Language Modeling with Pathways. (2022) doi:10.48550/arXiv.2204.02311.

88. Bommasani, R. *et al.* On the Opportunities and Risks of Foundation Models. (2022) doi:10.48550/arXiv.2108.07258.

89. Ramesh, A. *et al.* Zero-Shot Text-to-Image Generation. (2021) doi:10.48550/arXiv.2102.12092.

90. Rymuza, J. *et al.* Methods for constructing and evaluating consensus genomic interval sets. *Nucleic Acids Research* **52,** 10119–10131 (2024).

91. LeRoy, N. J. *et al.* Fast, memory-efficient genomic interval tokenizers for modern machine learning. *Submitted* (2025).

92. Sheffield, N. C. & Furey, T. S. Identifying and characterizing regulatory sequences in the human genome with chromatin accessibility assays. *Genes* **3,** 651–670 (2012).

93. Xue, B., Khoroshevskyi, O., Gomez, R. A. & Sheffield, N. C. Opportunities and challenges in sharing and reusing genomic interval data. *Frontiers in Genetics* **14,** (2023).

94. Sennrich, R., Haddow, B. & Birch, A. Neural Machine Translation of Rare Words with Subword Units. (2016) doi:10.48550/arXiv.1508.07909.

95. Chen, C. *et al.* This Looks Like That: Deep Learning for Interpretable Image Recognition. (2019) doi:10.48550/arXiv.1806.10574.

96. Sheffield, N. C. & Bock, C. LOLA: enrichment analysis for genomic region sets and regulatory elements in R and Bioconductor. *Bioinformatics (Oxford, England)* **32,** 587–589 (2016).

97. Yan, F., Powell, D. R., Curtis, D. J. & Wong, N. C. From reads to insight: a hitchhiker's guide to ATAC-seq data analysis. *Genome Biology* **21,** 22 (2020).

98. Quinlan, A. R. & Hall, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics (Oxford, England)* **26,** 841–842 (2010).

99. Neph, S. *et al.* BEDOPS: high-performance genomic feature operations. *Bioinformatics* **28,** 1919–1920 (2012).

100. Li, H. & Rong, J. Bedtk: finding interval overlap with implicit interval tree. *Bioinformatics* **37,** 1315–1316 (2021).

101. Feng, J., Ratan, A. & Sheffield, N. C. Augmented Interval List: a novel data structure for efficient genomic interval search. *Bioinformatics* **35,** 4907–4911 (2019).

102. Feng, J. & Sheffield, N. C. IGD: high-performance search for large-scale genomic interval datasets. *Bioinformatics* **37,** 118–120 (2021).

103. Gharavi, E. *et al.* Embeddings of genomic region sets capture rich biological associations in lower dimensions. *Bioinformatics (Oxford, England)* **37,** 4299–4306 (2021).

104. Gharavi, E. *et al.* Joint Representation Learning for Retrieval and Annotation of Genomic Interval Sets. *Bioengineering* **11,** 263 (2024).

105. Zheng, G. *et al.* Methods for evaluating unsupervised vector representations of genomic regions. *NAR Genomics and Bioinformatics* **6,** lqae86 (2024).

106. LeRoy, N. *et al.* Fast clustering and cell-type annotation of scATAC data using pre-trained embeddings. *NAR Genomics and Bioinformatics* **6,** lqae73 (2024).

107. Layer, R. M., Skadron, K., Robins, G., Hall, I. M. & Quinlan, A. R. Binary Interval Search: a scalable algorithm for counting interval intersections. *Bioinformatics* **29,** 1–7 (2013).

108. Li, H. *et al.* Inferring transcription factor regulatory networks from single-cell ATAC-seq data based on graph neural networks. *Nature Machine Intelligence* **4,** 389–400 (2022).

109. Mezger, A. *et al.* High-throughput chromatin accessibility profiling at single-cell resolution. *Nature Communications* **9,** 3647 (2018).

110. Baker, S. M., Rogerson, C., Hayes, A., Sharrocks, A. D. & Rattray, M. Classifying cells with Scasat, a single-cell ATAC-seq analysis tool. *Nucleic Acids Research* **47,** e10 (2019).

111. Ma, W., Lu, J. & Wu, H. Cellcano: supervised cell type identification for single cell ATAC-seq data. *Nature Communications* **14,** 1864 (2023).

112. Chen, H. *et al.* Assessment of computational methods for the analysis of single-cell ATAC-seq data. *Genome Biology* **20,** 241 (2019).

113. Baek, S. & Lee, I. Single-cell ATAC sequencing analysis: From data preprocessing to hypothesis generation. *Computational and Structural Biotechnology Journal* **18,** 1429–1439 (2020).

114. Wang, Y., Sun, X. & Zhao, H. Benchmarking automated cell type annotation tools for single-cell ATAC-seq data. *Frontiers in Genetics* **13,** (2022).

115. Lin, Y. *et al.* scJoint integrates atlas-scale single-cell RNA-seq and ATAC-seq data with transfer learning. *Nature Biotechnology* **40,** 703–710 (2022).

116. Chen, X. *et al.* Cell type annotation of single-cell chromatin accessibility data via supervised Bayesian embedding. *Nature Machine Intelligence* **4,** 116–126 (2022).

117. Buenrostro, J. D. *et al.* Integrated Single-Cell Analysis Maps the Continuous Regulatory Landscape of Human Hematopoietic Differentiation. *Cell* **173,** 1535–1548.e16 (2018).

118. Luecken, M. D. *et al.* A sandbox for prediction and integration of DNA, RNA, and proteins in single cells. in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)* (2021).

119. Altay, A. & Vingron, M. scATAcat: cell-type annotation for scATAC-seq data. *NAR Genomics and Bioinformatics* **6,** lqae135 (2024).

120. LeRoy, N. J. *et al.* Atacformer: A transformer-based foundation model for analysis and interpretation of ATAC-seq data. *Submitted* (2025).

121. Smith, J. P. & Sheffield, N. C. Analytical Approaches for ATAC-seq Data Analysis. *Current protocols in human genetics* **106,** e101 (2020).

122. Schaefer, M. *et al.* Multimodal learning of transcriptomes and text enables interactive single-cell RNA-seq data exploration with natural-language chats. *bioRxiv* 2024.10.15.618501 (2024) doi:10.1101/2024.10.15.618501.

123. Zhang, K. *et al.* A single-cell atlas of chromatin accessibility in the human genome. *Cell* **184,** 5985–6001.e19 (2021).

124. Herring, C. A. *et al.* Human prefrontal cortex gene regulatory dynamics from gestation to adulthood at single-cell resolution. *Cell* **185,** 4428–4447.e28 (2022).

125. Massoni-Badosa, R. *et al.* An atlas of cells in the human tonsil. *Immunity* **57,** 379–399.e18 (2024).

126. Lee, A. J. *et al.* Characterization of altered molecular mechanisms in Parkinson's disease through cell type–resolved multiomics analyses. *Science Advances* **9,** eabo2467 (2023).

127. Patel, A. G. *et al.* The Myogenesis Program Drives Clonal Selection and Drug Resistance in Rhabdomyosarcoma. *Developmental cell* **57,** 1226–1240.e8 (2022).

128. Collin, J. *et al.* A single cell atlas of human cornea that defines its development, limbal progenitor cells and their interactions with the immune cells. *The Ocular Surface* **21,** 279–298 (2021).

129. Clark, K., Luong, M.-T., Le, Q. V. & Manning, C. D. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. (2020) doi:10.48550/arXiv.2003.10555.

130. Reimers, N. & Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. (2019) doi:10.48550/arXiv.1908.10084.

131. Peng, Y. *et al.* Contrastive-learning of language embedding and biological features for cross modality encoding and effector prediction. *Nature Communications* **16,** 1299 (2025).

132. Liu, L., Kim, J. & Bansal, V. Can Contrastive Learning Refine Embeddings. (2024) doi:10.48550/arXiv.2404.08701.

133. Khoroshevskyi, O., LeRoy, N., Reuter, V. P. & Sheffield, N. C. GEOfetch: A command-line tool for downloading data and standardized metadata from GEO and SRA. *Bioinformatics* btad69 (2023) doi:10.1093/bioinformatics/btad069.

134. Chen, T. & Guestrin, C. XGBoost: A Scalable Tree Boosting System. in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 785–794 (2016). doi:10.1145/2939672.2939785.

135. Wei, W. *et al.* Chromatin-sensitive cryptic promoters putatively drive expression of alternative protein isoforms in yeast. *Genome Research* **29,** 1974–1984 (2019).

136. Dunham, I. *et al.* An integrated encyclopedia of DNA elements in the human genome. *Nature* **489,** 57–74 (2012).

137. Chen, H. *et al.* Quantized multi-task learning for context-specific representations of gene network dynamics. *bioRxiv: The Preprint Server for Biology* 2024.08.16.608180 (2024) doi:10.1101/2024.08.16.608180.

138. Gandhi, S. *et al.* Tahoe-x1: Scaling Perturbation-Trained Single-Cell Foundation Models to 3 Billion Parameters. *bioRxiv* 2025.10.23.683759 (2025) doi:10.1101/2025.10.23.683759.

139. Parks, B. & Greenleaf, W. Scalable high-performance single cell data analysis with BPCells. *bioRxiv* 2025.03.27.645853 (2025) doi:10.1101/2025.03.27.645853.

140. Abdennur, N. *et al.* abdenlab/oxbow: v0.4.1. (2025) doi:10.5281/zenodo.17211516.

141. Vig, J. A Multiscale Visualization of Attention in the Transformer Model. in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (eds. Costa-jussà, M. R. & Alfonseca, E.) 37–42 (Association for Computational Linguistics, Florence, Italy, 2019). doi:10.18653/v1/P19-3007.

142. Vashishth, S., Upadhyay, S., Tomar, G. S. & Faruqui, M. Attention Interpretability Across NLP Tasks. (2019) doi:10.48550/arXiv.1909.11218.

143. Li, X. *et al.* A Unified MRC Framework for Named Entity Recognition. in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (eds. Jurafsky, D., Chai, J., Schluter, N. & Tetreault, J.) 5849–5859 (Association for Computational Linguistics, Online, 2020). doi:10.18653/v1/2020.acl-main.519.

144. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. (2020) doi:10.48550/arXiv.1802.03426.

145. Vinh, N. X., Epps, J. & Bailey, J. Information theoretic measures for clusterings comparison: is a correction for chance necessary?. in *Proceedings of the 26th Annual International Conference on Machine Learning* 1073–1080 (Association for Computing Machinery, New York, NY, USA, 2009). doi:10.1145/1553374.1553511.

