

DNA Sequence Analysis using Masked Language Modeling

M.Sc Thesis

By

Shiv Kumar Sharma



Discipline of Physics

Indian Institute of Technology Indore

May 2026

DNA Sequence Analysis using Masked Language Modeling

A Thesis

Submitted in partial fulfilment of the
requirements for the award of the degree
of

Master of Science

By

Shiv Kumar Sharma

2403151024



DISCIPLINE OF PHYSICS
INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2026

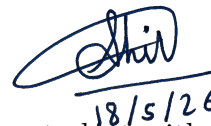


INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **DNA Sequence Analysis using Masked Language Modeling** in partial fulfillment of the requirements for the award of the degree of **MASTER OF SCIENCE** and submitted in the **DISCIPLINE OF PHYSICS**, Indian Institute of Technology Indore, is an authentic record of my own work carried out during the period from **July 2024** to **May 2026** under the supervision of **Dr. Sarika Jalan**, Professor, Indian Institute of Technology Indore.

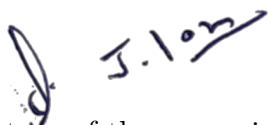
The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.


18/5/26

Signature of the student with date

(Shiv Kumar Sharma)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.



Signature of the supervisor with date

(Dr. Sarika Jalan)

Shiv Kumar Sharma has successfully given his M.Sc. Oral Examination held on **5 May 2026**.



Convener, DPGC

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to all those who have supported and encouraged me throughout the course of this thesis. This work would not have been possible without the guidance, motivation, and assistance of many individuals.

First and foremost, I am profoundly grateful to my supervisor, Dr. Sarika Jalan, for her constant guidance, insightful suggestions, patience, and unwavering support during every stage of this research. Her deep knowledge, constructive feedback, and encouragement greatly influenced both the direction and quality of this work. Working under her supervision has been an invaluable learning experience.

I would also like to sincerely thank the members of the Complex Systems Lab at IIT Indore for providing a stimulating and collaborative research environment. The numerous discussions, exchange of ideas, and shared experiences with the lab members contributed significantly to my understanding and growth as a researcher. I am equally thankful to all the faculty members of IIT Indore whose lectures, academic support, and mentorship helped build the strong foundation necessary for pursuing this research.

My sincere thanks go to my friends and peers for their encouragement, companionship, and support throughout this journey. Their motivation and positive presence made the challenges of research easier to navigate and the successes more meaningful to celebrate.

Finally, I owe my deepest gratitude to my family for their unconditional love, endless patience, sacrifices, and constant belief in me. Their encouragement and support have always been my greatest source of strength and inspiration. This achievement would not have been possible without them.

To everyone who has contributed to this journey, directly or indirectly, I extend my sincere and heartfelt thanks.

Abstract

This thesis investigates the application of transformer-based deep learning models for DNA sequence prediction and contextual genomic analysis under varying temperature conditions. Inspired by advances in natural language processing, DNA sequences are modeled as structured symbolic patterns in which nucleotide relationships can be learned through contextual attention mechanisms. The work primarily focuses on masked nucleotide prediction, sequence surprisal analysis, and comparative genomic behavior across different environmental conditions.

A masked language modeling framework based on the BERT architecture was developed to predict masked nucleotides from surrounding sequence context. The model was trained on genomic datasets corresponding to different temperature conditions, enabling the study of how environmental variations influence sequence predictability and contextual structure. The training pipeline incorporated nucleotide tokenization, dynamic masking, positional encoding, and attention-based learning to capture long-range dependencies within genomic sequences.

To analyze contextual uncertainty, surprisal values were computed using the negative logarithm of predicted nucleotide probabilities. Lower surprisal values corresponded to highly predictable and conserved regions, while higher values indicated contextual variability. Comparative analysis revealed temperature-dependent variations in prediction accuracy, surprisal distributions, and conserved token structures across datasets. Additional investigation of correctly predicted positions and common sequence regions further highlighted contextual similarities and differences between genomic samples.

The results demonstrate that transformer architectures can effectively capture hidden contextual patterns and long-range dependencies in DNA sequences. The proposed framework provides a computational approach for genomic sequence analysis using attention-driven models and contributes toward the integration of machine learning techniques with computational genomics.

Contents

Candidate's Declaration	ii
Acknowledgement	iii
Abstract	iv
List of Figures	viii
1 Introduction	1
1.1 Motivation	2
2 Literature Review	4
2.1 Traditional Computational Approaches in Genomics	4
2.2 Machine Learning in Genomic Sequence Analysis	4
2.3 Deep Learning Approaches for Biological Sequences	5
2.4 Transformer Architectures and Self-Attention Mechanisms	5
2.5 Transformer-Based Models for Genomic Sequences	6
2.6 Masked Language Modeling in Genomics	7
2.7 Surprisal and Information-Theoretic Sequence Analysis	7
2.8 Limitations of Existing Work	8
2.9 Research Gap and Problem Formulation	8
3 Dataset and Statistical Analysis	10
3.1 Genomic Dataset Description	10
3.2 Temperature-Based Dataset Organization	10
3.3 Sequence Cleaning and Preparation	11
3.4 Whole-Genome Statistical Analysis	12
3.4.1 Base Pair Frequency Analysis	12
3.4.2 Mutation Map Analysis	14
3.4.3 Transition and Transversion Analysis	16
3.5 Gene-Level Analysis	17
4 Transformer-Based Prediction Framework	20
4.1 Transformer Architecture	20
4.1.1 Encoder Architecture	22

4.1.2	Multi-Head Self-Attention	23
4.1.3	Positional Encoding	24
4.2	Self-Attention Mechanism	25
4.2.1	Scaled Dot-Product Attention	25
4.2.2	Biological Interpretation of Attention Weights	25
4.3	BERT-Based Masked Language Modeling Framework	26
4.3.1	Bidirectional Contextual Learning	26
4.3.2	Masked Language Modeling (MLM)	27
4.4	Model Configuration	28
4.4.1	Hidden Embedding Dimension	29
4.4.2	Number of Attention Heads	29
4.4.3	Number of Encoder Layers	29
4.4.4	Batch Size	30
4.4.5	Learning Rate and Optimization Schedule	30
5	Methodology	32
5.1	Consistent Window Analysis	32
5.1.1	HSP-16.2 Consistent Window Analysis	33
5.1.2	ISY-1 Consistent Window Analysis	34
5.1.3	HSF-1 Consistent Window Analysis	35
5.2	Metadata Construction	36
5.2.1	Competitive Positions and Surprisal Distribution	36
5.3	Surprisal-Based Prediction Analysis	38
5.3.1	Training and Testing Strategy	38
5.3.2	Definition of Surprisal	39
5.3.3	<i>ISY-1</i> Gene Surprisal Analysis	40
5.3.4	<i>HSF-1</i> Gene Suprisal Analysis	44
5.3.5	Interpretation of Surprisal Profiles	48
6	Results and Discussion	49
6.1	Overview	49
6.2	Sliding-Window Consistency Analysis	49
6.2.1	HSF1 — Large Gene Analysis	49
6.2.2	HSP-16.2 — Short Gene Analysis	50
6.2.3	ISY-1 — Intermediate Gene Analysis	51
6.3	Comparative Summary of Window Robustness	51
6.4	Surprisal Analysis at Competitive Positions	52
6.5	Biological Significance of Prediction Patterns	53
7	Conclusion	54

A	Implementation	55
A.1	Overview of the Pipeline	55
A.2	Environment Setup and Reproducibility	55
A.3	Data Ingestion	56
A.4	Tokeniser	56
A.5	Dataset and Data Collator	57
A.6	Model Architecture	58
A.7	Training Loop	59
A.8	Evaluation Metrics	60
A.8.1	Per-Position Surprisal	60
A.8.2	Unified Metadata (Probabilities + Surprisal)	61
A.8.3	Masked Prediction Accuracy	63
A.9	Configuration and Output	64
A.10	Summary	66
	References	67

List of Figures

3.1	Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R1–R4 at 15 ⁰	14
3.2	Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R5–R8 at 20 ⁰	15
3.3	Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R9–R12 at 25 ⁰	15
3.4	The Transition/Transversion (Ts/Tv) ratio for all sequenced samples.	16
3.5	Ts/Tv ratio for HSF-1 gene.	17
3.6	Ts/Tv ratio for ISY-1 gene.	18
3.7	Ts/Tv ratio for HSP-16.2 gene.	19
4.1	Schematic architecture of the Transformer encoder.	22
5.1	Consistent window analysis for the HSP-16.2 gene.	34
5.2	Consistent window analysis for the ISY-1 gene.	35
5.3	Consistent window analysis for the HSF-1 gene.	36
5.4	Surprisal values for competitive nucleotide positions in HSF1 sample.	38
5.5	first 10% surprisal profiles for ISY-1 samples 1–4.	40
5.6	First 10% surprisal profiles for ISY-1 samples 5–8.	41
5.7	First 10% surprisal profiles for ISY-1 samples 9–12.	41
5.8	Last 10% surprisal profiles for ISY-1 samples 1–4.	42
5.9	Last 10% surprisal profiles for ISY-1 samples 5–8.	42
5.10	Last 10% surprisal profiles for ISY-1 samples 9–12.	43
5.11	First 10% surprisal profiles for HSF-1 samples 1–4.	44
5.12	First 10% surprisal profiles for HSF-1 samples 5–8.	45
5.13	First 10% surprisal profiles for HSF-1 samples 9–12.	45
5.14	Last 10% surprisal profiles for HSF-1 samples 1–4.	46
5.15	Last 10% surprisal profiles for HSF-1 samples 5–8.	46
5.16	Last 10% surprisal profiles for HSF-1 samples 9–12.	47

Chapter 1

Introduction

DNA sequences contain the fundamental biological information responsible for the structure, functioning, and regulation of living organisms. Understanding the hidden patterns and dependencies within genomic sequences is one of the central challenges in computational biology and bioinformatics. With the rapid growth of genomic databases and sequencing technologies, there is an increasing need for computational approaches capable of extracting meaningful information from large-scale biological sequence data. In recent years, machine learning and deep learning techniques have emerged as powerful tools for analyzing genomic structures, identifying sequence patterns, and modeling biological relationships that are often difficult to capture using traditional statistical methods.

Genomic sequences can be naturally interpreted through the lens of language modeling, drawing a strong parallel with methods used in natural language processing. In human language, the meaning and grammatical correctness of a word depend strongly on its surrounding context. Similarly, nucleotides within DNA are not independently distributed; their arrangement is governed by coding regions, regulatory elements, structural constraints, and long-range biological interactions. These contextual dependencies make genomic data highly suitable for sequence modeling approaches that can learn complex and high-order patterns directly from data. Deep learning architectures, in particular, provide the capability to represent sequences in high-dimensional feature spaces, allowing simultaneous learning of local motifs and long-range contextual relationships.

Among modern deep learning methods, transformer-based architectures have shown exceptional performance in sequence modeling tasks due to their self-attention mechanisms. Self-attention enables each position in a sequence to selectively focus on other relevant positions and learn contextual importance weights during training. Unlike recurrent neural networks or convolutional models, transformers do not rely heavily on sequential processing or locality assumptions, making them especially effective for capturing long-range dependencies commonly present in genomic sequences. Within

this framework, the Masked Language Model (MLM) objective trains the model to predict masked nucleotides using surrounding sequence information, thereby learning bidirectional contextual representations of DNA sequences.

In this thesis, a transformer-based masked language modeling framework is developed for DNA sequence prediction and contextual genomic analysis. The study focuses on predicting masked nucleotides, analyzing sequence surprisal, and investigating contextual variations across genomic datasets corresponding to different temperature conditions. Surprisal analysis, based on the negative logarithm of predicted nucleotide probabilities, is used to quantify sequence predictability and identify conserved or irregular regions within genomic sequences. Comparative analysis of prediction accuracy, conserved token structures, and contextual similarities across multiple datasets is also performed to examine the influence of environmental conditions on genomic behavior.

The results obtained in this work demonstrate that transformer architectures are capable of effectively capturing hidden contextual patterns and long-range dependencies in DNA sequences. The proposed framework highlights the potential of attention-based deep learning methods for biological sequence analysis and contributes toward the growing integration of machine learning techniques with computational genomics. The study further provides a foundation for future investigations involving large-scale genomic modeling, contextual sequence analysis, and environmentally dependent genomic characterization.

1.1 Motivation

Genomic sequences are continuously influenced by environmental conditions, among which temperature plays a significant role in determining nucleotide stability, replication fidelity, mutation rates, and overall genomic organization. Variations in temperature can alter the structural and contextual properties of DNA, leading to changes in sequence predictability and conservation patterns. Understanding how genomic sequences respond to such environmental stress is important for studying biological adaptation, evolutionary dynamics, and the mechanisms through which organisms maintain genomic integrity under changing conditions.

A key challenge in computational genomics is identifying both conserved and variable regions within DNA sequences. Conserved regions often correspond to biologically important functional elements that remain stable across conditions, while

variable regions may indicate mutation-prone or environmentally responsive segments of the genome. Analyzing these patterns provides insight into how genomes preserve essential information while simultaneously adapting to external influences. Once such contextual patterns are identified, sequence prediction and masked-token prediction become valuable computational tools for modeling genomic behavior and understanding sequence-level dependencies.

Recent transformer-based genomic models such as DNABERT and Nucleotide Transformer have demonstrated the effectiveness of masked language modeling for learning DNA sequence representations. These models successfully capture motifs, contextual relationships, and local dependencies through k-mer tokenization and attention-based learning. However, most existing approaches are trained on general genomic data without incorporating environmental variables such as temperature. As a result, they are limited in their ability to model temperature-driven changes in sequence composition, contextual variability, mutation behavior, or conserved motifs across different conditions.

From a computational perspective, transformer architectures provide an efficient framework for genomic sequence analysis due to their ability to capture bidirectional contextual information. Unlike traditional sequential models, transformers analyze each nucleotide in relation to the entire surrounding sequence through self-attention mechanisms. The Masked Language Modeling (MLM) objective further strengthens this capability by repeatedly masking nucleotides and training the model to reconstruct them using contextual information. This process enables the model to learn the underlying structural grammar and statistical organization of genomic sequences.

Motivated by these observations, this work aims to investigate transformer-based DNA sequence prediction and contextual analysis under varying temperature conditions. By combining masked nucleotide prediction, surprisal analysis, and comparative sequence analysis, the study seeks to understand how environmental variations influence genomic predictability and contextual structure, while also exploring the capability of attention-based models to capture biologically meaningful sequence patterns.

Chapter 2

Literature Review

2.1 Traditional Computational Approaches in Genomics

Early computational genomics methods primarily relied on statistical and alignment-based approaches for analyzing biological sequences. Sequence alignment algorithms such as BLAST and Smith-Waterman were widely used to identify regions of similarity, evolutionary conservation, and homologous structures between genomic sequences. Although these methods are effective for local similarity searches, they often struggle to capture complex contextual dependencies and large-scale sequence structures.