146. Rosenberg, A. & Hirschberg, J. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* 410–420 (Association for Computational Linguistics, Prague, Czech Republic, 2007).

147. Sloan, C. A. *et al.* ENCODE data at the ENCODE portal. *Nucleic Acids Research* **44,** D726–D732 (2016).

148. Pliner, H. A., Shendure, J. & Trapnell, C. Supervised classification enables rapid annotation of cell atlases. *Nature Methods* **16,** 983–986 (2019).

149. Sheffield, N. C., Stolarczyk, M., Reuter, V. P. & Rendeiro, A. F. Linking big biomedical datasets to modular analysis with Portable Encapsulated Projects. *GigaScience* **10,** giab77 (2021).

150. Traag, V. A., Waltman, L. & van Eck, N. J. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports* **9,** 5233 (2019).

151. Hsu, P.-L. *et al.* Liger Kernel: Efficient Triton Kernels for LLM Training. (2025) doi:10.48550/arXiv.2410.10989.

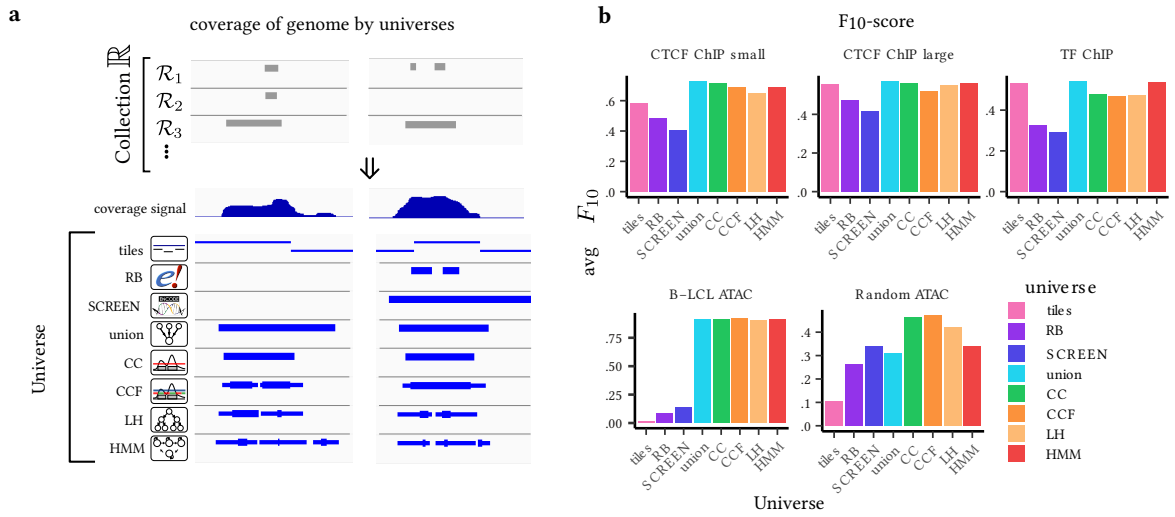
152. Wijmans, E., Huval, B., Hertzberg, A., Koltun, V. & Krähenbühl, P. Cut Your Losses in Large-Vocabulary Language Models. (2025) doi:10.48550/arXiv.2411.09009.

153. Morabito, S. *et al.* Single-nucleus chromatin accessibility and transcriptomic characterization of Alzheimer's disease. *Nature Genetics* **53,** 1143–1155 (2021).

154. Huey, J. D. & Abdennur, N. Bigtools: a high-performance BigWig and BigBed library in Rust. *Bioinformatics* **40,** btae350 (2024).

Appendix A: Supplemental figures and tables

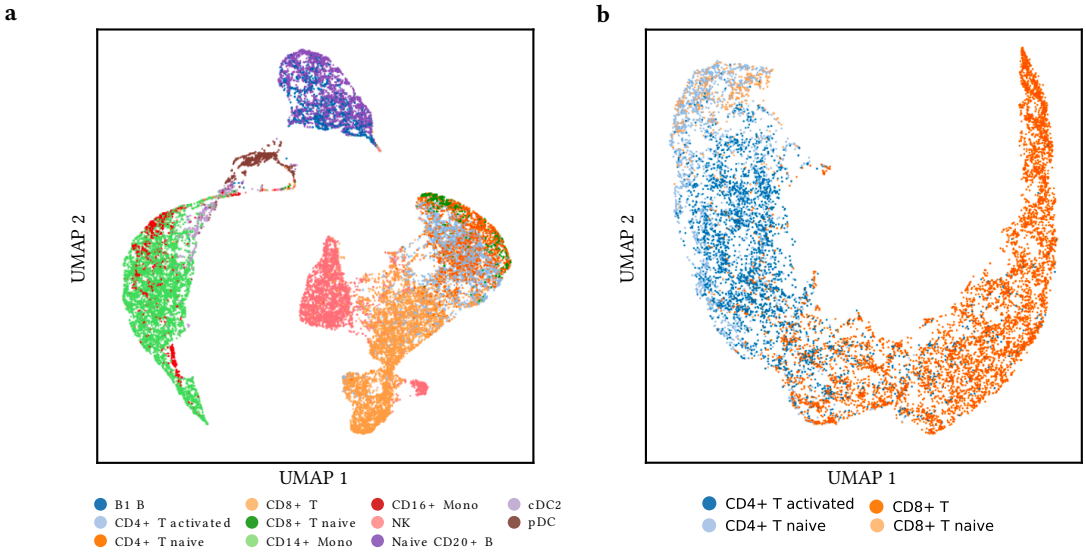
Infrastructure extended figures



Supplementary Figure A.1: Universes overview and results of base-level overlap score.

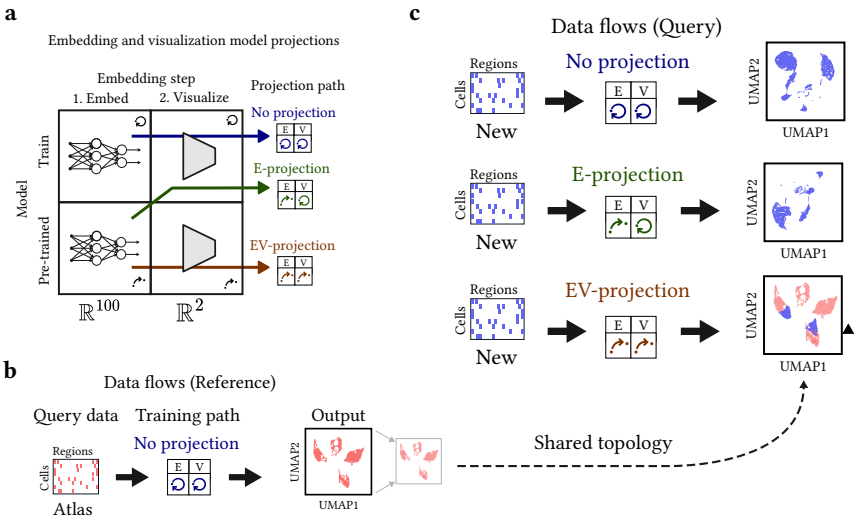
a. Different universes represent genome coverage by the collection to a different extent, example from the Random ATAC collection. Collection R consists of many different files, which are represented by the core signal track. Regions in R1, R2, R3 are best represented by CC, CCF and LH universes in terms of overlap. **b.** Average F_{10}-score for each collection and universes assessment. Across all collections, the HMM universe performs the best in terms of F_{10} score.

scEmbed extended figures



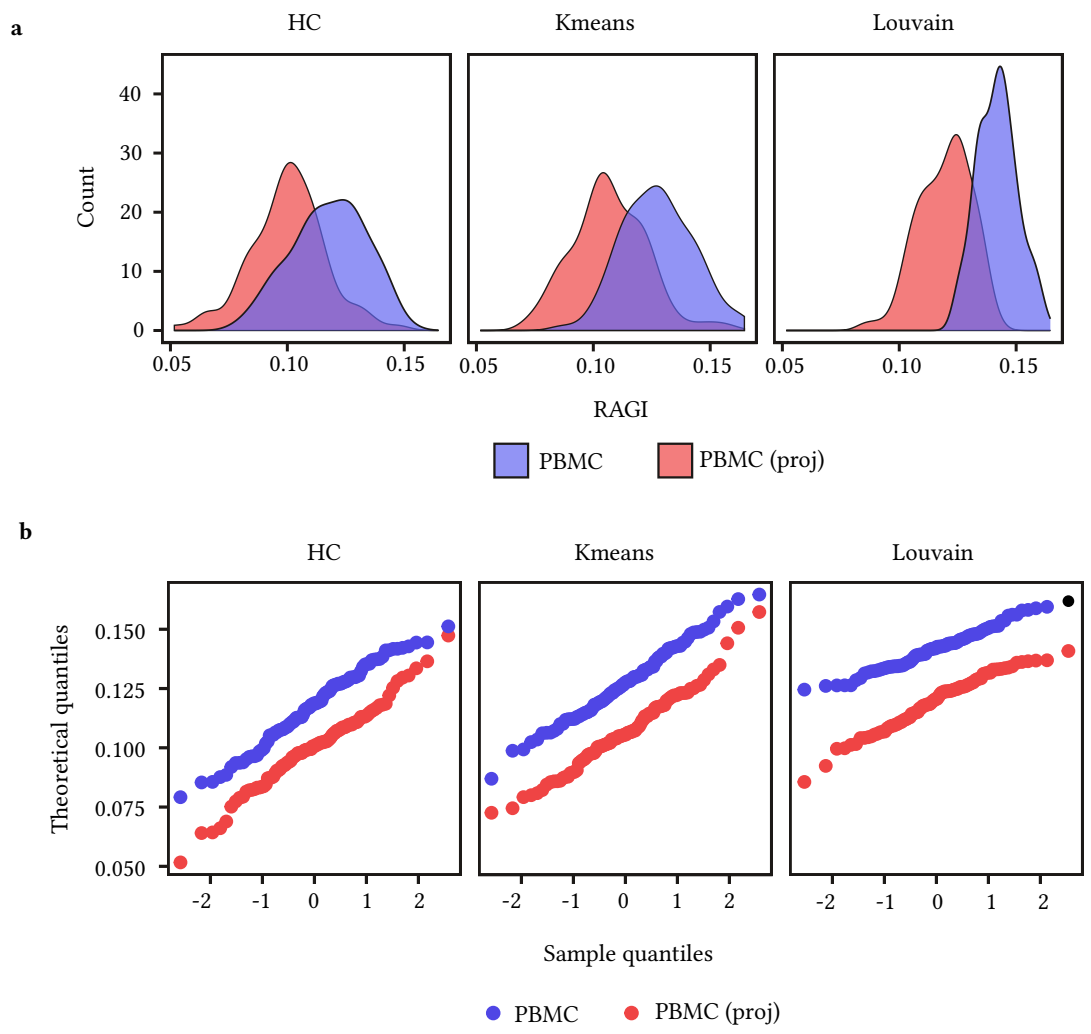
Supplementary Figure A.2: scEmbed clusters cells from the Luecken2021 dataset.

a. UMAP of scEmbed cell embeddings when training the model on all cell-types in the dataset, showing distinct clusters for each cell type. **b.** UMAP of the scEmbed cell embeddings after training only on T Cells.



Supplementary Figure A.3: scEmbed enables knowledge transfer to unseen datasets.

a. Diagram showing scEmbed's three projection paths. **b.** Overview of the standard 'no-projection' data flow. **c.** Overview of three data flows for new data. EV-projection places the new data in the same latent space as the reference data.

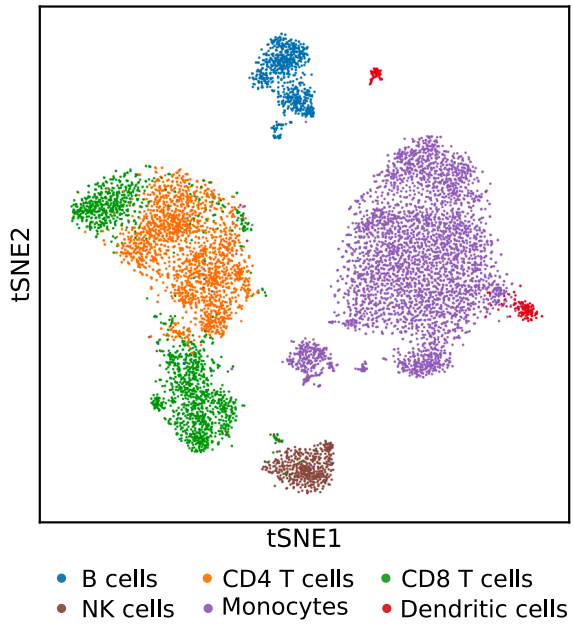


Supplementary Figure A.4: Distributions of the RAGI scores for all subsampled cells.

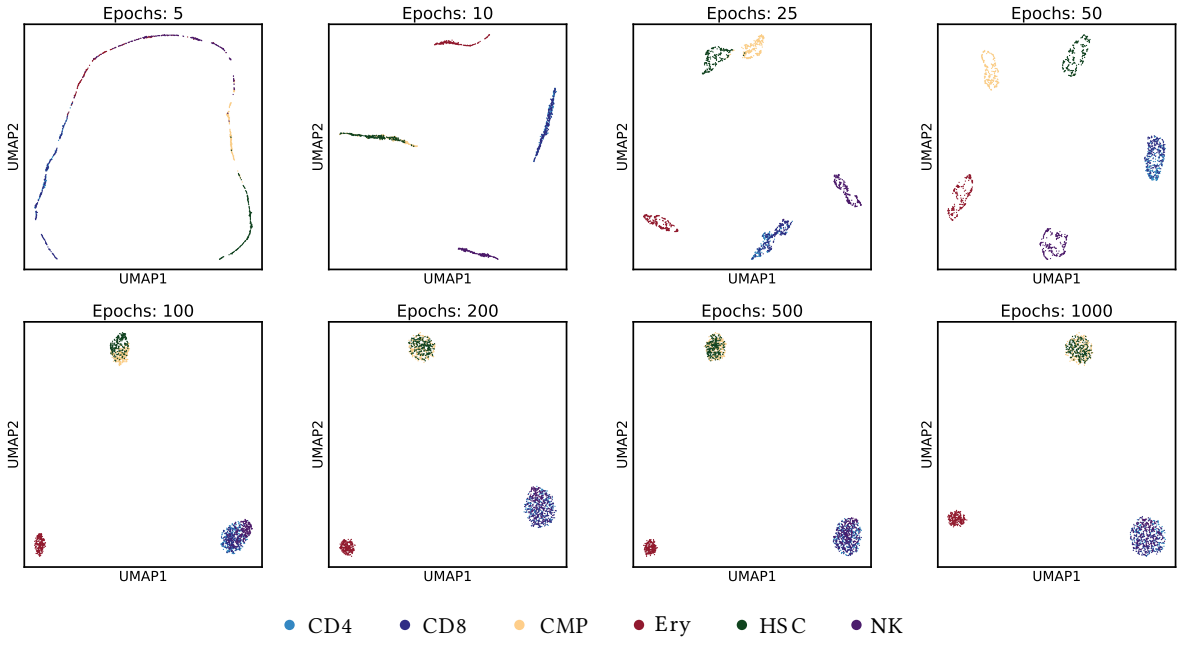
a. Distribution of RAGI scores for cells with embeddings from the new model versus projection through the model trained on the Buenrostro2018 dataset. **b.** QQ plots of the RAGI scores for cells with embeddings from the new model versus projection through the model trained on the Buenrostro2018 dataset.

Supplementary Table A.1: Label mapping between scEmbed and cellcano for consistent comparison of classification performance.

scEmbed label	Cellcano label
B1 B	B cells
CD4+ T activated	CD4 T cells
CD4+ T naive	CD4 T cells
CD8+ T	CD8 T cells
CD8+ T naive	CD8 T cells
CD14+ Mono	Monocytes
cDC2 Dendritic cells	NK NK cells
Naive CD20+ B	B cells



Supplementary Figure A.5: Cellcano cell type annotations for PBMC dataset.

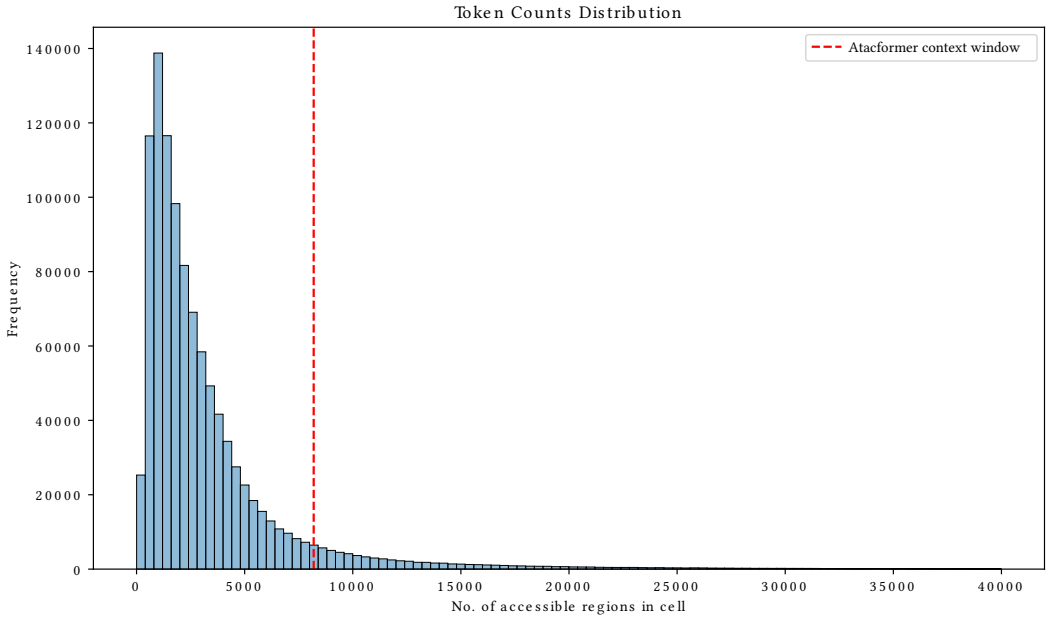


Supplementary Figure A.6: Epoch tests show that scEmbed learns well after 100 epochs.

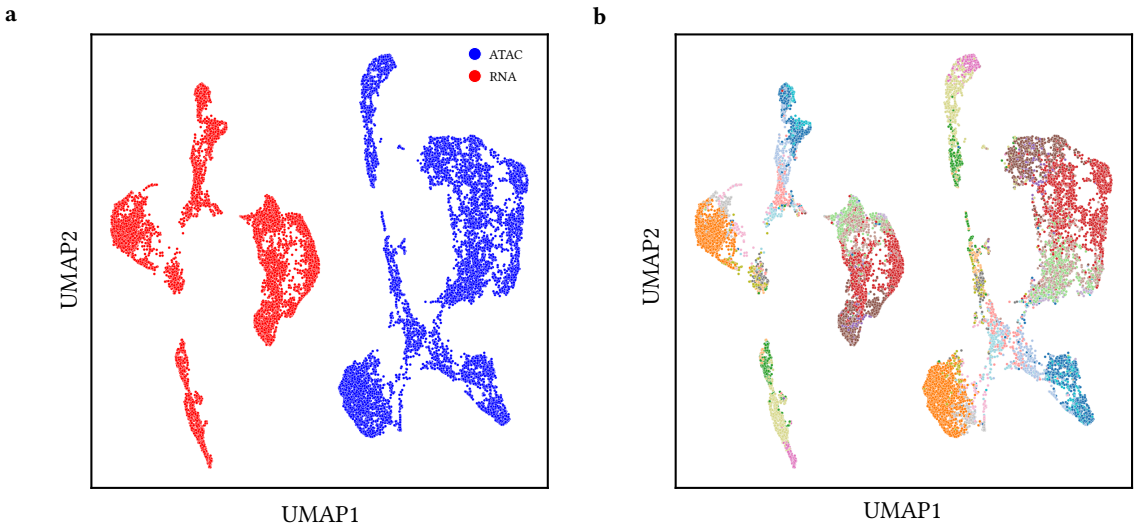
Atacformer extended figures

Supplementary Table A.2: Supplementary Table 1: Detailed information about all datasets used to curate the single-cell atlas for Atacformer.