Probabilistic approaches such as Markov models and Hidden Markov Models (HMMs) were later introduced to model nucleotide dependencies statistically. Markov models estimate the probability of a nucleotide based on preceding nucleotides and are useful for modeling local sequence behavior. Hidden Markov Models further extend this framework by incorporating hidden states representing biological properties such as coding and non-coding regions. These methods achieved success in gene prediction, motif discovery, and promoter analysis; however, they remain limited in their ability to capture long-range interactions and higher-order contextual information present within genomic sequences.

As genomic datasets increased in size and complexity, traditional approaches became computationally expensive and less effective for modeling large contextual dependencies. This motivated the adoption of machine learning techniques capable of automatically learning sequence representations from data.

2.2 Machine Learning in Genomic Sequence Analysis

Machine learning methods introduced data-driven approaches for analyzing genomic sequences without relying heavily on manually engineered biological features. Early machine learning models such as Support Vector Machines (SVMs), Random Forests, and Decision Trees were widely applied for classification tasks including promoter prediction, splice-site detection, protein-binding analysis, and mutation classification.

These approaches generally required handcrafted feature extraction methods such as k-mer frequencies, GC-content analysis, sequence motifs, or physicochemical descriptors.

Although machine learning methods improved predictive performance compared to traditional statistical models, their effectiveness strongly depended on the quality and relevance of manually designed features. Furthermore, these approaches lacked the capability to automatically capture hierarchical and contextual relationships within DNA sequences.

The limitations of feature engineering motivated the use of deep learning methods, which can directly learn meaningful representations from raw biological sequences.

2.3 Deep Learning Approaches for Biological Sequences

Deep learning methods have significantly transformed computational genomics by enabling automatic extraction of complex sequence patterns from large-scale genomic data. Convolutional Neural Networks (CNNs) were among the earliest deep learning architectures applied to genomic analysis. CNNs are effective at identifying local motifs and short-range sequence patterns by learning convolutional filters directly from nucleotide sequences. They have been successfully applied to transcription factor binding prediction, enhancer detection, and chromatin accessibility analysis.

Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks were later introduced to model sequential dependencies within biological sequences. These architectures process sequences sequentially and maintain hidden states that capture contextual information from previous positions. LSTM networks improved the modeling of long-range dependencies compared to standard RNNs and achieved promising results in sequence classification and genomic annotation tasks.

Despite their success, recurrent architectures suffer from several limitations, including difficulty in parallel computation, vanishing gradient problems, and inefficiency in modeling very long sequences. CNN-based methods also face challenges in capturing global contextual relationships because convolution operations primarily focus on local receptive fields. These limitations motivated the exploration of transformer-based architectures for biological sequence modeling.

2.4 Transformer Architectures and Self-Attention Mechanisms

Transformer architectures were originally introduced for natural language processing tasks and have since revolutionized sequence modeling through the use of self-attention mechanisms. Unlike recurrent models, transformers process all sequence elements simultaneously, enabling efficient parallel computation and improved modeling of long-range dependencies.

The core component of the transformer architecture is the self-attention mechanism, which allows each position in a sequence to attend to all other positions and learn contextual relationships dynamically. For a sequence representation consisting of query (Q), key (K), and value (V) vectors, attention is computed as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d represents the dimensionality of the key vectors. The self-attention mechanism enables the model to learn the relative importance of different sequence positions while constructing contextual representations.

Transformers also incorporate positional encoding to preserve sequential order information, since the architecture itself does not process tokens sequentially. Multi-head attention further improves the learning capability by allowing the model to capture multiple contextual relationships simultaneously.

Due to their ability to model long-range dependencies and contextual structures efficiently, transformer architectures have become highly suitable for genomic sequence analysis.

2.5 Transformer-Based Models for Genomic Sequences

Inspired by the success of transformers in language modeling, several transformer-based architectures have been developed for biological sequence analysis. These models treat DNA sequences as symbolic languages in which nucleotides or k-mers act as tokens analogous to words in human language.

DNABERT is one of the earliest transformer-based genomic language models that applied the Bidirectional Encoder Representations from Transformers (BERT) framework to DNA sequences. DNABERT uses k-mer tokenization combined with masked language modeling to learn contextual genomic representations. The model demonstrated strong performance in promoter prediction, transcription factor binding analysis, and genomic feature extraction.

Subsequent models such as GenomeBERT and Nucleotide Transformer further expanded transformer-based genomic modeling by training on larger genomic datasets and improving contextual representation learning. These approaches showed that transformer architectures are capable of learning biologically meaningful motifs, sequence structures, and contextual dependencies directly from raw genomic data.

However, most existing transformer-based genomic models are trained on general sequence datasets without explicitly incorporating environmental or experimental conditions such as temperature. Consequently, these models remain limited in their ability to capture environmentally dependent variations in sequence predictability,

mutation behavior, or contextual conservation.

2.6 Masked Language Modeling in Genomics

Masked Language Modeling (MLM) is a self-supervised learning objective widely used in transformer-based architectures such as BERT. In MLM, selected tokens within a sequence are randomly masked, and the model is trained to predict the masked tokens using surrounding contextual information.

For genomic sequences, the MLM objective enables the model to learn bidirectional contextual relationships between nucleotides. Given a sequence

$$[X = (x_1, x_2, \dots, x_n),]$$

the model estimates the conditional probability

$$[P(x_i \mid \text{context}),]$$

where the context consists of neighboring nucleotides surrounding the masked position. The model is trained by minimizing the cross-entropy loss between predicted nucleotide probabilities and the true nucleotide labels.

Masked language modeling is particularly useful for genomic analysis because it encourages the model to learn sequence grammar, contextual organization, and hidden dependencies without requiring labeled biological annotations. The learned probability distributions also provide meaningful estimates of nucleotide predictability and uncertainty, which can be further analyzed using information-theoretic measures such as surprisal.

2.7 Surprisal and Information-Theoretic Sequence Analysis

Information theory provides useful tools for quantifying uncertainty and predictability in sequential data. One such measure is surprisal, defined as the negative logarithm of the probability assigned to an observed event. For genomic sequences, surprisal can be expressed as

$$[S(x_i) = -\log P(x_i \mid \text{context}),] \tag{2.1}$$

where $P(x_i \mid \text{context})$ represents the probability predicted by the model for the nucleotide at position i .

Low surprisal values indicate highly predictable nucleotides that are strongly constrained by their surrounding context, while high surprisal values correspond to irregular or less predictable regions. Surprisal analysis therefore provides insight into conserved structures, contextual variability, and sequence uncertainty.

In natural language processing, surprisal has been widely used for analyzing linguistic complexity and contextual predictability. Its application to genomic sequences

remains relatively unexplored, particularly in the context of transformer-based biological language models and environmentally dependent genomic analysis.

2.8 Limitations of Existing Work

Although transformer-based genomic models have achieved promising performance in sequence representation learning and prediction tasks, several important limitations remain. Most existing models are trained on generic genomic datasets without considering environmental variables such as temperature, stress conditions, or mutation-inducing factors. As a result, they are unable to model temperature-driven contextual variations and environmental influences on genomic predictability.

Furthermore, current approaches primarily focus on sequence classification or motif prediction while providing limited analysis of contextual uncertainty and sequence-level predictability. Comparative analysis of conserved and variable regions across different environmental conditions is also relatively underexplored. Existing studies rarely combine masked language modeling with surprisal-based analysis to investigate contextual genomic behavior under varying experimental conditions.

Another limitation is the lack of interpretability regarding how prediction accuracy and contextual patterns change across temperature-dependent datasets. Understanding these variations is important for studying genomic stability, mutation behavior, and adaptive biological mechanisms.

2.9 Research Gap and Problem Formulation

The literature review highlights the growing importance of transformer-based architectures for genomic sequence analysis while also revealing significant gaps in environmentally dependent genomic modeling. Existing transformer models effectively learn contextual representations from DNA sequences; however, they generally lack mechanisms to analyze temperature-driven variations in sequence predictability and contextual conservation.

Motivated by these limitations, the present work aims to develop a transformer-based masked language modeling framework for DNA sequence prediction and contextual genomic analysis across multiple temperature conditions. The study focuses on predicting masked nucleotides, analyzing sequence surprisal, identifying conserved and variable regions, and comparing contextual sequence behavior under different environmental settings.

The primary research questions addressed in this work include:

- Can transformer-based masked language models effectively predict nucleotides in genomic sequences under varying temperature conditions?

- How does temperature influence contextual sequence predictability and surprisal distributions?
- Which regions of the genomic sequences remain conserved or variable across different environmental conditions?
- Can surprisal analysis reveal biologically meaningful contextual structures and sequence irregularities? ““

Addressing these questions provides a computational framework for studying environmentally dependent genomic behavior using transformer-based deep learning architectures and contributes toward the integration of modern sequence modeling techniques with computational genomics.

Chapter 3

Dataset and Statistical Analysis

3.1 Genomic Dataset Description

The datasets used in this study consist of genomic DNA sequences obtained under different temperature conditions to investigate temperature-dependent variations in genomic composition, mutation behavior, and sequence conservation. The analysis was performed using genomic sequences from the organism *Caenorhabditis elegans*, a widely studied model organism in genetics and molecular biology.

The genomic sequences were stored in FASTA format, where each sequence consists of the four primary nucleotide bases: Adenine (A), Thymine (T), Cytosine (C), and Guanine (G). The datasets were selected to enable comparative analysis across different environmental conditions while preserving biologically meaningful sequence structure.

The *Caenorhabditis elegans* genome contains approximately 20,000 genes with an average gene length of nearly 2.5 kb. This relatively compact and well-annotated genome makes it suitable for genomic sequence analysis and investigation of environmentally sensitive genomic regions.

The datasets were categorized into three temperature groups corresponding to different environmental conditions:

- 15°C
- 20°C
- 25°C

These temperature-based datasets were used to analyze sequence composition, mutation characteristics, conserved regions, and possible temperature-dependent genomic variation.

3.2 Temperature-Based Dataset Organization

To maintain consistency and simplify comparative analysis, the genomic datasets were renamed using a structured notation based on temperature category and sample index.

The files were organized as follows:

- T1-S1 to T1-S4 : sequences corresponding to 15°C
- T2-S1 to T2-S4 : sequences corresponding to 20°C
- T3-S1 to T3-S4 : sequences corresponding to 25°C

Here, ‘T’ denotes the temperature group, while ‘S’ represents the sample index within that group. This organization enabled systematic comparison of corresponding genomic samples across different environmental conditions.

For comparative analysis, corresponding samples from different temperature groups were analyzed together. For example:

- T1-S1 corresponds to the Sample-1 sequence at 15°C
- T2-S1 corresponds to the Sample-1 sequence at 20°C
- T3-S1 corresponds to the Sample-1 sequence at 25°C

Similar comparisons were performed for all remaining sample groups. This organization enabled direct investigation of sequence conservation, nucleotide variation, and mutation trends across different temperature conditions.

3.3 Sequence Cleaning and Preparation

Before performing statistical analysis, all genomic datasets underwent preprocessing to ensure consistency and biological validity.

The genomic sequences were initially extracted from FASTA files. During FASTA processing, all header lines beginning with the “>” symbol were removed while retaining only the nucleotide sequence information. The extracted sequences were converted into continuous uppercase nucleotide strings to maintain uniform formatting across all datasets.

Invalid or ambiguous nucleotide symbols that did not belong to the standard DNA alphabet[A, T, C, G] were removed during preprocessing. Sequences containing undefined symbols or corrupted regions were cleaned or excluded depending on the extent of irregularity.

Following sequence cleaning, the datasets were prepared for comparative statistical analysis of nucleotide composition, mutation behavior, and genomic variation across different temperature conditions.

3.4 Whole-Genome Statistical Analysis

The initial stage of analysis focused on the complete genomic sequences in order to investigate large-scale sequence composition, mutation behavior, and genomic stability across varying temperature conditions.

3.4.1 Base Pair Frequency Analysis

Base pair frequency analysis was performed to study the nucleotide composition of the genomic datasets. The relative frequencies of Adenine (A), Thymine (T), Cytosine (C), and Guanine (G) were computed for sequences corresponding to each temperature group.

This analysis provided insight into global genomic composition and nucleotide distribution across different environmental conditions. Comparative analysis of nucleotide frequencies enabled identification of conserved compositional patterns as well as possible temperature-dependent variation in sequence organization.

The base pair frequency distributions across the datasets remained largely stable, indicating that large-scale genomic composition was broadly conserved under varying temperature conditions.

Table 3.1: Dinucleotide transition frequency counts across datasets on 15⁰ temperature.

X_i	X_{i+1}	S_1	S_2	S_3	S_4
A	A	13573634	13573037	13573309	13573243
A	C	4847243	4846989	4847092	4847041
A	G	5076546	5076310	5076431	5076368
A	T	8876955	8876416	8876757	8876543
C	A	6205280	6204869	6205034	6204879
C	C	3365930	3365550	3365819	3365852
C	G	3132003	3131785	3131907	3131772
C	T	5080519	5080199	5080331	5080295
G	A	6232228	6231997	6232196	6232082
G	C	3341131	3340766	3340973	3340867
G	G	3345558	3345278	3345468	3345415
G	T	4842069	4841919	4842023	4841838
T	A	6363237	6362850	6363051	6362992
T	C	6229428	6229098	6229207	6229038
T	G	6206878	6206586	6206853	6206646
T	T	13576694	13576170	13576632	13576195

Table 3.2: Dinucleotide transition frequency counts across datasets on 20⁰ temperature.

X_i	X_{i+1}	S_5	S_6	S_7	S_8
A	A	13573043	13572982	13573175	13572834
A	C	4846886	4846894	4846939	4846911
A	G	5076206	5076214	5076280	5076159
A	T	8876282	8876450	8876488	8876296
C	A	6204725	6204722	6204779	6204737
C	C	3365573	3365487	3365571	3365576
C	G	3131678	3131629	3131665	3131635
C	T	5080056	5080169	5080106	5080158
G	A	6231911	6231859	6231907	6231860
G	C	3340690	3340736	3340703	3340678
G	G	3345198	3345272	3345544	3345479
G	T	4841764	4841783	4841854	4841749
T	A	6362739	6362978	6363022	6362770
T	C	6228883	6228890	6228908	6228941
T	G	6206480	6206534	6206518	6206492
T	T	13576271	13576239	13576307	13576062

Table 3.3: Dinucleotide transition frequency counts across datasets on 25⁰ temperature

X_i	X_{i+1}	S_9	S_{10}	S_{11}	S_{12}
A	A	13573023	13573039	13573220	13573241
A	C	4846939	4846933	4846931	4847084
A	G	5076228	5076300	5076285	5076291
A	T	8876353	8876480	8876537	8876450
C	A	6204779	6204768	6204807	6204902
C	C	3365553	3365605	3365669	3365544
C	G	3131697	3131700	3131669	3131712
C	T	5080149	5080224	5080132	5080305
G	A	6231898	6231979	6231974	6232027
G	C	3340722	3340777	3340764	3340766
G	G	3345264	3345462	3345731	3345315
G	T	4841883	4841895	4841850	4841835
T	A	6362844	6362967	6362973	6362897
T	C	6228964	6228982	6228913	6229069
T	G	6206577	6206650	6206633	6206624
T	T	13576506	13576354	13576396	13576550

3.4.2 Mutation Map Analysis

To investigate sequence-level variation across temperature conditions, mutation map analysis was performed between corresponding genomic samples and reference sequences.