Dataset	Tissue	Disease state	No. cells	GSE	Author
Human single-cell atlas	Atlas	Healthy	615,998	GSE184462	Ren 2021
Brain 107k	Brain	Healthy	107,057	GSE168408	Lister 2023
Atlas of tonsil	Tonsil	Healthy + Disease	70,775	-	Massoni-Badosa 2025
Luecken2021	Blood	Healthy	69,249	-	Luecken2021
Parkinsons 65k	Brain	Disease	65,589	GSE148434	Jung 2023
Muscle 23k	Muscle	Disease	23,593	GSE174376	Dyer 2022
Kidney 22k	Kidney	Healthy	22,772	-	10X
Cornea atlas	Eye	Healthy	1,209	GSE155683	Lako 2021

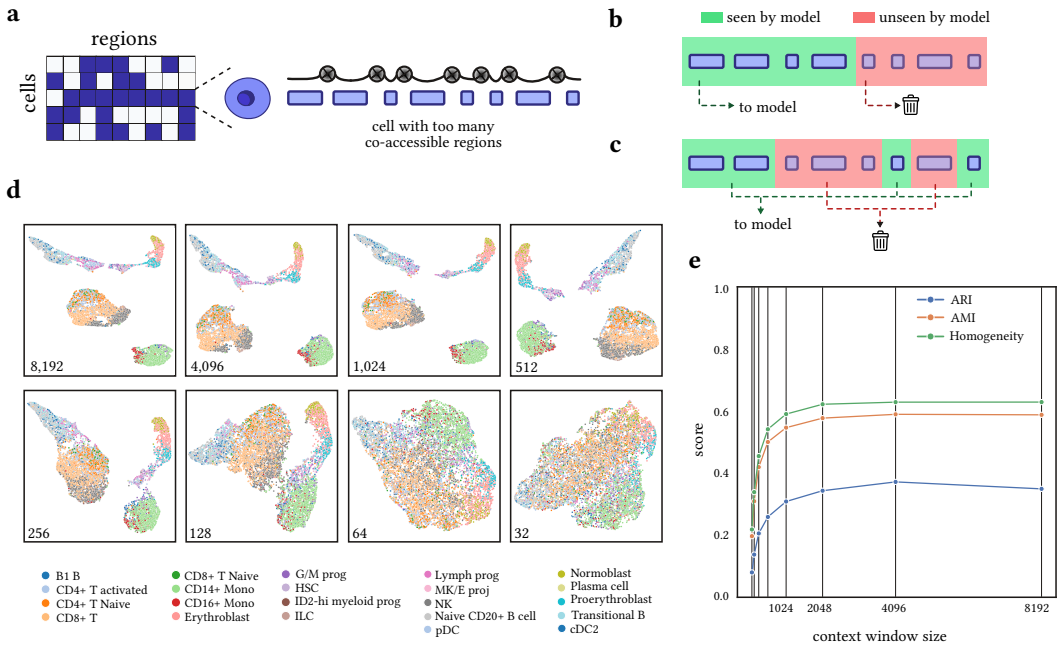


Supplementary Figure A.7: Distribution of context window lengths in the Atacformer training corpus.



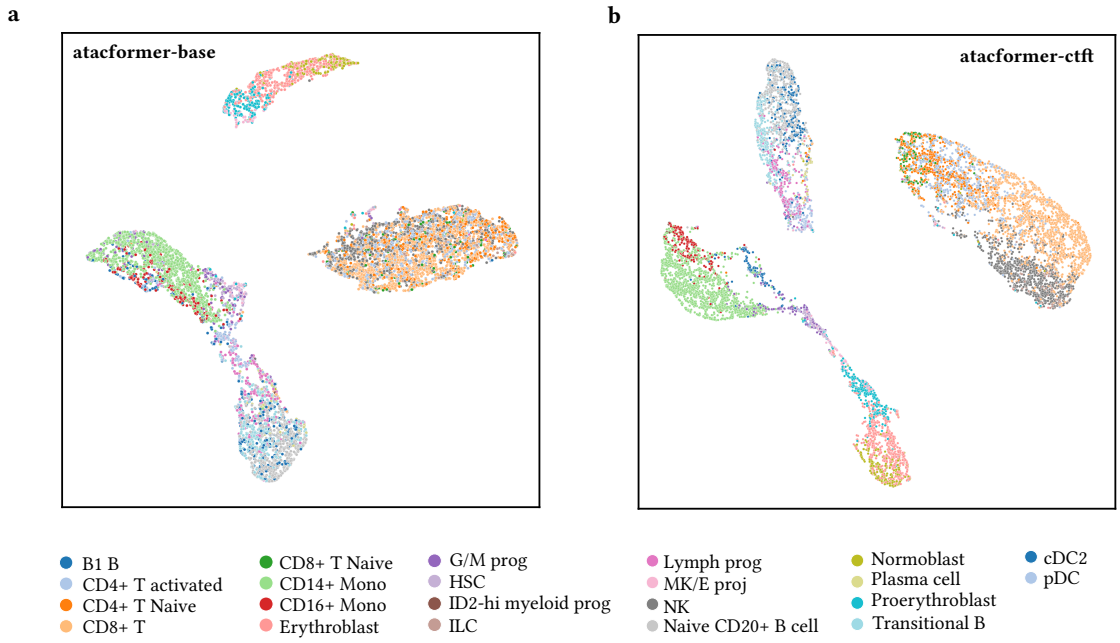
Supplementary Figure A.8: Dual UMAP visualization of both the ATAC and RNA co-embeddings.

a. ATAC and RNA co-embeddings visualized in a shared UMAP space, colored by modality. The two modalities are divided along a shared axis. **b.** ATAC and RNA co-embeddings visualized in a shared UMAP space, colored by cell-type. Cell-type information is preserved.



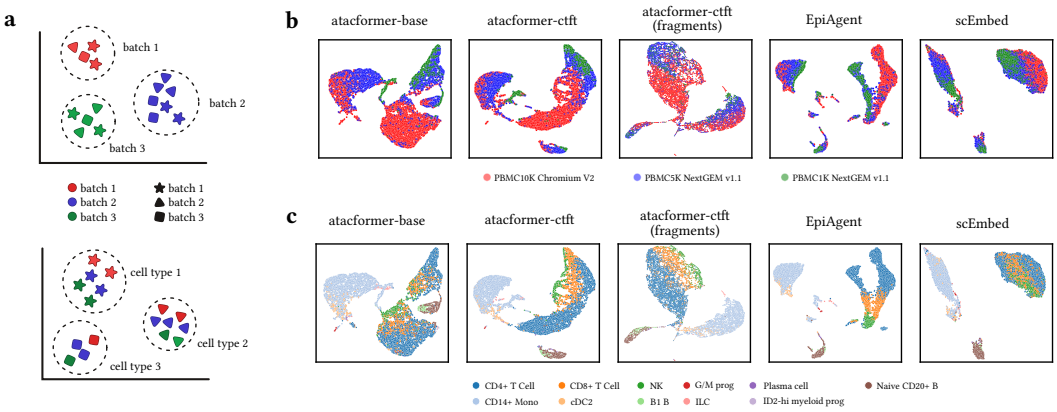
Supplementary Figure A.9: Atacformer is robust to severe degradation in context-window size.

a. Schematic showing how cells are tokenized in the Atacformer framework. When the number of tokens in a cell exceeds the context window of the model, we must choose which tokens to drop before processing. **b.** Schematic of the cut-off method, where we simply keep the first C tokens in a cell, while disregarding the rest (C being the context-window size). **c.** Schematic of the random sample method, where we randomly sample C tokens from the cell, while discarding the rest. **d.** UMAP visualizations of embeddings generated from the Luecken2021 dataset using various context window sizes at inference time. A marked decrease in visually distinct clusters occurs after 512. **e.** Line plot of three clustering metrics as a function of context-window size. All plots and metrics utilized the ATAC encoder of the `craft-100k-hg38` model described in Figure 5.2.



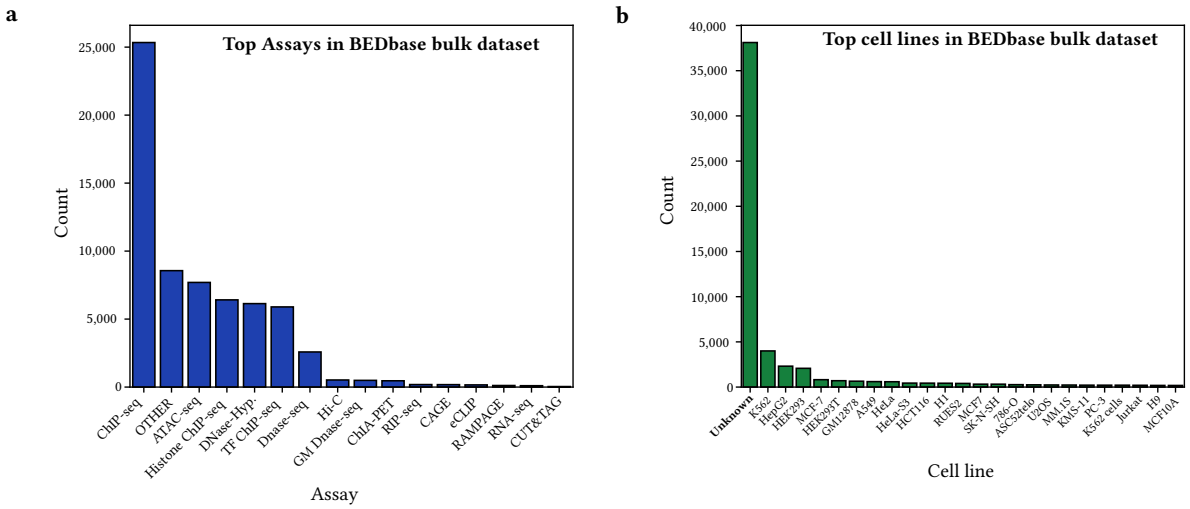
Supplementary Figure A.10: Fine-tuning Atacformer for a cell-clustering task improves latent space separation of individual cells.

a. UMAP visualization of Luecken2021 dataset clustered using `atacformer-base` (before fine-tuning). **b.** UMAP visualization of Luecken2021 dataset clustered using `atacformer-ctft` showing a marked improvement in clustering ability



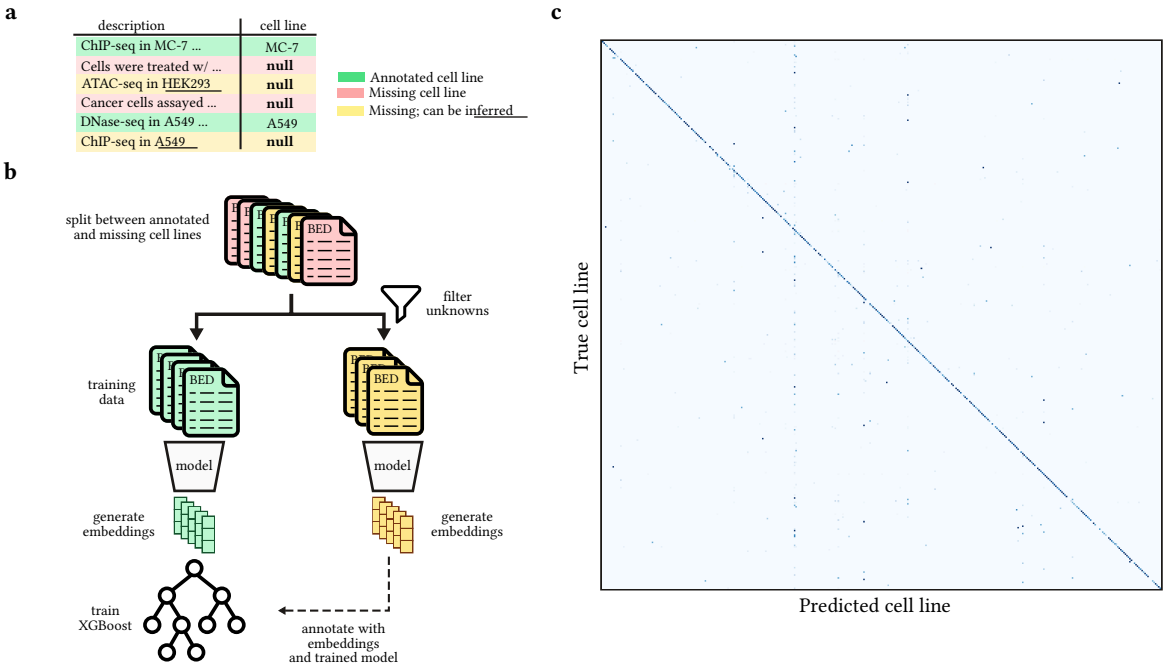
Supplementary Figure A.11: Atacformer performs strong zero-shot batch correction on processed and unprocessed data.

a. Schematic overview of batch effects (top) and mitigation (bottom) when analyzing multiple datasets at once. **b.** UMAP visualizations of three PBMC dataset cell embeddings, colored by dataset. Atacformer visually exhibits equal or better clustering performance when directly producing embeddings of fragment files. **c.** UMAP visualizations of three PBMC dataset cell embeddings, colored by cell-type. Atacformer retains key biological information when directly producing embeddings of fragment files.



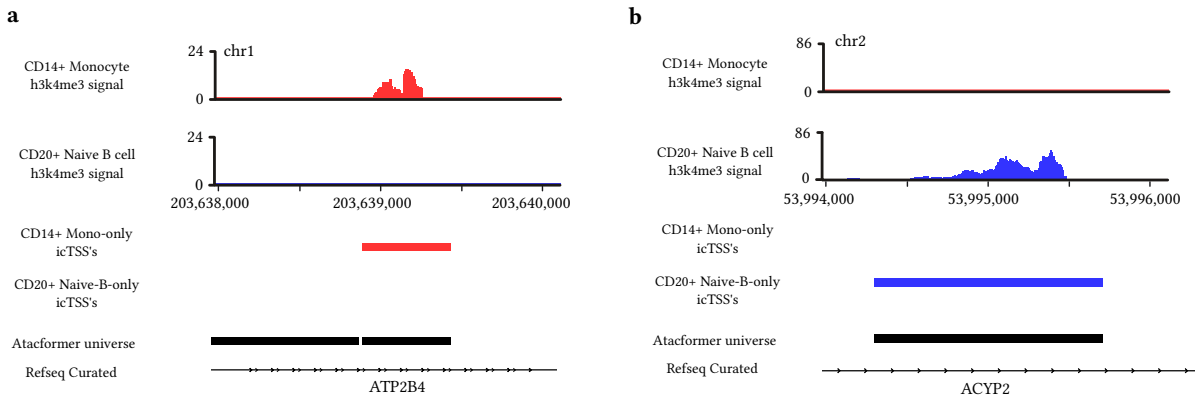
Supplementary Figure A.12: Training dataset assay and cell line distribution for the bulk-ATAC model.

a. Distribution of assay types in the BEDbase bulk data training set. **b.** Distribution of the top 25 cell lines represented in the BEDbase bulk data training set.



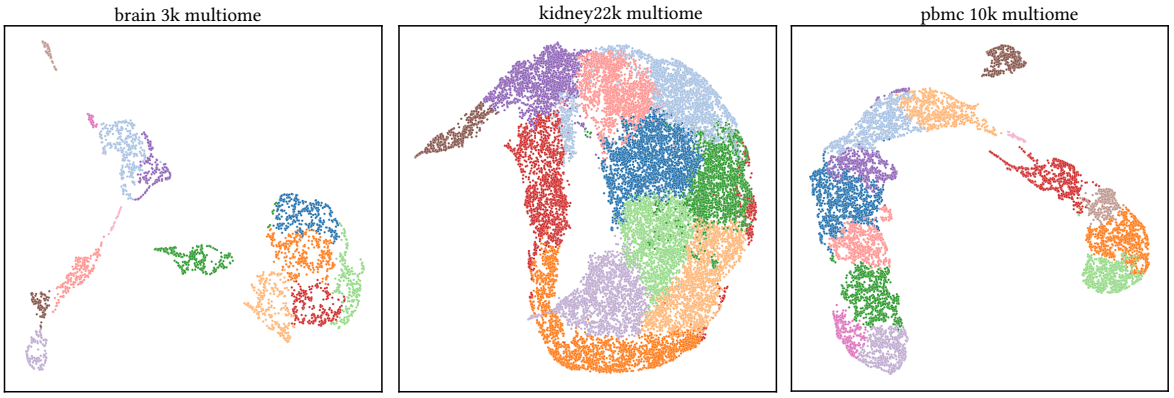
Supplementary Figure A.13: Cell line imputation for missing BEDbase data using a fine-tuned Atacformer model on bulk-ATAC data.

a. Schematic of the imputation procedure. **b.** Confusion matrix for entire cell line dataset.



Supplementary Figure A.14: Extra supplemental anecdotes of H3K4me3 enrichment in icTSS regions.

a. Example icTSS regions showing H3K4me3 in Monocytes. **b.** Example icTSS showing H3K4me3 enrichment in B cells.



Supplementary Figure A.15: Multi-dataset analysis of Atacformer embeddings.

UAMP visualizations of Atacformer embeddings generated from multiple datasets. Cluster colors indicate leiden clusters generated from the combined dataset. Atacformer performs well on datasets inside its training distribution (Brain and blood) while struggling on datasets outside its training distribution (Kidney).

Appendix B: Extended methods

Common methods across results

Clustering methodologies

We use three clustering algorithms: Hierarchical clustering (HC), k-means clustering, and Louvain clustering. For HC and k-means, we use the `scikit-learn` implementations. When ground-truth labels were known for a particular dataset, we used the number of unique labels to set the number of clusters to generate. Otherwise, we used prior knowledge to estimate the number of unique cell populations we would expect to find. For Louvain clustering, we use the `scanpy` implementation. Louvain is agnostic to a specified number of clusters. As such, we iteratively applied clustering to datasets while slowly increasing the resolution value from 0 to 3. With each iteration, the number of clusters was stored in a list along with the corresponding resolution. Once complete, we employed binary search on the list to identify the resolution that gave us the desired number of clusters. This value was used to generate the final clustering solution.

Embedding visualization

We used uniform manifold approximation and projection (UMAP) to visualize single-cell embeddings[144]. We used the `umap-learn` Python package and specified two dimensions for each visualization. In addition, a random state of 42 was set for visualization workflows. All other parameters were set to package defaults

Clustering evaluation

Three scores are employed when a dataset has ground-truth labels: The adjusted rand index (ARI), the adjusted mutual info score (AMI), and the homogeneity score.

Adjusted Rand Index

The ARI is a metric for evaluating the similarity between two data clusterings. This is achieved by counting pairs that are assigned to the same cluster label. Mathematically, it is computed by:

$$\text{ARI} = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}} \tag{1}$$

where n_{ij}, a_i, b_j are diagonal values, row sums, and column sums respectively from the contingency table that describes the frequency distribution of the cluster labels from ground-truth and predicted clusterings. We use the adjusted rand score function from the `scikit-learn` python package.

Adjusted Mutual Info Score

The AMI, intuitively, is a measure of the amount of information that two clusterings share. It's used to evaluate how well two clusterings agree with each other[145]. We compute AMI through the `scikit-learn` package using the adjusted mutual info score function.

Homogeneity Score

The homogeneity score is an entropy-based external cluster evaluation metric that measures how far from perfect an incorrect clustering solution is[146]. We employ the `scikit-learn` homogeneity score function to measure this metric for each dataset.

Cell-type classification evaluation

When ground-truth labels are known, we evaluate cell-type classification performance using accuracy, precision, recall, and F1 score. These metrics provide a comprehensive view of the model's performance in assigning the correct cell types to individual cells.

F1 score

The F1 score is the harmonic mean of precision and recall, and provides a balance between these two measures. Precision is the number of true positives divided by the sum of true positives and false positives, and recall is the number of true positives divided by the sum of true positives and false negatives. Formally, these are defined as:

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2}$$

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{3}$$

The F1 score is defined as:

$$\mathrm{F1} = 2 \times \mathrm{P} \times \frac{\mathrm{R}}{\mathrm{P} + \mathrm{R}} \tag{4}$$

To compute these measures, we compare the predicted labels, denoted as $L_p = l_{p1}, l_{p2}, ..., l_{pn}$, to the ground truth labels, denoted as $L_g = l_{g1}, l_{g2}, ..., l_{gn}$, where n is the total number of data points (or clusters). We utilize the metrics.f1 score function from `scikit-learn` to compute this value.

Infrastructure extended methods

Gtars and uniwig

The uniwig functionality is implemented in Rust as part of our gtars library gtars, providing efficient genomic data processing capabilities. The tool accepts BED format files as input and uses efficient multi-threading to compute genome-wide coverage at single-base-pair resolution. Coverage computation is parallelized across chromosomes to maximize performance on multi-core systems. The output consists of three BigWig files representing start, core, and end coverage signals, which are generated using the `bigtools` library for compatibility with existing genomic browsers and analysis tools. A command-line API is provided for integration into existing workflows, with configurable parameters for coverage thresholds and output formats.

Universe construction methods

We implemented several methods for constructing consensus genomic interval sets, or "universes," as part of our `geniml` library geniml.

Coverage cutoff (CC) universe

This method constructs a universe by identifying a statistically principled coverage threshold. The core idea is to model the probability of any given genomic position being "covered" versus "not covered" (background) across the entire collection of input files. The method then derives a cutoff value by finding all positions where the probability of being covered is greater than the probability of being background.