Pairwise sequence comparisons were carried out to identify nucleotide substitutions occurring across different temperature groups. The mutation maps provided a positional representation of sequence variation and enabled visualization of conserved regions, mutation-prone segments, and environmentally sensitive genomic positions.

The analysis also enabled the identification of genomic regions exhibiting relatively stable sequence structure across all temperature conditions.

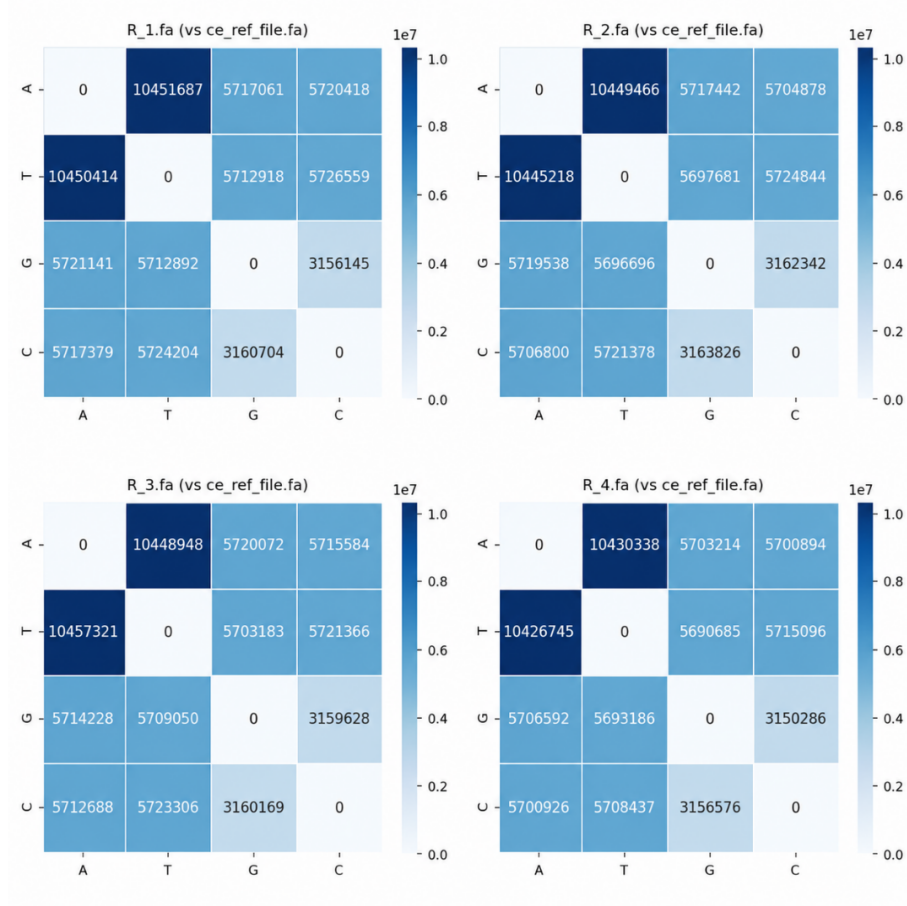


Figure 3.1: Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R1–R4 at 15°C.

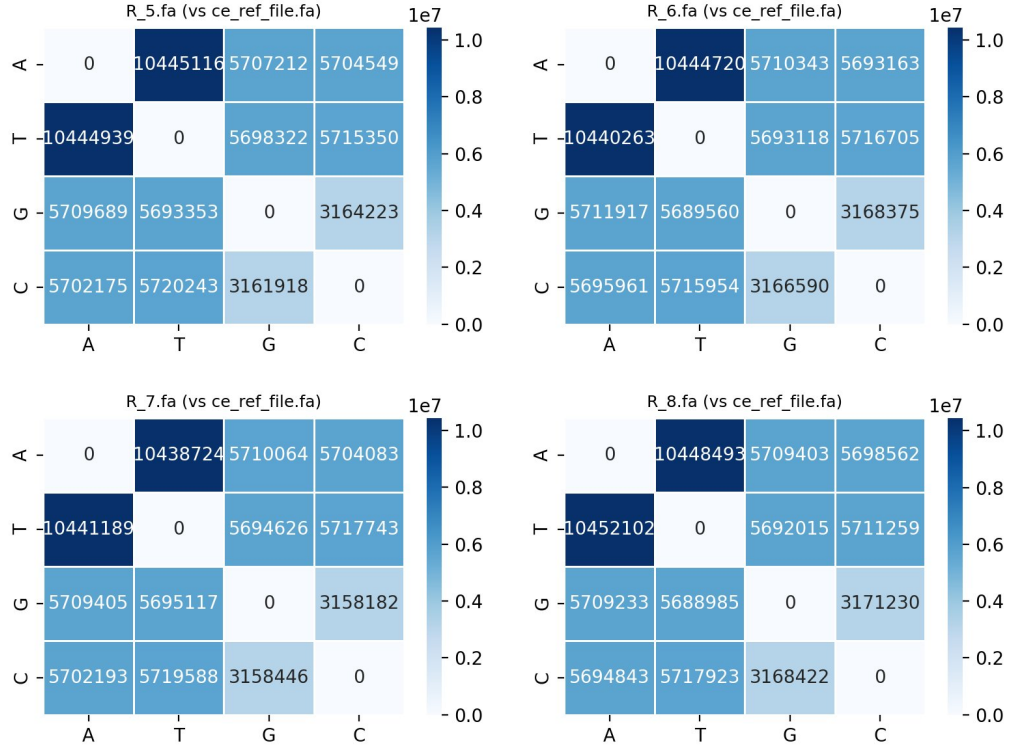


Figure 3.2: Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R5–R8 at 20⁰.

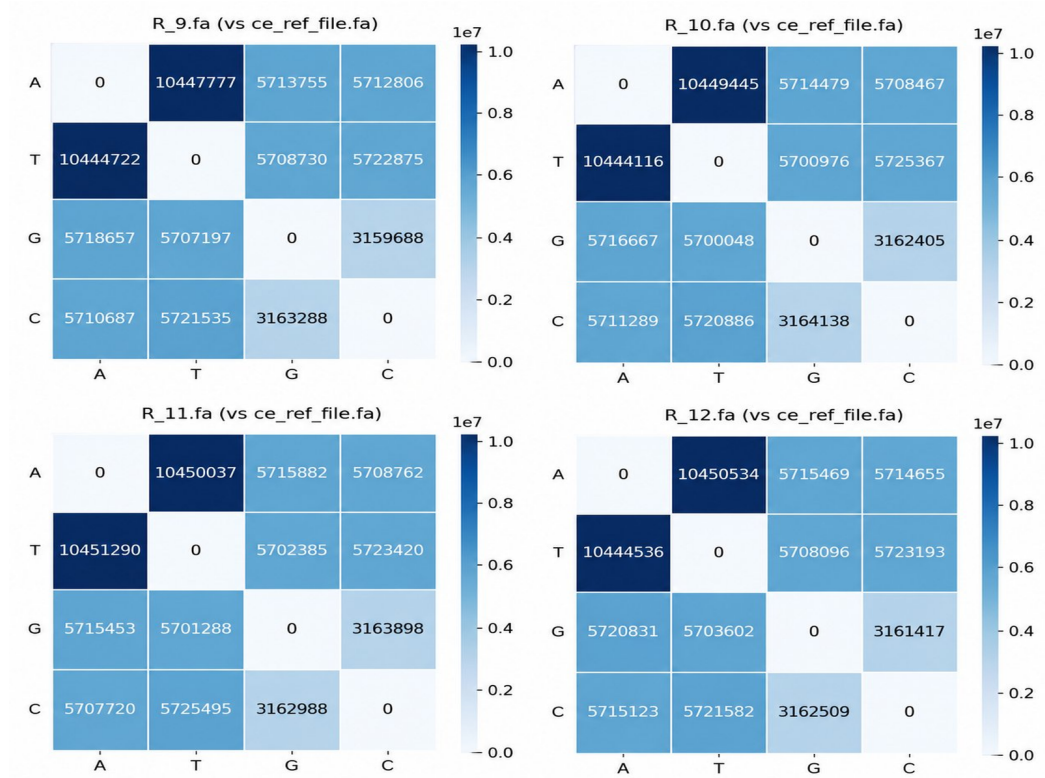


Figure 3.3: Heatmap representation of nucleotide substitution frequencies between the reference genome and datasets R9–R12 at 25⁰.

3.4.3 Transition and Transversion Analysis

All nucleotide substitutions identified during mutation map analysis were classified into two categories: transitions and transversions.

Transitions correspond to substitutions between nucleotides belonging to the same chemical group:

$$[A \leftrightarrow G][C \leftrightarrow T] \quad (3.1)$$

where A and G are purines, while C and T are pyrimidines.

Transversions correspond to substitutions between purines and pyrimidines:

$$[A \leftrightarrow C, ; A \leftrightarrow T, ; G \leftrightarrow C, ; G \leftrightarrow T] \quad (3.2)$$

To characterize mutation tendencies, transition-transversion (Ts/Tv) ratio analysis was performed for all temperature groups. Since transitions generally involve lower structural disruption compared to transversions, the Ts/Tv ratio serves as an important indicator of mutation behavior and genomic stability.

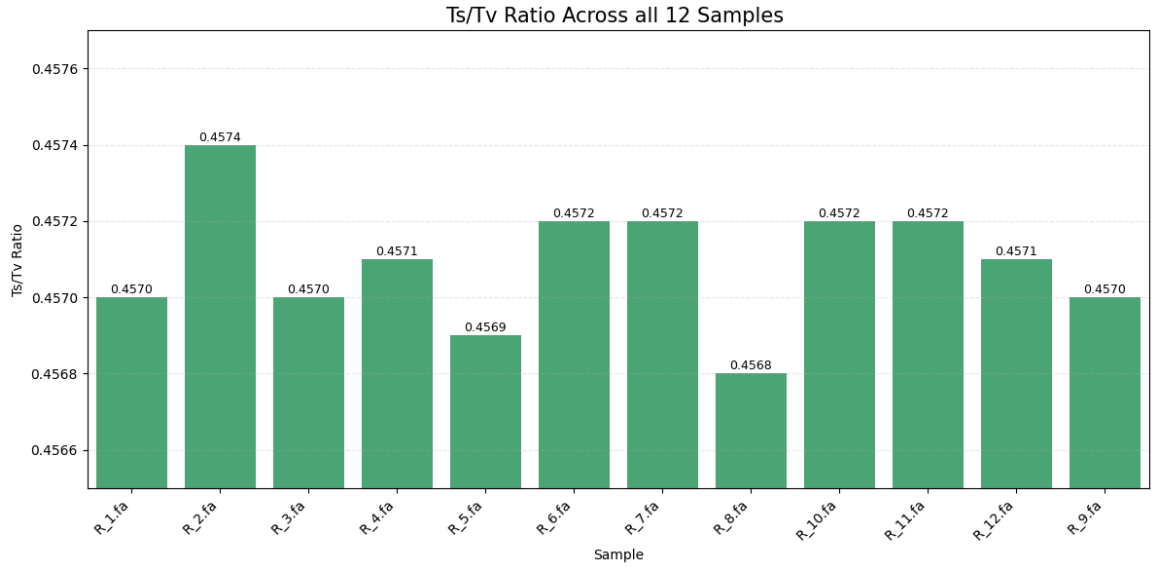


Figure 3.4: Bar plot of the Transition/Transversion (Ts/Tv) ratio for all sequenced samples. The overall ratio remains highly consistent (0.457).

Comparative analysis revealed that the Ts/Tv patterns remained relatively consistent across all temperature conditions, indicating the absence of strong temperature-driven mutation bias within the analyzed genomic regions. This observation suggested that large-scale genomic composition and mutation characteristics remained broadly conserved despite environmental variation.

Although the whole-genome analysis revealed relatively stable global mutation behavior, temperature-dependent effects are often subtle and localized within specific

functional genes rather than uniformly distributed throughout the genome. Consequently, the analysis was shifted from complete genomic sequences toward gene-level investigation in order to study environmentally sensitive genomic regions in greater detail.

3.5 Gene-Level Analysis

Following the whole-genome statistical analysis, the study was extended toward gene-level investigation to analyze functionally important genomic regions associated with stress response and regulatory behavior.

Gene-level analysis enables investigation of coding regions, regulatory motifs, and environmentally sensitive genomic elements that may exhibit subtle temperature-dependent variation not easily observable at the whole-genome level.

The present work primarily focused on three biologically significant genes associated with stress response and genomic regulation:

HSF-1

HSF-1 (Heat Shock Factor-1) acts as a master regulator of the heat shock response pathway. It activates molecular chaperone genes, including members of the Heat Shock Protein (HSP) family, and plays a critical role in maintaining proteostasis and cellular stress resistance under elevated temperature conditions.

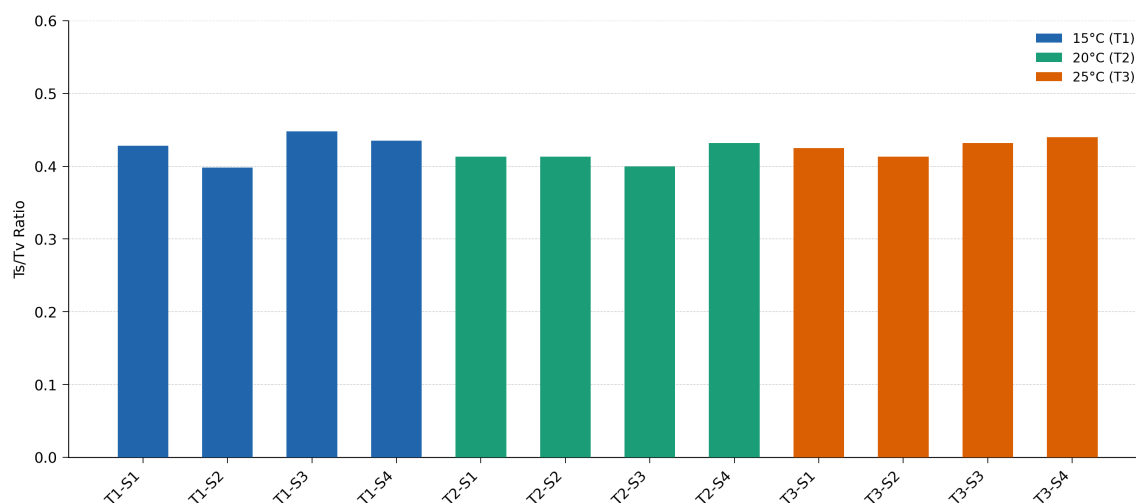


Figure 3.5: HSF-1 Ratios were tightly clustered between 0.40 and 0.45 across all temperature groups, with minimal variation.