Practically, this results in a simple rule: a genomic position, i, is included in the universe if its coverage frequency, $\text{freq}_{c(i)}$, is greater than or equal to the *average* coverage across the genome $\left(\frac{S_c}{g}\right)$, where S_c is the total coverage and g is the genome length. This provides a data-driven approach to defining a consensus set based on signal enrichment. And S_c is computed as:

$$S_c = \sum_{\text{i=1}}^{\text{g}} \text{freq}_{c(i)} \tag{5}$$

Practically, we utilize the `pyBigWig` library to import and process the BigWig coverage files generated by `uniwig`. The final universe is constructed using `numpy` for efficient numerical operations.

Maximum likelihood (LH) universe

The Maximum Likelihood (LH) universe improves upon the simple coverage model by incorporating information about region boundaries to better preserve distinct genomic features and prevent them from being merged.

This method uses three separate signal tracks as input: starts, cores (coverage), and ends. It then builds a probabilistic model to calculate the likelihood of each genomic position belonging to one of four states: start, core, end, or background. Using dynamic programming, the algorithm finds the most likely sequence of these states across the entire genome, effectively segmenting it into a set of flexible regions. Finally, a filtering step is applied to remove very small or low-likelihood regions to enhance the quality of the final universe.

Coverage data is processed using `pyBigWig` for efficient BigWig file reading, while `numpy` provides the numerical computing foundation for matrix operations and array manipulations in the dynamic programming algorithm. For performance optimization, the method conditionally uses `numba` when available to just-in-time compile the core dynamic programming routine, significantly accelerating the likelihood calculations.

Hidden Markov Model (HMM) universe

The Hidden Markov Model (HMM) approach offers a more sophisticated and tunable alternative to the LH method. It models the genome as a sequence of four hidden states: start (0), core (1), end (2), and background (3). The three signal tracks (starts, cores, and ends) are treated as emissions observed from these hidden states.

The implementation utilizes a Poisson emission model with predefined transition matrices and lambda parameters that were empirically optimized. To handle large chromosomes efficiently, the algorithm employs a segmented prediction strategy that identifies regions containing non-zero coverage and applies HMM decoding only to those segments, with background state assigned to empty regions.

The method processes each chromosome independently, reading BigWig files using the `pyBigWig` library with optimized numpy array operations when available. Coverage data is stored as 16-bit unsigned integers to minimize memory usage while maintaining precision. The core HMM computation uses `scipy.stats` for negative binomial quantile calculations and a custom Poisson model implementation for state prediction.

Tokenization methods

We developed the `gtars-tokenizers` library as a high-performance tool for mapping genomic interval data to predefined vocabularies, facilitating their use in machine learning applications. Written in Rust, we implement and make available two interval overlap algorithms: 1) a binary interval tree search[107], and 2) an augmented-interval list method[101].

Environment bindings

For python binings, we leverage the `pyo3` crate to implement an interface that enables in-memory processing from python. For R-bindings, we use the `extendr` crate to provide a similar interface. The command-line interface is built using the `clap` crate, allowing users to specify input files, vocabularies, and output formats. Finally, we leverage `wasm-bindgen` to compile the library to WebAssembly, enabling its use in web applications and other environments that support WASM.

Tokenization benchmarking

For benchmarking datasets, we used the SCREEN universe from ENCODE as the reference vocabulary and subsampled it to create vocabularies of three different sizes: 1 million, 100,000, and 10,000 regions[147]. For query datasets, we randomly downloaded ten files from ENCODE with varying numbers of genomic regions to assess performance across different query sizes and characteristics.

To evaluate the performance of our tokenization methods, we developed a comprehensive benchmarking framework using Python. The benchmarking system utilizes `subprocess` for command execution and `threading` for concurrent memory monitoring during benchmark runs. Memory usage is tracked using the `psutil` library, which provides cross-platform process monitoring capabilities at configurable intervals. Configuration management is handled through `yaml` for flexible test parameter specification, while `polars` provides efficient data processing and CSV export functionality for results analysis. The framework executes multiple repetitions of each benchmark to ensure statistical reliability, measuring both execution time and peak memory consumption. This automated approach enables systematic comparison of different tokenization algorithms across various dataset sizes and configurations.

Software and data availability

The `gtars`, `uniwig` and the `gtars-tokenizers` libraries are open-source and available on GitHub at https://github.com/databio/gtars. The universe creation methods are implemented in the `geniml` library, which is also open-source and available at https://github.com/databio/geniml.

scEmbed extended methods

Model architecture and training

We used the `gensim` python implementation of Word2Vec as the core model for scEmbed. Word2Vec has many configurable hyperparameters[65], including context window size, embedding size, learning rate scheduling and number of epochs. All experiments were conducted with a fixed set of hyperparameters. We used defaults for scEmbed, informed by experiments on Region2Vec optimization[105]. Specifically, we use a window size of 5 and an embedding dimension of 100. We also use 100 epochs (Supplementary Figure A.6) for all experiments unless otherwise noted. We adopt an exponential learning rate schedule with a decay rate of 0.95. After training, we extract the learned region embeddings from the Word2Vec model and subsequently transfer them to an equivalent model in `pytorch`. This improves accessibility and interoperability with other deep learning frameworks.

Data and data processing

Detailed overview of datasets:

Luecken2021. The Luecken2021 dataset is a multimodal single-cell benchmarking dataset (27). The data is a first-of-its-kind multimodal benchmark dataset of 120,000 single cells from the human bone marrow of 10 diverse donors measured with two commercially-available multi-modal technologies: nuclear GEX with joint ATAC, and cellular GEX with joint ADT profiles. The data was retrieved from the gene expression omnibus (GEO) using the GEO accession GSE194122.

Buenrostro2018. The Buenrostro2018 dataset consists of single-cell chromatin accessibility profiles across 10 populations of immunophenotypically defined human hematopoietic cell types (26). Deduplicated single-cell bam files along with a consensus peak set were provided by Chen et. al. (13). Using bedtools (32), region overlaps with the consensus peak set were computed for each bam file at a minimum overlap of 1bp. Using the -c flag, the number of overlaps with each region in the consensus peak set was calculated. Overlap count files were subsequently converted into a cell by peak binary accessibility matrix formatted as a comma-separated-value file (csv). Finally, the binary accessibility csv was converted into a scanpy AnnData object using the scanpy.read csv API. This was used as input to the scEmbed model.

5k PBMC. The PBMC dataset comes from 10X genomics and consists of peripheral blood mononuclear cells (PBMCs) from a healthy donor. Three files were downloaded directly from the 10X genomics website: 1) the sparse peak matrix in `.mtx` format, 2) the cell barcode labels in tsv format, and 3) the consensus peak set in bed format. Using Python, along with pandas and scanpy, these files were processed into a scanpy AnnData object. This was used as input to the scEmbed model.

Synthetic Bone Marrow. The synthetic bone marrow dataset was described and provided by Chen et. al. (13). The binary accessibility matrix was downloaded directly from the Pinello Lab's GitHub as a .rds file. Using R, this file was read, parsed, and exported as a csv. Like the previous two datasets, this csv was processed into a scanpy AnnData object using pandas and scanpy.

Tokenization of new data

To tokenize cells, we use our previously designed genomic tokenizers[91]. Briefly, we take an individual cell and identify each region where it shows signal. We define signal as anything in the binary accessibility feature matrix with a value greater than zero. These regions are then collected and we use interval overlap arithmetic to create a set of tokenized genomic intervals for each cell. Specifically, we employ the Augmented Interval list[101] to perform fast, memory-efficient overlap computation between a query cell and the target universe/vocabulary. These tokens are then shuffled and fed into the Word2Vec model for training.

Model benchmarking and evaluation

To validate scEmbed, we followed an earlier approach[112] to benchmark it on clustering tasks using published reference scATAC data. We leveraged four main datasets in this work: 1) the Buenrostro2018 dataset, single-cell chromatin accessibility profiles from 10 human hematopoietic cell types[117]; 2) Luecken2021, a multimodal single-cell benchmarking dataset of 120,000 single cells from the human bone marrow of 10 diverse donors measured with two commercially available multimodal technologies[118]; 3) 10X genomics 5k PBMCs, a single-cell dataset of 5000 peripheral blood mononuclear cells from a healthy donor; and 4) a synthetic bone marrow dataset, a binary accessibility dataset described and provided by Chen et al.[112].

We benchmarked scEmbed on the Buenrostro2018 dataset[117] as well as a more recent and comprehensive scATAC-seq dataset from Luecken2021[118]. We trained scEmbed for 100 epochs (Supplementary Figure A.6) then used the resulting region embeddings to construct cell embeddings. Following previous benchmarking procedures, we clustered the cell embeddings with three clustering methods: K-means, hierarchical clustering (HC), and Louvain clustering. There are two scenarios for which we can evaluate clustering: known ground-truth labels and unknown ground-truth labels. The synthetic bone marrow, Buenrostro2018, and Luecken2021 datasets have known ground-truth labels while the PBMC data have unknown ground-truth labels. When ground-truth labels are known, we employ three scores: the adjusted rand index (ARI), the adjusted mutual info score (AMI), and the homogeneity score. When ground-truth labels are not known, we use the Residual Average Gini Index (RAGI)[112]

Dropout experiments

To explore its transfer learning ability and test robustness to missing data, following Xiong et al.[39], we evaluated scEmbed on datasets with increasing levels of information loss. Starting from the already sparse Buenrostro2018 cell-feature matrix (2.8% non-zero)[90], we randomly dropped non-zero values in the binary accessibility matrix until ~80% of the non-zero data were lost, resulting in a matrix that was 0.5% non-zero

Residual Average Gini Index

When ground truth labels are unknown, all aforementioned evaluation metrics are no longer applicable. As such, we need a measure that can still evaluate dataset clustering based on what one would expect, given some sort of prior knowledge about the system. For this, we employ a similar strategy described by Chen et. al. called the Residual Average Gini Index (RAGI). Briefly, the RAGI score compares the accessibility of housekeeping genes with previously characterized marker genes[148]. RAGI measures the average residual specificity of a clustering solution with respect to marker genes, suggesting that a good clustering solution should have clusters enriched for different marker genes and these genes should be highly accessible in only a few clusters, compared to the less informative housekeeping genes.

Transfer learning and projection

scEmbed was designed from the outset to facilitate fast, powerful transfer learning. This transfer approach allows scEmbed to take advantage of pre-trained reference models. We call this 'projection' because we 'project' new data into the latent space of the original dataset, creating cell embeddings for new data using a pre-trained model. Projection occurs in three steps: first, we train a model on reference data to produce region embeddings for each region in the reference consensus region set. For datasets where consensus peaks do not exist, methods that create such sets from raw data could be used as a pre-processing step[90]. Second, we take a new single-cell dataset and map the regions to the reference consensus region set using region overlaps. This represents each single cell in the new dataset using the set of regions from the reference dataset, for which we also have region embeddings from the reference model. Finally, we compute the average of all region embeddings for each cell in the new dataset. This approach leverages the information from a larger atlas of accessibility data to analyze a new dataset. In fact, the original training data need not come from scATAC-seq at all. Using this approach, a model trained with bulk ATAC-seq could be used to project scATAC-seq data. This provides an enormous advantage by utilizing the patterns of region co-occurrence from the vast volume of publicly available region set data to inform cell embeddings of single-cell data.

Projection visualization and cell-type annotation

Projection data flows

We distinguish between three data flows that can occur with scEmbed: no projection E-projection and EV-projection (Supplementary Figure A.3 A). First, we train a reference model using the typical no projection flow (Supplementary Figure A.3 B). This is the standard workflow of training a new scEmbed model on some input data and visualizing the resulting embeddings by fitting a UMAP model to reduce the dimensionality to two. Then, given a new query dataset, we could analyze it with any of the three data flows (Supplementary Figure A.3 C). In a no projection flow, we would not use the reference model at all; we train and visualize using only the new dataset. In embedding-only projection data flow (E-projection), new data are first embedded using the pre-trained reference model.

These embeddings may then be visualized by fitting a UMAP model to reduce dimensionality to two dimensions. The third data flow, and the novel innovation that accomplishes our goal of reference-based visualization, is the embedding and visualization workflow (EV-projection). In this data flow, new data are first embedded using the pre-trained reference model, as in E-projection; then, these embeddings are further projected through a UMAP model that was fit on the reference data embeddings, rather than newly fit. With the EV-projection flow, plotting the two-dimensional cell representations on top of the reference data UMAP plot allows one to visualize where in the original embedding space the new data ended up. This is possible because the EV-projection re-uses the same topology from the UMAP model fit to the reference data (Supplementary Figure A.3 C).

Evaluation of cell type annotation

We use Cellcano, a novel scATAC-seq cell annotation method, to assign ground truth labels to our new PBMC data[111] (Supplementary Figure A.5). We follow the online tutorials (https://marvinquiet.github.io/Cellcano/) and leverage their provided reference dataset to process the new PBMC data. Once ground truth labels have been assigned by Cellcano and putative cell types are assigned, we can evaluate the performance of our model using the previously described classification metrics (see common methods).

Atacformer extended methods

Data collection and pre-processing

To develop a large training dataset of single-cell ATAC-seq data, we identified, downloaded, and uniformly processed data from three main sources: 1) the Gene Expression Omnibus, 2) the Human Cell Atlas, and 3) the 10X genomics dataset repository. Detailed information on dataset contents and availability can be found in supplemental. All datasets, unless noted, were downloaded as raw `.fastq` files. We designed and built a multi-stage pipeline to uniformly process the `fastq` files. First, we utilized CellRanger ATAC `2.1.0` to convert the fastqs into processed fragment files. Next, each fragment file was imported and initially processed using SnapATAC2[37]. We utilized all recommended parameters noted in the "atlas-scale analysis" tutorial ("https://kzhang.org/SnapATAC2/tutorials/atlas.html"). Both steps were parallelized on our computing cluster using the `looper` and `PEP` framework[149].

Next, we again leveraged SnapATAC2 to perform atlas-wide dimensionality reduction, batch correction, and clustering. We performed a two-stage clustering approach. First, a coarse clustering across the entire dataset using Leiden clustering[150], and then a secondary intra-cluster clustering also using leiden within each cluster. This yielded 359 distinct single-cell clusters within the atlas across all datasets. Each of the 359 clusters was pseudo-bulked into separate `.fragment.tsv` files for downstream analysis.

Generation of a uniform model vocabulary

A uniform vocabulary is essential for genomic region tokenization. For this, we leveraged both public and previously developed tools by our lab for generating genomic interval consensus sets. We utilized `macs3` to perform peak-calling[25] on each of the 359 pseudo-bulked fragment files, resulting in 359 corresponding `.narrowPeak` files. Specifically, we used the `peakcall` function with the following parameters: `-g hs -f BEDPE -q 0.01 --nomodel --shift -75 --extsize 200 --keep-dup all -B --SPMR`.

We next utilized our tool `uniwig` to unify all 359 peak sets into big wig (`.bw`) coverage tracks for the start, cores, and ends of all called peaks across all clusters. For `uniwig` we used the following parameters: `-m 5 -s 1 -y wig -z 2`. This resulted in three coverage track files for the starts, cores, and ends.

Finally, we used the coverage tracks as input into our previously published universe creation methods in `geniml`[90]. Using both the coverage cutoff and hidden markov model (HMM) algorithms, two

consensus sets were created. These two bedfiles were finally merged into one unifying vocabulary for the model using `bedtools merge`[98].

The final vocabulary has 890,704 distinct genomic regions into which all cells are tokenized.

Genomic interval tokenization

We conceptualized a unique tokenization method for our models that is designed to be as flexible and simple as possible, enabling broad use of Atacformer for many data types including bulk-ATAC seq data. Any entity that can be represented as a BED-file is valid input to the model. As an example, a single-cell from a scATAC-seq experiment can be thought of as a "bag of co-accessible regions"consisting of a few thousand open chromatin regions. Each of these regions is overlapped with the model's pre-defined vocabulary and serves as input to the embedding module and subsequent transformer encoder.

We leverage two highly-efficient methods for interval overlap computation: AIList[101] and BITS[107]. We've implemented both algorithms in Rust and have made them available in Python for in-memory tokenization with a huggingface-compatible API. Code and documentation for our tokenizers can be found on GitHub in our `gtars` crate/package "https://github.com/databio/gtars".

ELECTRA pre-training methodology

To pretrain Atacformer on single-cell data, we employed an ELECTRA-style replaced-token detection strategy. While most transformer encoder models use masked language modeling (MLM) for self-supervised pretraining[42], we found that MLM was poorly aligned with the properties of our model and data modality in two key ways. First, MLM requires computing a full probability distribution over the vocabulary at each training step. With nearly 1 million tokens, this becomes computationally intractable. Only recently have methods emerged to address this issue. Motivated by the growing vocabulary sizes in modern large language models (LLMs), techniques like Liger kernels[151] and Cut Cross Entropy[152] use linear-time approximations of cross entropy to dramatically reduce space and time complexity. We found these strategies applicable to Atacformer as well, since MLM is fundamentally a token prediction task. However, a second, more fundamental problem emerges: we identified that MLM effectively enforces an order among masked tokens, while shuffling the tokens should have no effect on the information content of a single-cell or corresponding regionset. More specifically, we recognized that the model predicting the correct tokens, but out of order was common and would provide an incorrect training signal to the model.

ELECTRA side-steps both of these problems entirely by reframing the pre-training task as binary classification: for each token, the model predicts whether it was replaced or not. This approach

does not depend on the model predicting tokens in a specific order, and moreover, doesn't require computing a probability for all 890K tokens. Although genomic coordinates offer a natural means to introduce sequence order, we sought to avoid the model overly relying on deterministic positional cues, instead incentivizing it to capture meaningful biological patterns and relationships. To that end, Atacformer is distinctly free of any form of positional information in its input embeddings.

Formal specification of tokenization and pre-training

We begin by fixing a **global vocabulary** U of non-overlapping genomic regions derived from our consensus universe (see previous section):

$$U = \{v_{\{1\}}, ..., v_{\{V\}}\} \tag{6}$$

For each single cell c we observe an unordered set of raw, **co-accessible** regions

$$R_c = \{r_1, ..., r_{N_c}\} \tag{7}$$

Tokenization reduces these raw intervals to their canonical vocabulary representatives via a simple interval intersection:

$$I_c = \{v \in U; \exists r \in R_c; v \cap r \neq \emptyset\} \tag{8}$$

We then map each vocabulary element to its integer identifier, producing a sequence of token indices

$$T_c = \{\mathrm{id}(v); v \in I_c\} \tag{9}$$

To create a learning signal we apply **ELECTRA-style corruption**[129]. Each position j is independently selected for replacement with probability $p = 0.45$:

$$\delta_j \sim \mathrm{Bernoulli}(p), z_j \sim \mathrm{Unif}(U) \tag{10}$$

The corrupted token sequence is therefore:

$$c, j = \begin{cases} z_j \text{ if } \delta_j = 1 \\ T_{\{c,j\}} \text{ if } \delta_j = 0 \end{cases} \tag{11}$$

Every token id is looked up in a shared embedding matrix $E \in \mathbb{R}^{V \times d}$ to obtain dense vectors

$$x_j = E_{T_{c,j}}, \tag{12}$$

$$X_c = \left(x_1, ..., x_{|I_c|}\right) \tag{13}$$

These embeddings pass through L stacked transformer encoder layers (no positional encodings are supplied):

$$H^{(0)} = X_c, \left(H^{\ell-1}\right) \tag{14}$$

$$H^{\ell} = \text{TransformerLayer}_{\ell} \tag{15}$$

$$L = 1, ..., \ell \tag{16}$$

The final contextual embedding at each position j is

$$h_j = H_j^{(L)}. \tag{17}$$

A lightweight classifier head converts each contextual vector into the probability that the original token was **replaced**:

$$s_j = W_o h_j + b, \quad \hat{y}_j = \sigma\big(s_j\big). \tag{18}$$

Training minimizes the binary cross-entropy **replaced-token detection** loss over all positions in the cell:

$$\mathcal{L}_c = -\sum_{j=1}^{|I_c|} \left[\delta_j \log\big(\hat{y}_j\big) + \big(1-\delta_j\big) \log\big(1-\hat{y}_j\big)\right] \tag{19}$$

Averaging $\mathcal{L}_c$ over the mini-batch and optimizing with AdamW completes the pre-training procedure.

Cell embedding calculation

To generate single-cell embeddings, we first tokenize a single-cell according to the steps outlined above. We then pass the tokens through the model to obtain a contextualized embedding representation for each region-token in that cell:

$$H^{\ell} = \text{AtacformerBase}\big(\{e_1, ..., e_j\}\big) \in \mathbb{R}^{j \times d_{\text{model}}} \tag{20}$$

Where e_j is the initial token embedding for the j^{th} region token. We then obtain cell-embeddings by pooling all contextualized region-embeddings through mean-pooling:

$$E_{\text{cell}} = \frac{1}{j} \sum_{i=0}^{j-1} h_i \in \mathbb{R}^{d_{\text{model}}} \tag{21}$$

where h_i is the contextualized embedding vector for the i^{th} token.