ISY-1

ISY-1 is involved in pre-mRNA splicing and accurate regulation of gene expression. Due to its regulatory role and comparatively stable behavior, ISY-1 serves as an important control or reference gene for comparative genomic analysis under varying environmental conditions.

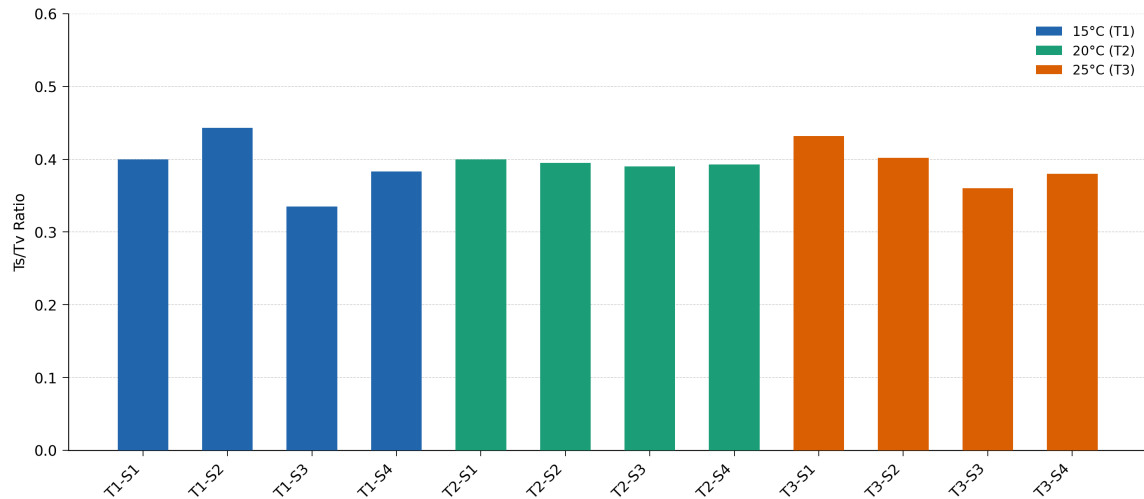


Figure 3.6: ISY-1 Ts/Tv ratios ranged from 0.34 to 0.44, with the 15°C group showing the most variability. The 20°C and 25°C groups were relatively stable.

HSP-16.2

HSP-16.2 is a small heat shock protein associated with environmental stress protection mechanisms. It functions as a molecular holdase chaperone and prevents protein aggregation during thermal stress conditions. The gene is strongly associated with heat-induced cellular response and therefore provides an important target for temperature-dependent genomic analysis.

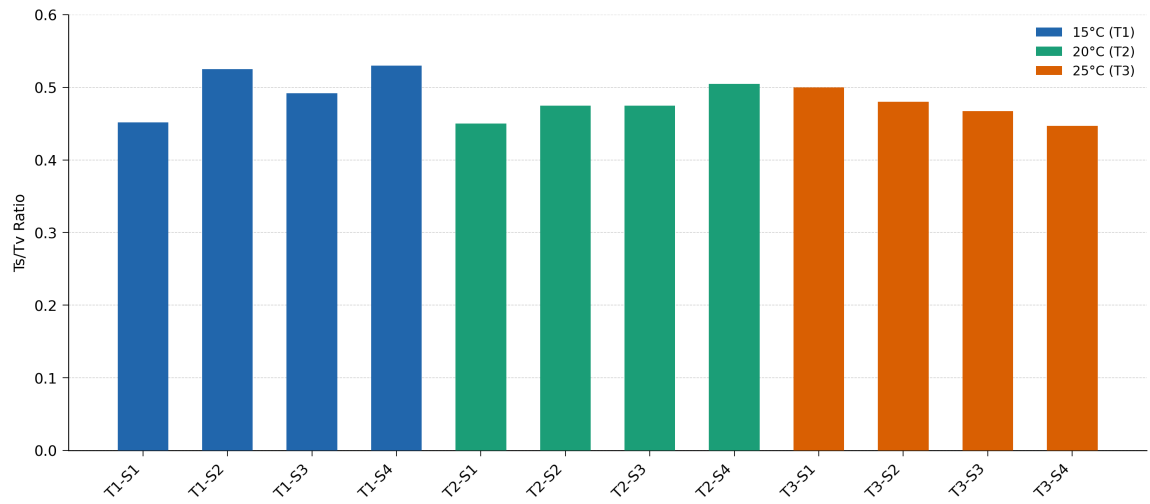


Figure 3.7: HSP-16.2 showed the highest ratios overall, ranging from 0.45 to 0.53, with the 20°C group consistently elevated around 0.50–0.51.

These genes were selected to investigate localized variation, conserved regions, and environmentally sensitive genomic behavior under different temperature conditions.

Chapter 4

Transformer-Based Prediction Framework

The rapid advancement of deep learning has introduced a new class of sequence modeling architectures that fundamentally differ from classical recurrent approaches. Among these, the Transformer architecture has emerged as the dominant paradigm for learning rich contextual representations from sequential data. Originally conceived for machine translation in natural language processing (NLP), transformers have since demonstrated extraordinary versatility across domains including vision, protein structure prediction, and most recently, genomic sequence analysis.

In the context of DNA sequence modeling, nucleotides play an analogous role to words in a sentence: their meaning, function, and effect are deeply context-dependent. A single nucleotide substitution at one position may alter gene expression, splice-site recognition, or transcription factor binding at positions hundreds of base pairs away. This long-range contextual dependency renders traditional convolutional or recurrent neural networks insufficient for capturing the full structural grammar of genomic sequences. The self-attention mechanism at the heart of the transformer architecture is uniquely suited to overcome this limitation, as it computes direct pairwise interactions between every pair of sequence positions regardless of their distance.

This chapter presents the transformer-based Masked Language Modeling (MLM) framework employed in this work. We describe the architectural components of the transformer encoder, the theoretical basis of the self-attention mechanism, the BERT-style bidirectional pre-training strategy, the MLM objective function, and the specific model configuration choices made for genomic sequence analysis. Together, these elements constitute a principled, self-supervised framework for learning contextual nucleotide representations without requiring any manually annotated genomic labels.

4.1 Transformer Architecture

The transformer architecture departs fundamentally from sequential processing by introducing a fully attention-based computation graph. Rather than encoding sequence information through hidden states propagated step-by-step as in recurrent neural networks (RNNs) or Long Short-Term Memory (LSTM) networks, the transformer operates on the entire input sequence simultaneously. This parallel processing capability

confers two major advantages: significantly faster training through GPU parallelism, and the ability to model dependencies between sequence positions without the gradient vanishing problem that afflicts deep recurrent models.

At the highest level, a transformer model receives a sequence of discrete tokens as input. In natural language processing these tokens are words or subwords; in the genomic setting employed here, tokens correspond to individual nucleotides or fixed-length k -mer substrings of DNA. Each token is first mapped to a dense embedding vector in a continuous representation space. Positional encodings are then added to these embeddings to supply the model with ordering information, since the attention operation is inherently permutation-invariant. The resulting token representations are then passed through a stack of encoder layers, each of which refines the contextual representation through multi-head self-attention and position-wise feed-forward transformations.

The overall transformer framework used in this work consists of four primary components:

1. **Token and Positional Embedding Layer:** Maps discrete nucleotide tokens to continuous dense vectors and injects sequence-order information through positional encodings.
2. **Multi-Head Self-Attention:** The central mechanism that computes context-aware representations by allowing each sequence position to attend to all other positions.
3. **Position-wise Feed-Forward Networks:** Non-linear transformation layers applied independently at each position to increase representational capacity.
4. **Residual Connections and Layer Normalization:** Stabilization mechanisms that enable effective training of deep encoder stacks by mitigating vanishing gradient effects and normalizing layer activations.

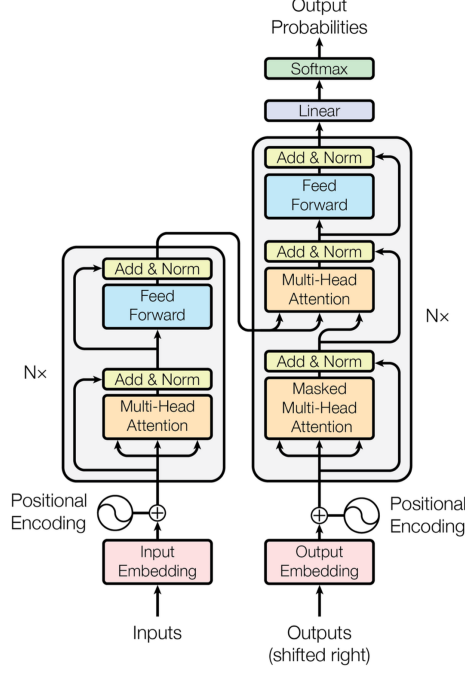


Figure 1: The Transformer - model architecture.

Figure 4.1: Schematic architecture of the Transformer encoder.

These components are arranged into a deep stack of encoder layers. Each layer receives the output of the preceding layer as its input, progressively building richer and more abstract contextual representations of the nucleotide sequence. The depth of the stack is a key architectural hyperparameter that controls the model’s capacity for hierarchical feature learning.

4.1.1 Encoder Architecture

The encoder is the primary computational block responsible for transforming a raw nucleotide sequence into a sequence of contextual embedding vectors. In contrast to decoders used in sequence-to-sequence tasks (such as machine translation), the encoder used in this work follows the BERT paradigm, which employs only the encoder stack to produce bidirectional sequence representations. No autoregressive decoder is required because the learning objective is nucleotide prediction through masked language modeling rather than sequence generation.

Each encoder layer ℓ applies a two-step transformation to its input $\mathbf{H}^{(\ell-1)} \in \mathbb{R}^{n \times d_{\text{model}}}$, where n is the sequence length and d_{model} is the hidden embedding dimension:

1. Multi-Head Self-Attention Sublayer:

$$\mathbf{H}'^{(\ell)} = \text{LayerNorm}\left(\mathbf{H}^{(\ell-1)} + \text{MultiHead}\left(\mathbf{H}^{(\ell-1)}\right)\right) \quad (4.1)$$

2. Feed-Forward Sublayer:

$$\mathbf{H}^{(\ell)} = \text{LayerNorm}(\mathbf{H}'^{(\ell)} + \text{FFN}(\mathbf{H}'^{(\ell)})) \quad (4.2)$$

The residual connection $(\mathbf{H}^{(\ell-1)} + \cdot)$ in Equations (4.1) and (4.2) enables gradient flow directly from the output of any encoder layer to the input layers during backpropagation, making training of deep encoder stacks stable. Layer normalization subsequently normalizes the summed representation across the hidden dimension, reducing internal covariate shift and accelerating convergence.

The position-wise feed-forward network (FFN) applies two linear transformations with a non-linear activation in between, independently at each sequence position:

$$\text{FFN}(\mathbf{x}) = \text{ReLU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1) \mathbf{W}_2 + \mathbf{b}_2 \quad (4.3)$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_{\text{model}} \times d_{\text{ff}}}$ and $\mathbf{W}_2 \in \mathbb{R}^{d_{\text{ff}} \times d_{\text{model}}}$ are learnable weight matrices and d_{ff} is typically set to $4 \times d_{\text{model}}$. The FFN layer increases the nonlinear transformation capacity of each encoder layer and allows the model to learn complex position-level feature interactions beyond what attention alone can express.

4.1.2 Multi-Head Self-Attention

Multi-head self-attention is the defining mechanism of the transformer architecture and the primary driver of its ability to model long-range contextual dependencies. The central insight is that a single attention function may be insufficient to capture all the different types of contextual relationships present in a complex sequence. Multi-head attention addresses this by running h independent attention operations—called attention heads—in parallel, each learning to focus on different aspects of the sequence context.

For each attention head $i \in \{1, \dots, h\}$, three separate linear projections are applied to the input representation $\mathbf{H}$:

$$\mathbf{Q}_i = \mathbf{H} \mathbf{W}_i^Q, \quad \mathbf{W}_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k} \quad (4.4)$$

$$\mathbf{K}_i = \mathbf{H} \mathbf{W}_i^K, \quad \mathbf{W}_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k} \quad (4.5)$$

$$\mathbf{V}_i = \mathbf{H} \mathbf{W}_i^V, \quad \mathbf{W}_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v} \quad (4.6)$$

where $d_k = d_v = d_{\text{model}}/h$ ensures that the total computational cost of multi-head attention is comparable to single-head attention with the full hidden dimension.

The outputs from all h heads are concatenated and linearly projected back to the original model dimension:

$$\text{MultiHead}(\mathbf{H}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \mathbf{W}^O \quad (4.7)$$

where $\text{head}_i = \text{Attention}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i)$ and $\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ is a learnable output projection matrix.

In genomic sequence modeling, different attention heads learn to specialize in qualitatively different biological interactions. Some heads may focus on local nucleotide co-occurrence patterns resembling k -mer statistics, others may capture conserved motifs such as transcription factor binding sites or splice signals, and still others may encode long-range chromosomal interaction patterns. This diversity of learned representations makes multi-head attention particularly powerful for biological sequence analysis.

4.1.3 Positional Encoding

A fundamental property of the self-attention operation is its permutation invariance: the attention output for any sequence position is identical regardless of the ordering of the remaining positions. While this property enables efficient parallel computation, it implies that without explicit positional information the model cannot distinguish between nucleotides at different sequence positions.

To address this, positional encodings are added to the token embeddings before the first encoder layer. In the original transformer formulation (Vaswani et al., 2017), fixed sinusoidal positional encodings are used:

$$\text{PE}(\text{pos}, 2i) = \sin\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right) \quad (4.8)$$

$$\text{PE}(\text{pos}, 2i + 1) = \cos\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right) \quad (4.9)$$

where pos is the absolute position index within the sequence and i is the embedding dimension index. The sinusoidal functions produce unique positional signatures for each position and allow the model to generalize to sequences longer than those seen during training. Alternatively, learnable positional embeddings—as used in BERT (Devlin et al., 2019)—associate a trainable vector with each absolute position up to a maximum sequence length.

For genomic sequences, the choice of positional encoding carries biological significance: the relative positions of nucleotides within a gene, regulatory element, or codon are physically meaningful, and the model must preserve this ordering information to accurately learn sequence grammar and functional dependencies.

4.2 Self-Attention Mechanism

The scaled dot-product attention is the fundamental operation underlying both the self-attention and multi-head attention computations described above. Given query, key, and value matrices, the attention function computes a weighted sum of value vectors, where the weights are determined by the similarity between query and key vectors.

4.2.1 Scaled Dot-Product Attention

Formally, given queries $\mathbf{Q} \in \mathbb{R}^{n \times d_k}$, keys $\mathbf{K} \in \mathbb{R}^{n \times d_k}$, and values $\mathbf{V} \in \mathbb{R}^{n \times d_v}$, the scaled dot-product attention is defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right) \mathbf{V} \quad (4.10)$$

The matrix product $\mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{n \times n}$ produces a raw similarity score matrix in which entry (i, j) quantifies how strongly position i should attend to position j . Scaling by $1/\sqrt{d_k}$ is a critical regularization step: without it, the dot products grow in magnitude proportionally to d_k , pushing the softmax function into regions of extremely small gradients that impair learning.

The softmax function converts each row of the scaled similarity matrix into a probability distribution over all sequence positions:

$$a_{ij} = \frac{\exp(e_{ij}/\sqrt{d_k})}{\sum_{k=1}^n \exp(e_{ik}/\sqrt{d_k})}, \quad e_{ij} = \mathbf{q}_i \cdot \mathbf{k}_j \quad (4.11)$$

The resulting attention weight a_{ij} represents the relative importance that position i assigns to position j when constructing its contextual representation. The final contextual embedding for position i is then:

$$\mathbf{z}_i = \sum_{j=1}^n a_{ij} \mathbf{v}_j \quad (4.12)$$

This mechanism allows the model to construct a contextualized representation for each nucleotide position as a weighted mixture of value vectors from all other positions, where the mixture weights are determined dynamically by sequence content rather than fixed by architecture.

4.2.2 Biological Interpretation of Attention Weights

The attention weight matrix $\mathbf{A} = [a_{ij}]$ provides an interpretable window into the model's internal reasoning. High attention weights a_{ij} indicate that nucleotide position j is contextually important for predicting or representing position i . In genomic

applications, this corresponds to biologically meaningful interactions: a transcription factor binding site may exhibit high mutual attention between conserved nucleotide positions within the motif, while splice donor and acceptor sites may share high attention weights reflecting their co-dependent functional role in defining an exon.

This interpretability distinguishes transformer models from many black-box deep learning approaches and enables post-hoc biological validation of the learned sequence representations.

4.3 BERT-Based Masked Language Modeling Framework

The specific transformer framework employed in this work is modeled on the Bidirectional Encoder Representations from Transformers (BERT) architecture, originally introduced by Devlin et al. (2019) for natural language understanding tasks. BERT’s design philosophy—pre-training a deep bidirectional transformer on large unlabeled corpora through self-supervised objectives—maps naturally to the genomic domain, where vast repositories of unannotated DNA sequences are available but labeled functional annotations are expensive and incomplete.

4.3.1 Bidirectional Contextual Learning

A fundamental distinction between BERT and earlier language models such as GPT (2018) is the direction of contextual learning. GPT processes sequences left-to-right in an autoregressive manner, meaning that the representation of position i depends only on positions $1, 2, \dots, i - 1$. BERT instead learns *bidirectional* representations: the contextual embedding of position i is informed by all other positions in the sequence, both to the left and to the right.

Formally, for a sequence $X = (x_1, x_2, \dots, x_n)$, the BERT encoder produces a contextual representation for position i conditioned on the full sequence context:

$$\mathbf{h}_i = f_\theta(x_i \mid x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \quad (4.13)$$

where f_θ denotes the transformer encoder parameterized by θ .

This bidirectionality is particularly important for genomic sequences. Biological function rarely arises from upstream context alone: the activity of a promoter depends on downstream gene structure, splice site recognition depends on both upstream exonic and downstream intronic context, and codon usage biases reflect constraints from both neighboring codons and the surrounding open reading frame. Bidirectional contextual learning therefore enables the model to capture the full information structure of genomic sequences in a biologically principled manner.

4.3.2 Masked Language Modeling (MLM)

Masked Language Modeling is the self-supervised pre-training objective that enables BERT to learn bidirectional contextual representations without requiring any labeled annotations. The MLM objective operationalizes the following intuition: a model that has truly understood the statistical and functional grammar of a sequence should be able to reconstruct any arbitrarily removed portion of that sequence from its surrounding context.

Masking Procedure

Given an input nucleotide sequence $X = (x_1, x_2, \dots, x_n)$, the MLM procedure randomly selects a subset of positions $\mathcal{M} \subset \{1, 2, \dots, n\}$ for masking. Consistent with the original BERT protocol (?), approximately 15% of all positions are selected for each training sequence. Of these selected positions:

- 80% are replaced with a special [MASK] token, prompting the model to actively predict the original nucleotide.
- 10% are replaced with a randomly sampled nucleotide, exposing the model to noisy input and improving robustness.
- 10% are left unchanged, encouraging the model to maintain accurate contextual representations even at unmasked positions.

This mixed masking strategy prevents the model from trivially learning to predict only at positions where the [MASK] token appears, thereby producing more generalizable contextual representations.

Prediction Objective

The masked input sequence $\tilde{X}$ is passed through the transformer encoder to produce contextual embeddings for all positions. At each masked position $i \in \mathcal{M}$, a linear classification head followed by a softmax activation maps the contextual embedding $\mathbf{h}_i$ to a probability distribution over the nucleotide vocabulary $\mathcal{V} = \{A, T, G, C\}$:

$$P(x_i = v \mid \tilde{X}) = \frac{\exp(\mathbf{w}_v^\top \mathbf{h}_i + b_v)}{\sum_{v' \in \mathcal{V}} \exp(\mathbf{w}_{v'}^\top \mathbf{h}_i + b_{v'})} \quad (4.14)$$

where $\mathbf{w}_v \in \mathbb{R}^{d_{\text{model}}}$ and $b_v \in \mathbb{R}$ are learnable parameters of the output classification head corresponding to nucleotide v .

MLM Loss Function

The model is trained by minimizing the MLM cross-entropy loss, which measures the average negative log-likelihood of predicting the correct nucleotide at each masked position:

$$\mathcal{L}_{\text{MLM}} = -\frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \log P(x_i | \tilde{X}) \quad (4.15)$$

Minimizing $\mathcal{L}_{\text{MLM}}$ forces the encoder to learn rich contextual representations that encode the statistical regularities, conserved motifs, and long-range dependencies present in genomic sequences. Since no external labels are required, the MLM objective can be applied to arbitrarily large collections of unannotated genomic sequence data, enabling large-scale pre-training.

Why MLM is Suitable for Genomic Sequences

The MLM framework aligns particularly well with the structure of genomic data for several reasons. First, genomic sequences exhibit strong positional dependencies: nucleotides are not independently distributed but are constrained by codon structure, secondary structure formation, regulatory motif conservation, and evolutionary selective pressure. The MLM objective forces the model to internalize these constraints in order to minimize prediction loss.

Second, the self-supervised nature of MLM allows the model to exploit the massive volume of available genome sequence data without requiring expensive experimental annotation. Third, the bidirectional context used during prediction ensures that the learned representations encode full-sequence information rather than truncated unidirectional context, making the pre-trained model directly applicable to downstream tasks such as variant effect prediction, enhancer classification, and splice site detection.

4.4 Model Configuration

The transformer-based genomic MLM framework was implemented using a configurable BERT-style encoder architecture specifically adapted for nucleotide sequence analysis. The model configuration was determined by balancing contextual learning capacity, downstream task performance, and computational feasibility given the scale of the genomic datasets used during pre-training.

The key architectural and training hyperparameters are described in detail below, along with the rationale for the specific values chosen.

4.4.1 Hidden Embedding Dimension

The hidden embedding dimension d_{model} defines the size of the continuous vector space in which each nucleotide token is represented after embedding and throughout the encoder stack. Every intermediate computation within the transformer—including the attention projections, residual additions, and feed-forward transformations—operates in this d_{model} -dimensional space.

A larger hidden dimension provides greater representational capacity: the model can encode more complex contextual information into the embedding of each nucleotide. However, increasing d_{model} also increases the number of learnable parameters quadratically (in the attention weight matrices) and raises memory requirements proportionally to the sequence length times d_{model} . The hidden dimension was therefore selected to provide sufficient capacity for learning meaningful genomic representations while remaining computationally tractable for the hardware available.

4.4.2 Number of Attention Heads

The number of attention heads h determines how many independent attention operations are computed in parallel within each multi-head attention sublayer. Importantly, each head operates on a projected subspace of dimension $d_k = d_{\text{model}}/h$, so increasing h divides the representational capacity across more specialized but lower-dimensional subspaces.

In practice, a larger number of attention heads allows the model to simultaneously capture a broader variety of contextual relationships. In the genomic setting, this translates to the ability to encode multiple qualitatively different sequence features in parallel: local di- and tri-nucleotide statistics, medium-range motif co-occurrences, and long-range chromosomal interaction signals may each be represented by distinct attention heads. Empirically, transformer models benefit from a sufficiently large number of heads relative to the hidden dimension, with $h = d_{\text{model}}/64$ often used as a practical guideline.

4.4.3 Number of Encoder Layers

The depth of the transformer encoder—measured by the number of stacked encoder layers N —is one of the most important determinants of model capacity. Each additional encoder layer refines the contextual representation produced by the preceding layer, enabling the model to learn increasingly abstract and hierarchical features of the input sequence.

In shallow encoder stacks (small N), the model primarily captures local and low-level sequence features such as nucleotide co-occurrences and short-range dependency patterns. As N increases, higher encoder layers develop representations that encode

increasingly long-range and abstract sequence properties, analogous to the hierarchy of feature abstraction observed in deep convolutional networks for image recognition.

However, very deep encoder stacks require substantially more parameters, longer training times, and larger training datasets to avoid overfitting. The number of encoder layers was selected as a compromise between representational depth and practical trainability on the available genomic pre-training corpus.

4.4.4 Batch Size

The batch size defines the number of sequence segments processed simultaneously in each gradient update step during training. Batch size selection involves a fundamental trade-off: larger batches produce lower-variance gradient estimates (since each gradient is averaged over more samples), improving training stability and enabling the use of higher learning rates. However, large batches require proportionally more GPU memory and may lead to convergence to sharper local minima with poorer generalization (?).

For transformer-based genomic models, memory constraints are particularly significant because the $n \times n$ attention weight matrix grows quadratically with sequence length n . The batch size was therefore selected in conjunction with the maximum sequence length to ensure that all required intermediate activations fit within available GPU memory while maintaining effective gradient estimation.

4.4.5 Learning Rate and Optimization Schedule

The learning rate is one of the most critical hyperparameters governing transformer pre-training. Following the training procedure established in the original BERT paper (?), a linear warm-up schedule followed by linear decay was employed. During the warm-up phase, the learning rate increases linearly from zero to the peak learning rate over a predefined number of warm-up steps. This warm-up period prevents large, destabilizing gradient updates in the early stages of training when the model parameters are far from their optimal values.

Formally, the learning rate schedule is given by:

$$\eta(t) = \begin{cases} \eta_{\max} \cdot \frac{t}{t_{\text{warmup}}} & \text{if } t < t_{\text{warmup}} \\ \eta_{\max} \cdot \frac{T - t}{T - t_{\text{warmup}}} & \text{if } t \geq t_{\text{warmup}} \end{cases} \quad (4.16)$$

where t is the current training step, T is the total number of training steps, t_{warmup} is the number of warm-up steps, and $\eta_{\max}$ is the peak learning rate.

The Adam optimizer (?) was used throughout training, combining adaptive per-parameter learning rates with momentum estimation. Adam is particularly well-suited for transformer training because the adaptive learning rate correction compensates for

the different gradient scales that arise across layers and attention heads of varying depth and function. Weight decay regularization was additionally applied to all non-bias parameters to reduce overfitting, following the AdamW variant (Loshchilov & Hutter, 2017).

The combination of warm-up scheduling, adaptive optimization, and weight decay regularization constitutes the standard recipe for stable and effective transformer pre-training and was found to produce reliable convergence behavior across the genomic datasets used in this work.

Chapter 5

Methodology

This chapter describes the analytical framework developed for studying contextual nucleotide predictability and conserved sequence behavior across temperature-dependent genomic datasets. The methodology is organized into three major components:

- Consistent window analysis
- Metadata construction
- Surprisal-based prediction analysis

The framework was applied on three biologically significant genes associated with stress response and genomic regulation in *Caenorhabditis elegans*:

- HSF-1
- ISY-1
- HSP-16.2

These genes were selected because of their functional importance in heat shock response, stress adaptation, and regulatory behavior under varying environmental conditions.

5.1 Consistent Window Analysis

The first stage of the methodology involved identifying consistent and robust prediction regions across corresponding genomic samples under different temperature conditions. The model was trained on samples from the T1 dataset and tested on the corresponding samples from the T2 and T3 datasets, where for each sample index i , the model trained on T1- S_i was evaluated on T2- S_i and T3- S_i . Instead of analyzing isolated nucleotide predictions alone, a sliding window framework was developed to locate contiguous sequence regions exhibiting stable contextual predictability.

For each corresponding sample pair between the T2 and T3 datasets, masked nucleotide prediction results were compared position-wise and the number of correctly predicted indices within fixed-size windows was calculated. The analysis was performed independently for the three selected genes:

- HSP-16.2
- ISY-1
- HSF-1

For every sample pair:

$$T2-S_i \quad \text{vs} \quad T3-S_i$$

the following quantities were computed:

- Total number of correct prediction indices
- Most robust prediction window
- Number of correct indices inside the robust window
- Distribution of correct predictions across sliding windows

The robust windows correspond to genomic regions where contextual prediction remained relatively stable across temperature conditions, indicating stronger sequence conservation and contextual constraint.

5.1.1 HSP-16.2 Consistent Window Analysis

The HSP-16.2 gene exhibited moderate contextual consistency across the temperature datasets. Multiple localized regions showed relatively stable prediction behavior, particularly in the central sequence regions.

The analysis revealed that the most robust windows were concentrated approximately between indices 170–520 depending on the sample pair. These regions contained the largest concentration of correctly predicted indices and therefore represented contextually conserved regions within the gene.

Compared to HSF-1 and ISY-1, HSP-16.2 displayed lower total correct prediction counts, suggesting comparatively weaker contextual conservation under temperature variation.