Triplet loss calculation

For cell-type fine-tuning we use a standard triplet-loss formula. For each training step, the model sees three cells: 1) an anchor cell which may be of any cell-type, 2) a positive example which is of the same cell-type as the anchor, and 3) a negative example which is of another cell-type than the anchor cell. We pass each cell through the model and mean-pool token embeddings to obtain a single embedding to represent the entire cell; one for the anchor (a), the positive (p), and the negative (n). Loss for a single mini-batch is computed as such:

$$\mathrm{L(a,p,n)} = \max\{d(a_i, p_i) - d(a_i, n_i) + \mathrm{margin}, 0\} \tag{22}$$

where

$$d(x_i, y_i) = \mid x_i - y_i \mid_p \tag{23}$$

We use the torch module `torch.nn.TripletMarginLoss` with default values, $\mathrm{margin} = 1.0$, and $p = 2.0$.

Datasets for clustering evaluation

PBMC5k NextGEM v 1.1

For PBMC5K, we obtained raw matrix, peak, and barcode files from the 10X website: "https://www.10xgenomics.com/datasets/5-k-peripheral-blood-mononuclear-cells-pbm-cs-from-a-healthy-donor-next-gem-v-1-1-1-1-standard-2-0-0". These three files were converted to an `AnnData` object from the `scanpy` package.

Brain dataset

For the pre-annotated brain dataset, we utilize a multi-omic single-nucleus study of 191,890 nuclei in late-stage Alzheimer's disease (AD)[153]. Cells in this dataset were annotated using gene expression data to assign ground-truth labels to each cell. These labels were used for downstream clustering metrics evaluation.

Simulated

Evaluating model performance on real, pre-annotated datasets is subjected to the bias in the annotation procedure. This can cause misleading results according to inaccuracies in the labeling process. To that end, we supplemented our two datasets with a third, simulated dataset. Following a similar procedure to Chen et. al.[95], we generated a simulated scATAC-seq dataset using bulk-ATAC data from ENCODE. We first generate a peak by count matrix from 5 bulk-ATAC seq datasets: NK Cells (ENCSR305QTE), Memory B Cells (ENCSR610AQP), Naive B Cells (ENCSR685OFR), Dendritic Cells (ENCSR237VSF), CD4+ T cells (ENCSR841LHT), and CD8+ T cells (ENCSR476VJY). We leverage the

simulation code provided by the Pinello lab: https://github.com/pinellolab/scATAC-benchmarking/blob/master/Synthetic_Data/Simulate_scATAC/BoneMarrow/simulate_bonemarrow_depth.ipynb

PBMC dataset cell-type annotation

Because ground-truth labels are necessary for adequately assessing the clustering performance of cell-embeddings, we performed cell-type annotation on all three datasets. Each annotation was performed in an identical manner. To do so, we followed a very similar approach to the cell-type annotation approach described by LeRoy *et. al.* [106]. Briefly, we leveraged a pre-trained scEmbed model trained specifically on a high-quality blood dataset, Luecken2021[118]. Embeddings were generated for both the reference dataset (Luecken2021) and the query datasets (PBMC 1/5/10k). Then, using the shared latent space, we performed a K-nearest-neighbors (KNN) label transfer task. We used `scEmbed` from the `geniml` module on GitHub: "https://github.com/databio/geniml" and the `KNeighborsClassifier` from `sklearn.neighbors`. Due to the intrinsic sparsity of many detailed T-cell subtypes, we collapsed these rare variants into broader T-cell categories. This aggregation prevents overfragmentation during clustering, ensuring a more statistically robust and biologically meaningful representation of T-cell populations.

Labeling data with scVI

To label the brain dataset, we utilized the `scvi-tools` package[38]. Specifically, we used the `scanvi` model to perform semi-supervised cell-type annotation. We first created an `AnnData` object from the raw count matrix and then split the data into labeled and unlabeled sets. The labeled set consisted of 20% of the total cells, while the remaining 80% were unlabeled. We then trained the `scanvi` model on the labeled data for 400 epochs with default parameters. After training, we used the model to predict cell-type labels for the unlabeled cells. Finally, we combined the predicted labels with the original labeled data to obtain a complete set of cell-type annotations for the entire dataset.

Bulk training data selection

To generate a large dataset for fine-tuning on bulk data, we first started with all bed-files annotated with hg38 on BEDbase. Because Atacformer has a context window of 8,192 tokens, we next filtered down these bedfiles into a subset that could reasonable fit within this context window, subsampling tokens as necessary. We set the cutoff for number of regions in the bedfile to be 81,920 (10x the context window).

We tokenized the BED files that met this criteria and used them as input into the training pipeline, subsampling tokens from the file when necessary.

Spearman correlation

The spearman corrrelation can be computed as follows:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{24}$$

where $d_{\{i\}}$ is the difference between the two ranks of each observation and n is the number of observations. To compute the value, we leveraged the `scipy.stats` module and the `spearmanr` function.

Bulk ATAC-seq data imputation

To evaluate Atacformer's ability to impute missing regions in bulk ATAC-seq data, we curated distinct training and test sets from BEDbase. A significant portion of samples on BEDbase lacked explicit cell-line annotations, with the metadata field often marked as `null`. We discovered that for many of these cases, the cell line could be inferred by parsing the sample's free-text description. For instance, if a description contained "HEK293", we assigned that sample the "HEK293" cell-line label. Our test set was constructed exclusively from these samples where the cell line was inferred. The training set, in contrast, was composed of all samples from BEDbase that had explicit, non-null cell-line annotations. This strategy provided a natural train/test split for evaluating the model's performance on a realistic imputation task.

Multiome data processing

To curate a large multiome dataset, we downloaded and processed four datasets: three from the 10X genomics dataset repository and then the previously described Luecken2021 dataset[118]. The three 10X datasets were: 1) brain3k multiome "https://www.10xgenomics.com/datasets/frozen-human-healthy-brain-tissue-3-k-1-standard-2-0-0, 2) kidney22k "https://www.10xgenomics.com/datasets/human-kidney-cancer-nuclei-isolated-with-chromium-nuclei-isolation-kit-saltyez-protocol-and-10x-complex-tissue-dp-ct-sorted-and-ct-unsorted-1-standard", and 3) pbmc10k multiome "https://www.10xgenomics.com/datasets/10-k-human-pbm-cs-multiome-v-1-0-chromium-controller-1-standard-2-0-0". For each dataset, we downloaded the cell by feature matrix as a matrix-market file (`.mtx`), the barcodes as a `.txt` file, and the features as a `.tsv` file. We combined these files into an `.h5ad` file for each dataset using the `scanpy`, `pandas` and `scipy` python packages.

Each dataset was tokenized into the universe as previously described and used for training CRAFT.

CRAFT architecture

The CRAFT architecture closely follows the design of the CLIP model, which jointly trains two separate encoders to project different modalities into a shared latent space. In our implementation, we replaced the original CLIP encoders with domain-specific architectures: the ATAC encoder was substituted with the atacformer, a transformer-based model tailored for chromatin accessibility data, and the RNA encoder was replaced with geneformer, optimized for gene expression profiles:

```
# adapted from Radford2021 (Fig 3).
# gene_encoder  - Geneformer
# atac_encoder  - Atacformer
# R[n, h, w, c] - minibatch of aligned RNA-seq profiles
# A[n, l]       - minibatch of aligned ATAC-seq profiles
# W_i[d_i, d_e] - learned proj of rna profile to embed
# W_t[d_t, d_e] - learned proj of atac profile to embed
# t             - learned temperature parameter

# extract feature representations of each modality
R_f = gene_encoder(R) #[n, d_i]
A_f = atac_encoder(A) #[n, d_t]

# joint multimodal embedding [n, d_e]
R_e = l2_normalize(np.dot(R_f, W_i), axis=1)
A_e = l2_normalize(np.dot(A_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(R_e, A_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_r = cross_entropy_loss(logits, labels, axis=0)
loss_a = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_r + loss_a)/2
```

We train CRAFT starting with a pre-traiend Geneformer and Atacformer model. Namely, we use `Geneformer/gf-12L-30M-i2048` and `databio/atacformer-base-hg38"` respectively. We trained for 15 epochs using a linear learning rate scheduler with a maximum learning rate of $5e-5$.

pbmc5k dataset processing for RNA-imputation experiments

To prepare a dataset, we utilize closely follow the "RNA integration" tutorial offered by the SnapATAC2 documentation: "https://scverse.org/SnapATAC2/tutorials/annotation.html". Briefly, the pbmc5k dataset was imported from SnapATAC2, filtered, dimensionality-reduced, and subsequent cell-type annotation was performed using `scvi`.

CRAFT RNA decoder

Using `pytorch`, we built a small decoder to predict a cell's RNA-expression profile from its corresponding shared latent space ATAC embedding. We used a simple feedforward neural network with one hidden layer.

To obtain the shared latent space embedding from the ATAC data, we first encoded the cell's ATAC profile using an encoder network. The encoder consisted of a fully connected layer that projected the high-dimensional ATAC input into a lower-dimensional latent space, followed by a non-linear activation function (ReLU). The output of this encoder served as the input to the RNA decoder.

The overall architecture thus consisted of an ATAC encoder, which mapped the input ATAC features to a latent representation, and an RNA decoder, which predicted the RNA expression profile from this latent space.

Annotation of Atacformer universe for TSS distance and region type

To annotate the distance to the nearest TSS to each token in our vocabulary, we first downloaded the most recent comprehensive gene annotation (GTF) file from GENCODE("https://www.gencodegenes.org/human/release_38.html"). We filtered this file to obtain just the TSS annotations using common unix command-line tools like `awk` and `sort`. Next we leveraged `bedtools` to obtain distances to the nearest TSS. Specifically, we used the `bedbase closest` command with the `-t first` flag to ensure each region in our universe was only associated with one TSS.

Similarly, we downloaded the latest cCRE annotations from ENCODE screen (https://screen.encodeproject.org) for hg38. We utilized bedtools intersect to annotate each region with a discrete label (pELS, dELS, CTCF, etc).

H3K4me3 null distribution generation

To evaluate H3K4me3 signal enrichment in our icTSS regions across cell types, we first generated a null distribution representing the expected signal overlap in randomly selected genomic regions of comparable size. Specifically, we randomly sampled N regions from the Atacformer universe – where N equals the number of icTSS regions in each set (B cells and monocytes) – using standard Unix command-line utilities such as shuf. This sampling procedure was repeated 500 times to build a distribution of random region sets. For each set, we computed the mean coverage using bigWigAverageOverBed from the bigtools package[154], then averaged the resulting signal across all regions. The resulting distribution was plotted as a histogram, and the same statistic was computed for the true icTSS regions to quantify their enrichment relative to the null.