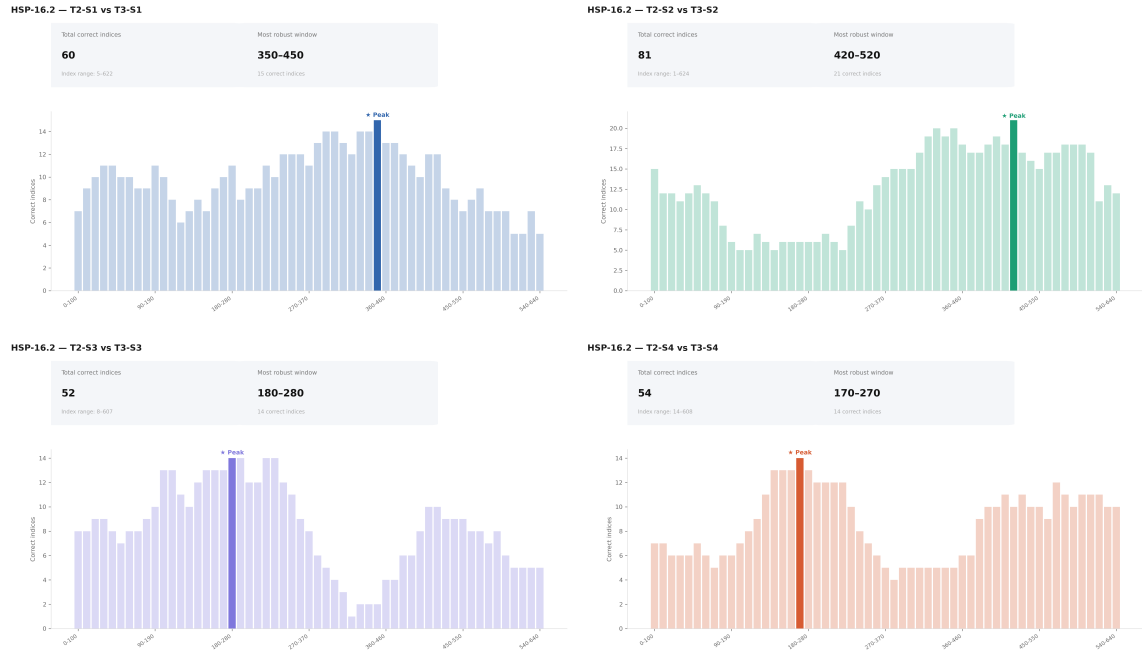


Figure 5.1: Consistent window analysis for the HSP-16.2 gene across corresponding temperature-dependent sample pairs. Each subplot illustrates the distribution of correct prediction indices across sliding windows together with the most robust prediction region.

5.1.2 ISY-1 Consistent Window Analysis

The ISY-1 gene demonstrated relatively stable contextual behavior across all temperature conditions. The sliding window analysis showed broader regions of consistent prediction compared to HSP-16.2.

Most robust windows were concentrated approximately between indices 320–1040, depending on the sample pair. These regions consistently exhibited higher numbers of correct predictions, indicating strong contextual conservation and stable sequence organization.

Since ISY-1 functions as a regulatory gene involved in pre-mRNA splicing and gene-expression control, the observed stability aligns with its biologically conserved functional role.

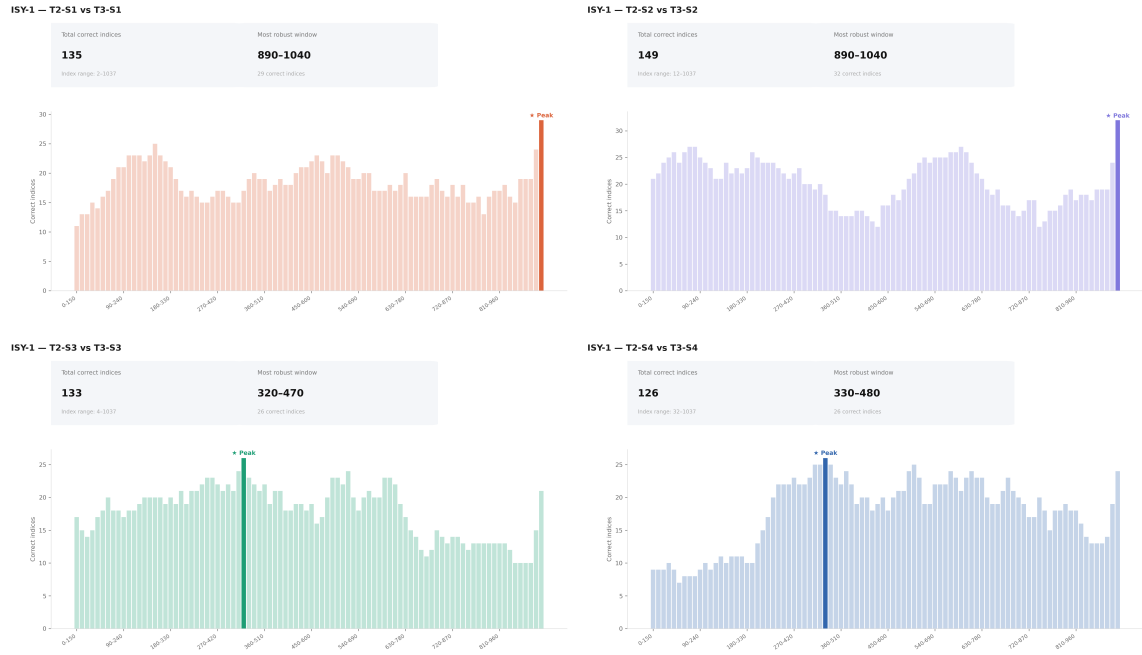


Figure 5.2: Consistent window analysis for the ISY-1 gene across corresponding temperature-dependent sample pairs. The plots highlight stable prediction regions and robust contextual windows preserved across environmental variation.

5.1.3 HSF-1 Consistent Window Analysis

Among the analyzed genes, HSF-1 exhibited the strongest contextual consistency and largest number of correctly predicted indices across temperature conditions.

The most robust windows were primarily concentrated between indices 1200–2400, depending on the sample pair. These regions consistently showed high densities of correct predictions, indicating strong contextual organization and conserved sequence structure.

Since HSF-1 acts as the master regulator of the heat shock response pathway, the presence of highly conserved contextual regions suggests strong evolutionary and functional constraints associated with stress-response regulation.

The HSF-1 analysis also exhibited comparatively broader stable regions than both HSP-16.2 and ISY-1, indicating more extensive long-range contextual dependencies.

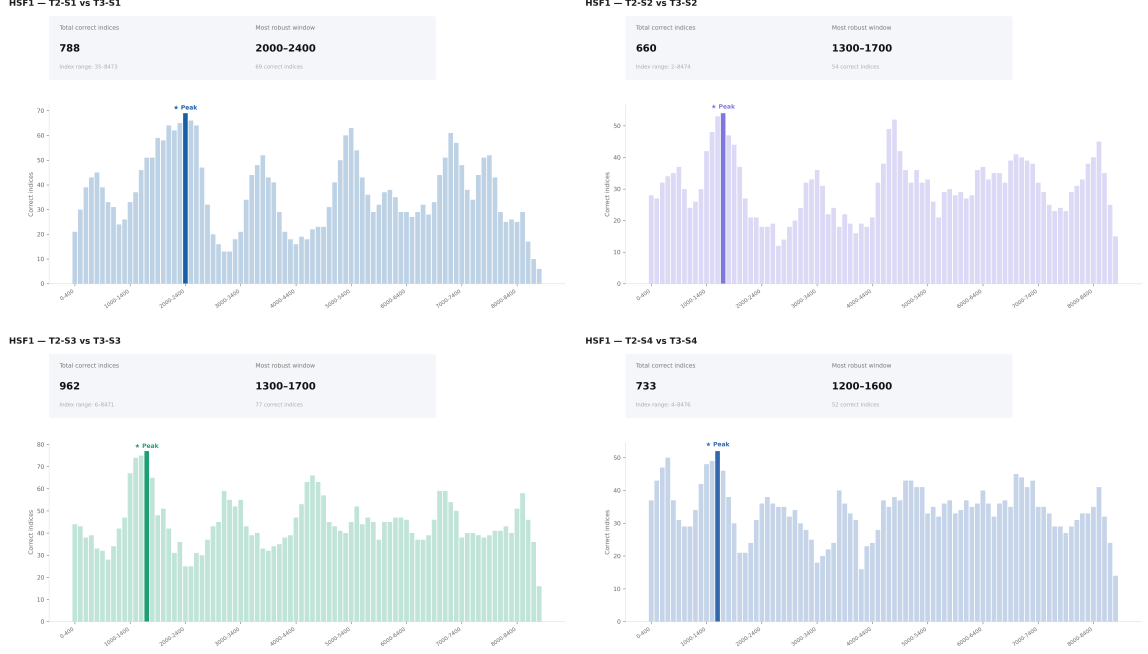


Figure 5.3: Consistent window analysis for the HSF-1 gene across corresponding temperature-dependent sample pairs. The robust windows exhibit high densities of correct predictions and strong contextual conservation across temperature variation.

5.2 Metadata Construction

5.2.1 Competitive Positions and Surprisal Distribution

To further characterise model uncertainty, all evaluated nucleotide positions were screened for *competitive positions*. A position was classified as competitive when the top two predicted nucleotide probabilities were sufficiently close to each other, indicating that the transformer model could not confidently distinguish between multiple nucleotide candidates.

The surprisal distribution of competitive positions for HSF1 File 11 is shown in Figure 5.4. The histogram reveals that the majority of competitive positions cluster near a surprisal value of 1.0 bit, corresponding to near-equal binary uncertainty between two nucleotide bases.

The distribution was divided into three interpretive categories:

- **Lower competitive positions** ($S < 0.85$ bits): These positions represent cases where the model still exhibits partial confidence toward one nucleotide despite remaining within the competitive threshold. Approximately 12.8% of all competitive positions belong to this category.
- **Near 50/50 competitive positions** ($0.85 \leq S < 1.00$ bits): This region dominates the distribution, accounting for approximately 79.1% of competitive

positions. These positions correspond to nearly equal probability assignment between the two most likely nucleotides, indicating maximum contextual ambiguity within the binary prediction space.

- **Above 50/50 positions** ($S \geq 1.00$ bits): Approximately 8.0% of competitive positions exhibit surprisal values above 1.0 bit. These positions represent highly uncertain regions where the prediction probability is distributed across three or more nucleotide candidates rather than a simple two-base competition.

The calculated mean surprisal value for competitive positions was:

$$\mu_S = 0.9501 \text{ bits}$$

while the median surprisal was:

$$\tilde{S} = 0.9864 \text{ bits}$$

The proximity of both values to 1.0 bit confirms that most competitive positions occur near the theoretical maximum of binary uncertainty. The relatively narrow spread of the distribution further suggests that competitive sites are not randomly distributed but instead form a distinct uncertainty class within the genomic prediction landscape.

The histogram additionally demonstrates a pronounced concentration of positions immediately below the 50/50 uncertainty boundary, indicating that the model frequently encounters genomic contexts where two nucleotide bases receive nearly identical contextual support.

Positions with surprisal values exceeding 1.0 bit are of particular biological interest because they indicate regions where contextual information alone is insufficient for confident nucleotide discrimination. Such positions may correspond to repetitive regions, low-complexity sequence motifs, alignment ambiguity, or highly variable genomic regions.

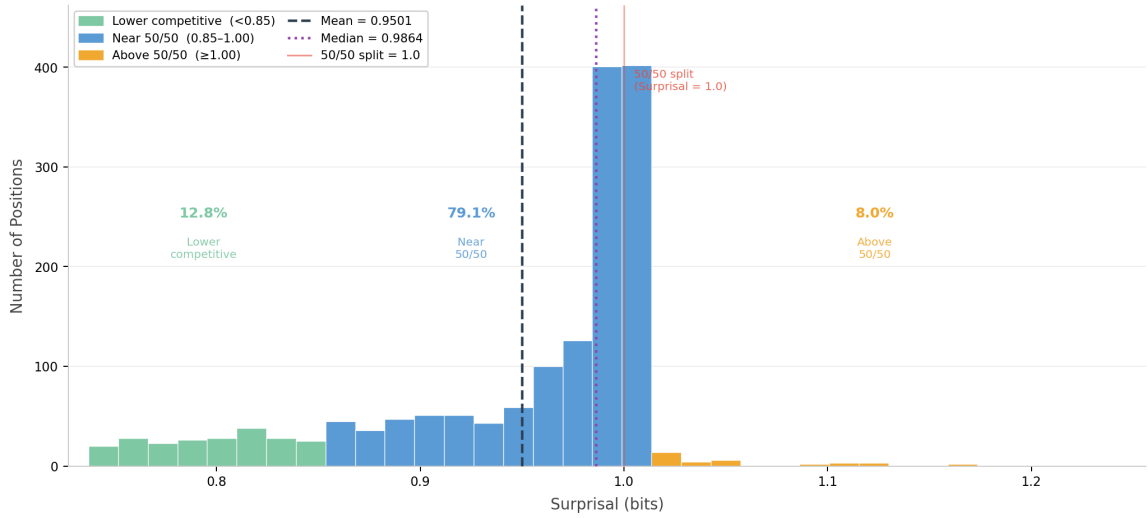


Figure 5.4: Distribution of surprisal values for competitive nucleotide positions in HSF1 sample. The histogram illustrates three uncertainty regions: lower competitive positions, near 50/50 uncertainty positions, and above-50/50 highly ambiguous positions. Dashed and dotted vertical lines indicate the mean and median surprisal respectively, while the red reference line marks the theoretical 50/50 uncertainty boundary at 1.0 bit.

5.3 Surprisal-Based Prediction Analysis

The final stage of the methodology involved surprisal-based analysis of nucleotide predictability across genomic sequences using a transformer-based masked language model (MLM). The objective of this analysis was to quantify how effectively the contextual information present within a genomic sequence can predict unseen nucleotide positions.

5.3.1 Training and Testing Strategy

For each genomic sequence, the transformer model was trained using only the middle 80% region of the sequence, while the first and last 10% regions were reserved exclusively for testing.

For a sequence of length L :

- First 10% → Left testing region
- Middle 80% → Training region
- Last 10% → Right testing region

The model therefore learned contextual nucleotide relationships only from the central region and was subsequently evaluated on completely unseen boundary regions of the same sequence.

For example, in the sequence pair:

$T1-S1$

the transformer model was trained only on the middle 80% of the $T1-S1$ sequence. After training, surprisal scores were calculated independently on:

- First 10% of $T1-S1$
- Last 10% of $T1-S1$

This process was repeated independently for every sequence sample across all temperature conditions and genes.

Such a training strategy ensures that:

- The model does not memorize the testing regions
- Prediction performance reflects genuine contextual learning
- Surprisal values measure true sequence predictability
- Conserved contextual dependencies can be identified

5.3.2 Definition of Surprisal

For a nucleotide position i , surprisal was defined as:

$$S(x_i) = -\log_2 P(x_i \mid \text{context})$$

where

$$P(x_i \mid \text{context})$$

represents the probability assigned by the transformer model to the true nucleotide given its surrounding sequence context.

The surprisal framework provides an information-theoretic measure of contextual predictability:

- Low surprisal $\rightarrow$ highly predictable nucleotide
- High surprisal $\rightarrow$ contextually uncertain nucleotide

Lower surprisal values indicate that the surrounding sequence context strongly constrains nucleotide identity, whereas larger surprisal values indicate weaker contextual certainty and higher prediction difficulty.

5.3.3 ISY-1 Gene Surprisal Analysis

First 10% Region Surprisal Profiles

The surprisal profiles corresponding to the first 10% region of the ISY-1 sequences exhibit noticeable positional fluctuations, reflecting variations in nucleotide predictability across the initial sequence segments. Most samples display moderate surprisal values with intermittent sharp peaks, indicating the coexistence of highly predictable and contextually uncertain nucleotide positions.

In several datasets, localized peaks with surprisal values exceeding 2.5 suggest regions where the transformer model encounters reduced contextual confidence. These peaks likely correspond to structurally variable or less repetitive subsequences. Conversely, low-surprisal regions (< 1.0) indicate conserved contextual patterns that are consistently recognized by the model across multiple samples.

Comparative analysis further shows that lower-temperature samples generally exhibit smoother surprisal transitions, whereas higher-temperature datasets display stronger oscillatory behavior and increased variability. This observation suggests that thermal variation may influence sequence organization and contextual dependencies within the genomic data.

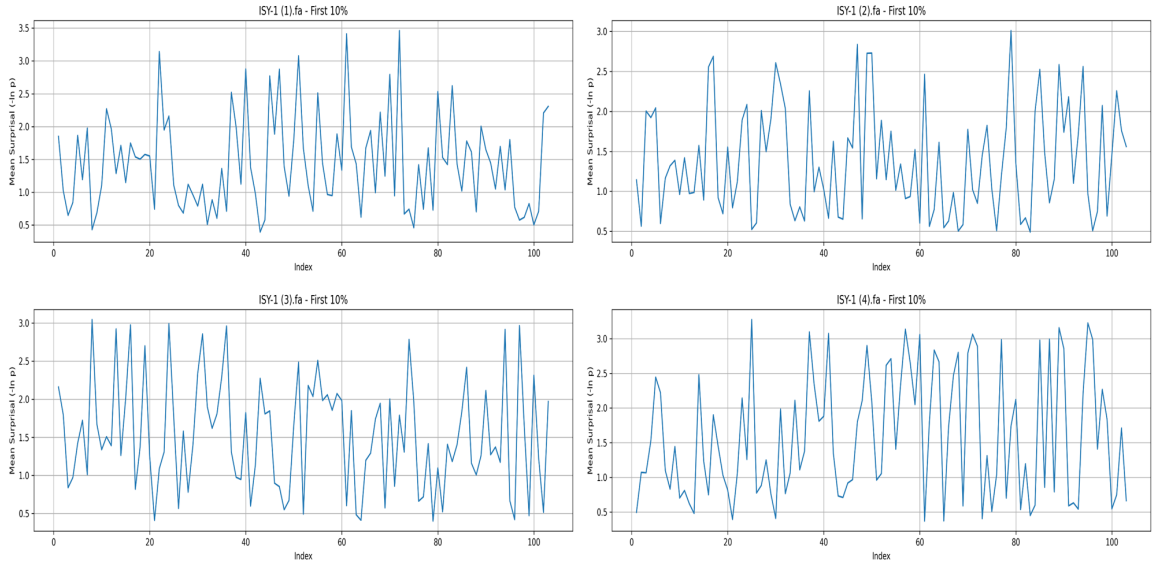


Figure 5.5: Mean surprisal profiles for the first 10% region of the ISY-1 gene at 15°C (samples 1–4). The plots show fluctuations in nucleotide-level surprisal across the initial segment of the sequence, highlighting regions of relatively high contextual uncertainty interspersed with low-surprisal conserved positions.

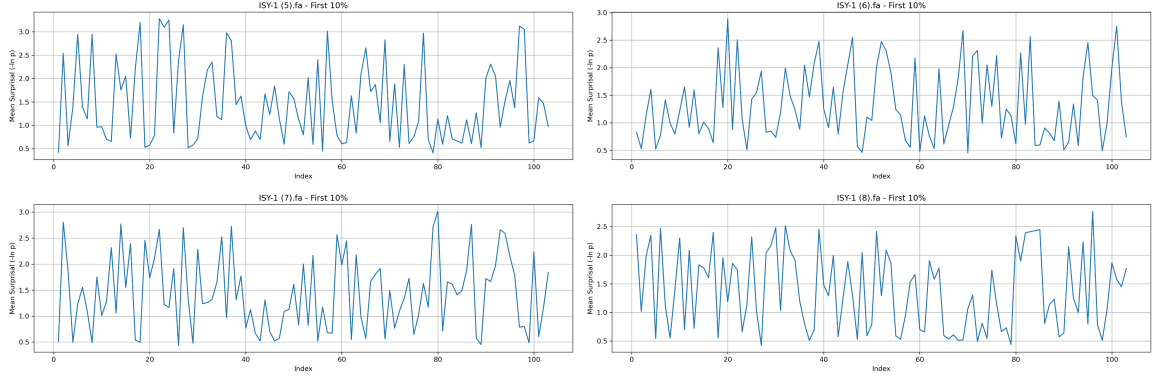


Figure 5.6: Mean surprisal profiles for the first 10% region of the ISY-1 gene at 20°C (samples 5–8). The intermediate-temperature samples display moderate surprisal fluctuations, with alternating predictable and high-uncertainty regions, suggesting partially preserved sequence regularity under changing thermal conditions.

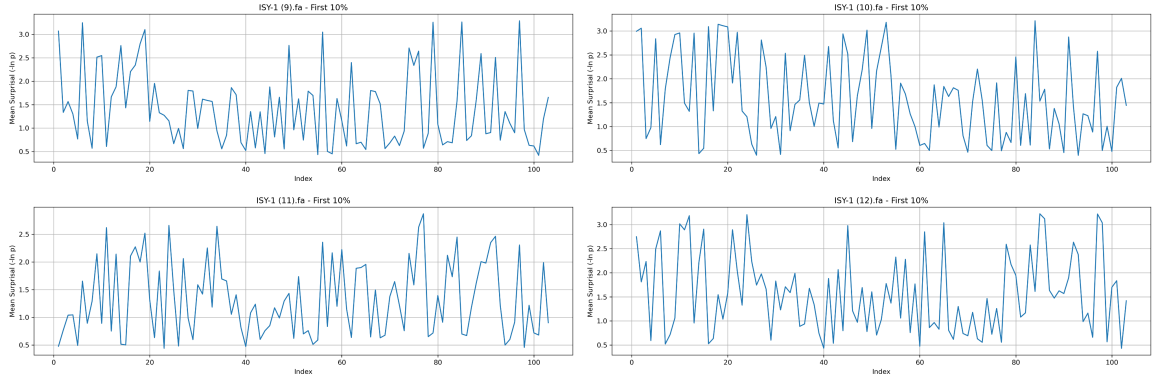


Figure 5.7: Mean surprisal profiles for the first 10% region of the ISY-1 gene at 25°C (samples 9–12). Compared with lower-temperature samples, the surprisal distributions exhibit stronger local variability and multiple sharp peaks, indicating increased contextual irregularity in specific sequence regions.

Overall, the first 10% surprisal profiles reveal that the transformer model successfully captures both conserved and irregular contextual structures present in the initial regions of the sequences.

Last 10% Region Surprisal Profiles

The surprisal profiles corresponding to the last 10% region of the ISY-1 sequences exhibit substantial fluctuations across all samples, indicating heterogeneous contextual predictability in the terminal regions of the genomic sequences. Despite variations in amplitude and local peak structure, several common characteristics can be observed across the twelve datasets.

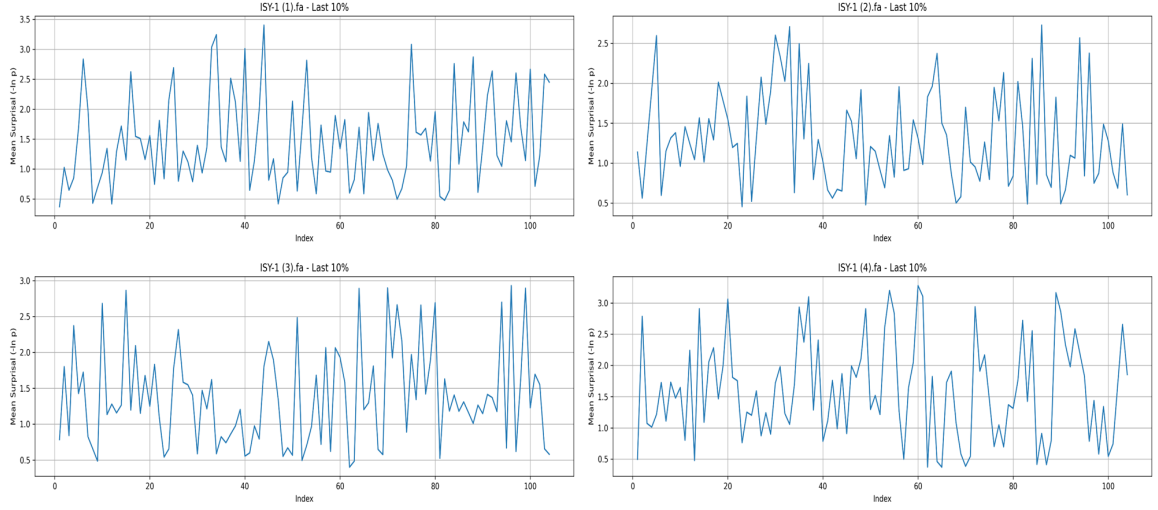


Figure 5.8: Mean surprisal profiles for the last 10% region of the ISY-1 gene sequences corresponding to samples 1–4. The plots illustrate fluctuations in nucleotide predictability across the terminal segment of the sequences, highlighting variations in contextual uncertainty captured by the transformer-based language model.

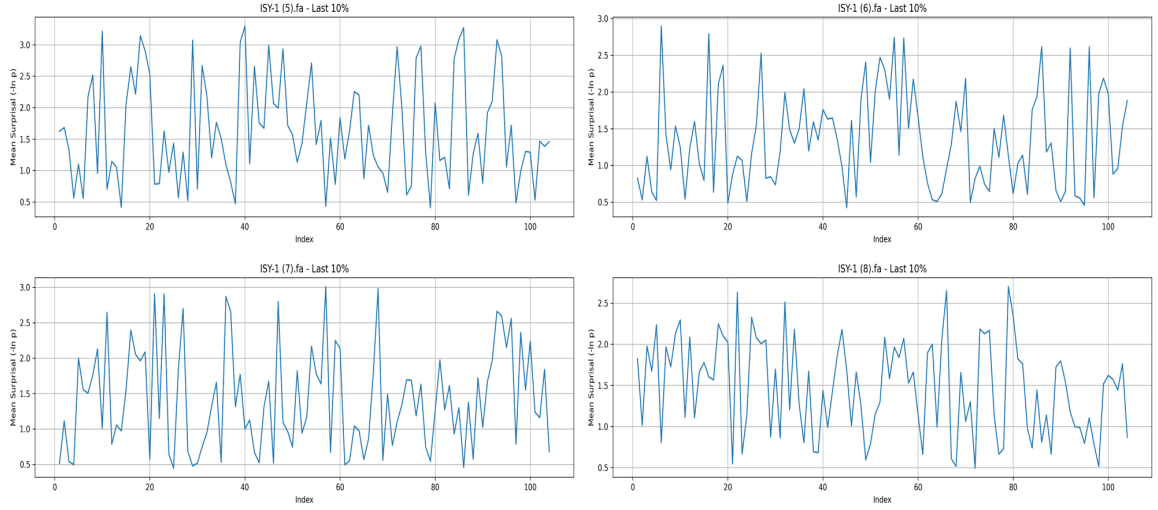


Figure 5.9: Mean surprisal profiles for the last 10% region of the ISY-1 gene sequences corresponding to samples 5–8. Distinct peaks and troughs in surprisal indicate localized differences in sequence predictability and contextual structure across the higher-temperature datasets.

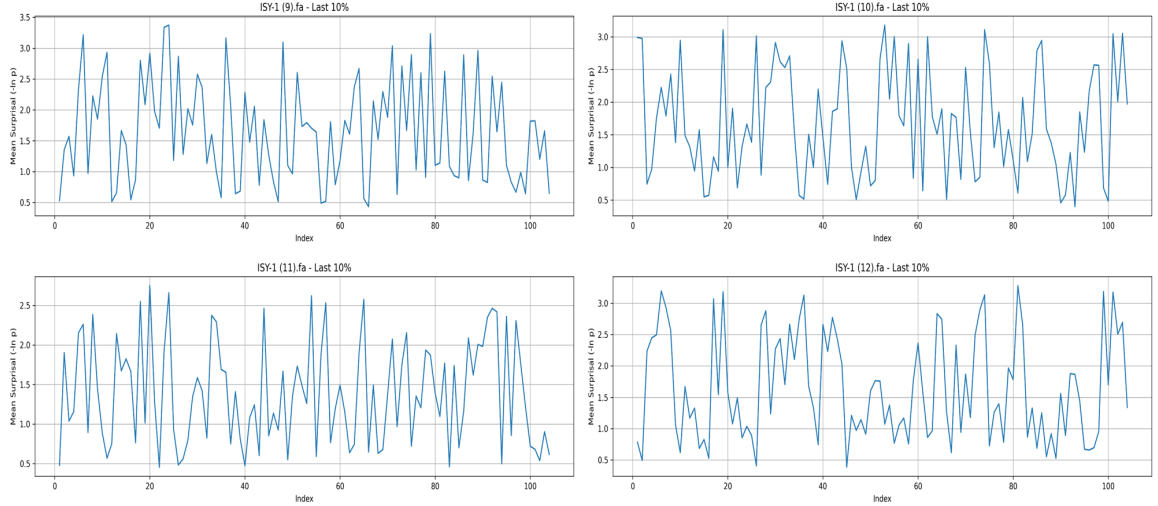


Figure 5.10: Mean surprisal profiles for the last 10% region of the ISY-1 gene sequences corresponding to samples 9–12. The observed variability in surprisal values suggests temperature-dependent changes in sequence regularity and model confidence in nucleotide prediction.

Samples trained and tested under similar conditions generally maintain surprisal values within the moderate range ($\sim 0.5 - 2.5$), suggesting that the transformer model retains contextual understanding in these sequence regions. However, intermittent sharp peaks reaching values above 3.0 are consistently observed in multiple samples, particularly in ISY-1 (1), ISY-1 (4), ISY-1 (5), ISY-1 (9), ISY-1 (10), and ISY-1 (12). These high-surprisal spikes correspond to locally unpredictable nucleotide positions where the learned contextual patterns become less certain.

The lower-temperature reference samples exhibit relatively smoother transitions between neighboring positions, whereas higher-temperature test samples display more irregular oscillatory behavior and frequent abrupt peaks. This suggests that increasing temperature introduces contextual perturbations that reduce sequence predictability and alter local nucleotide dependencies captured by the transformer model.

Across most samples, regions with low surprisal (< 1.0) remain interspersed between high-surprisal spikes, indicating that structurally conserved or repetitive subsequences are still recognized reliably by the model even in the terminal sequence regions. The coexistence of predictable and highly uncertain positions highlights the non-uniform structural organization of the sequences.

A comparative inspection further reveals that several samples share qualitatively similar peak locations and fluctuation patterns, implying the persistence of common contextual motifs across temperature conditions. Nevertheless, the magnitude and density of surprisal spikes increase in certain higher-temperature datasets, supporting the hypothesis that thermal variation influences the contextual complexity of the nucleotide arrangements.

Overall, the last 10% surprisal analysis demonstrates that transformer-based language models are capable of detecting localized contextual irregularities in genomic sequences, while simultaneously revealing temperature-dependent changes in sequence predictability and structural consistency.

5.3.4 *HSF-1* Gene Suprisal Analysis

First 10% Region Surprisal Profiles

The surprisal profiles corresponding to the first 10% region of the *hsf-1* sequences exhibit pronounced positional fluctuations with frequent sharp spikes distributed throughout the analyzed region. Most samples maintain baseline surprisal values around 0.8 to 1.2, indicating that a large portion of the sequence remains contextually predictable for the transformer model. However, intermittent high-surprisal peaks are consistently observed across all datasets, reflecting localized regions of increased contextual uncertainty.

Several samples display surprisal peaks approaching or exceeding 2.5, suggesting the presence of structurally variable or less conserved nucleotide subsequences within the initial region of the gene. These abrupt transitions indicate positions where the model assigns lower confidence to nucleotide prediction due to weaker contextual regularity. In contrast, extended low-surprisal regions (< 1.0) correspond to comparatively conserved sequence patterns that are repeatedly captured by the learned contextual representations.

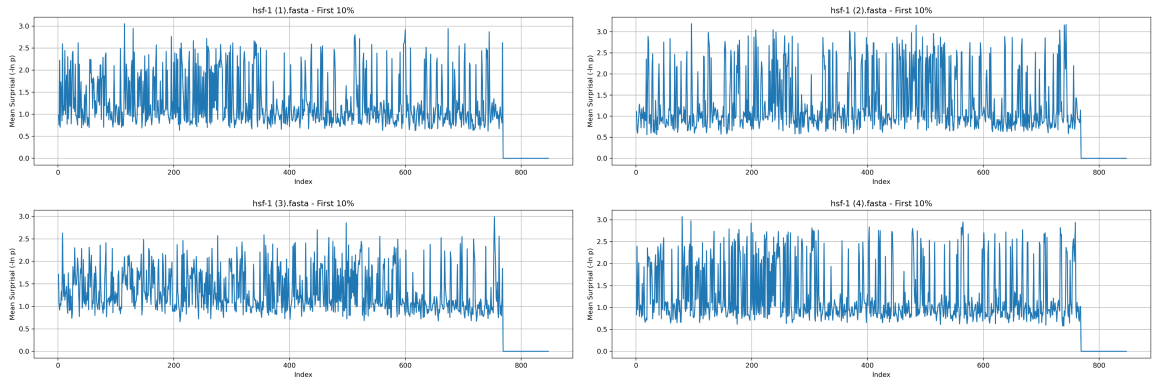


Figure 5.11: First 10% surprisal profiles for HSF-1 samples 1–4. The plots exhibit moderate baseline surprisal values with intermittent high-amplitude peaks, indicating localized regions of reduced contextual predictability within the initial sequence segments.

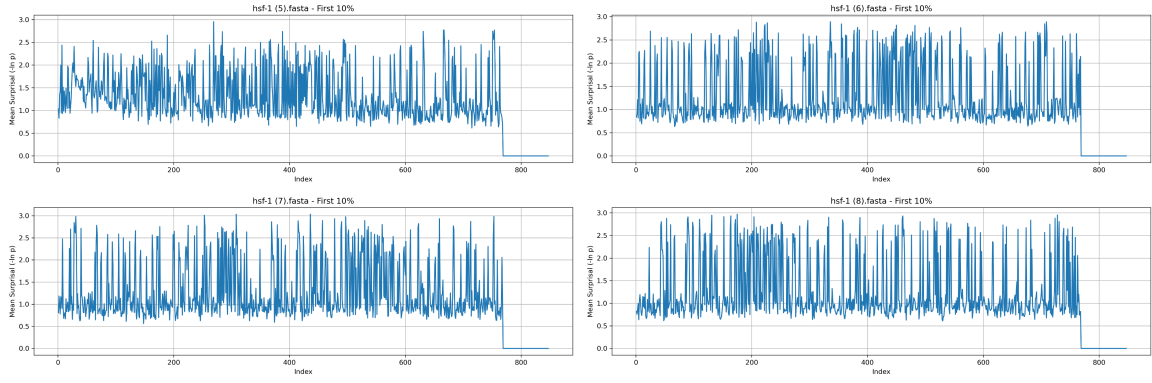


Figure 5.12: First 10% surprisal profiles for HSF-1 samples 5–8. Increased fluctuations and recurrent sharp surprisal peaks are observed across multiple regions, reflecting stronger contextual variability and oscillatory sequence behavior.

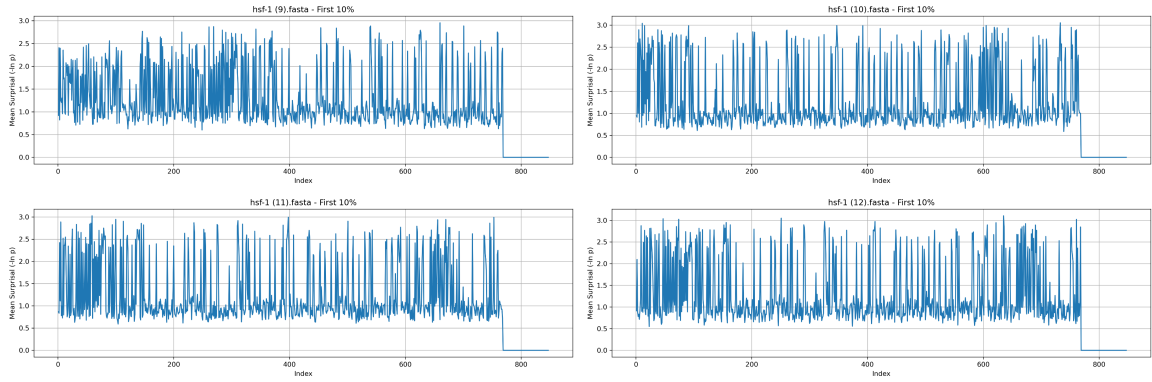


Figure 5.13: First 10% surprisal profiles for HSF-1 samples 9–12. The profiles display persistent positional variability with alternating low- and high-surprisal regions, suggesting heterogeneous contextual organization and thermally influenced sequence dynamics.

Comparative examination of the datasets further reveals that lower-temperature samples generally show relatively smoother surprisal distributions with moderate variability, whereas higher-temperature samples exhibit denser and more frequent high-surprisal oscillations. This increased fluctuation pattern suggests enhanced contextual complexity and reduced sequence regularity under elevated temperature conditions. Overall, the surprisal dynamics of the first 10% region of *hsf-1* indicate the coexistence of stable conserved motifs and locally irregular sequence segments, highlighting potential temperature-dependent variations in genomic organization and contextual structure.

Last 10% Region Surprisal Profiles

The surprisal profiles corresponding to the last 10% region of the *hsf-1* sequences demonstrate substantial positional variability characterized by recurrent sharp fluctuations and intermittent high-surprisal peaks across nearly all samples. Most regions

maintain baseline surprisal values within the range of approximately 0.8 to 1.3, indicating that a significant portion of the terminal sequence remains contextually predictable for the transformer-based language model. Nevertheless, numerous abrupt spikes exceeding 2.0 are distributed throughout the profiles, reflecting localized regions of increased contextual uncertainty and sequence irregularity.

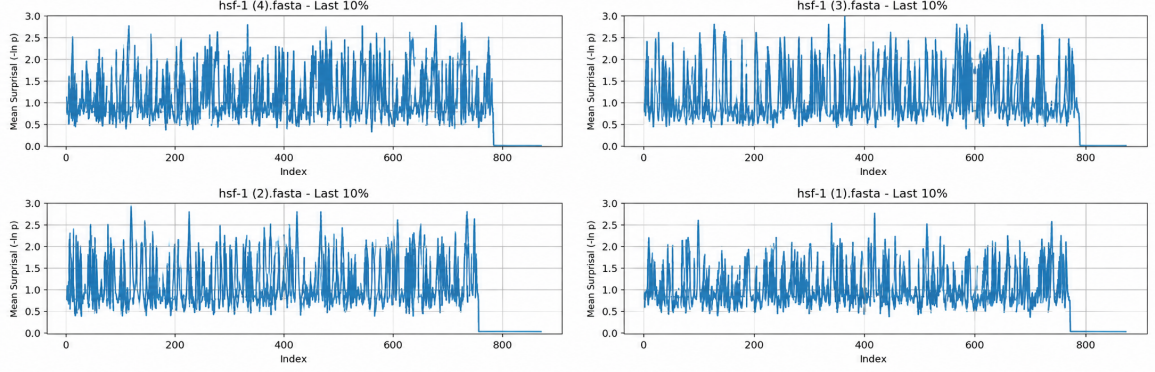


Figure 5.14: Last 10% surprisal profiles for HSF-1 samples 1–4. The profiles exhibit persistent positional fluctuations with moderate baseline surprisal values and recurring sharp peaks, indicating localized regions of contextual uncertainty within the terminal sequence segments.

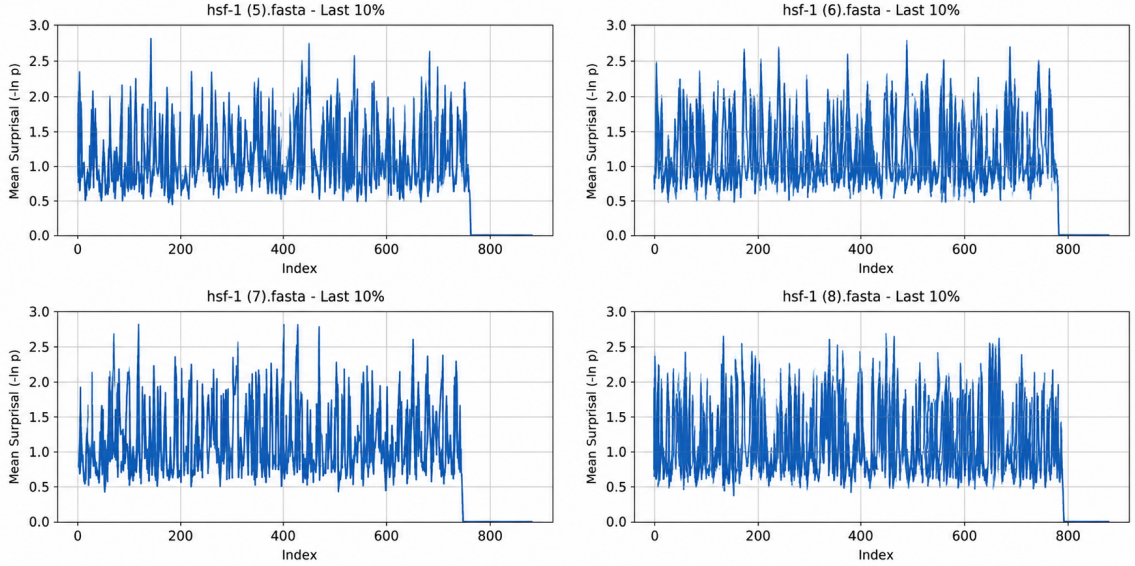


Figure 5.15: Last 10% surprisal profiles for HSF-1 samples 5–8. Increased oscillatory behavior and frequent high-amplitude surprisal peaks are observed across multiple positions, reflecting enhanced contextual variability and heterogeneous sequence organization near the sequence termini.

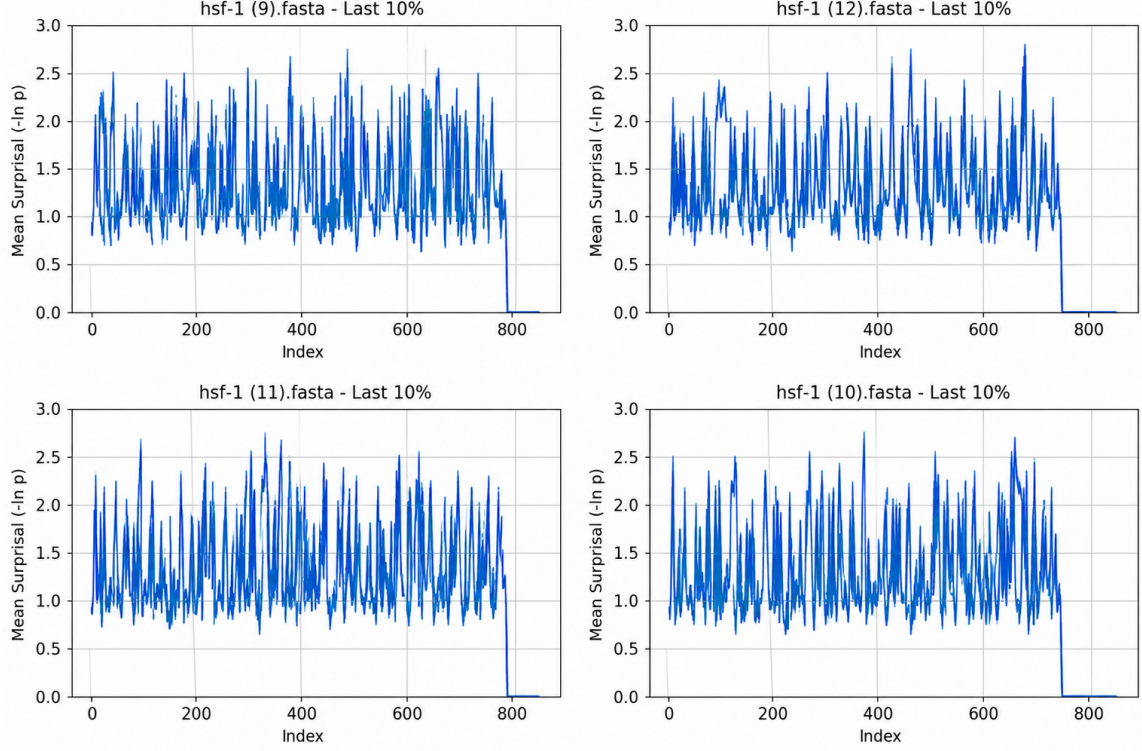


Figure 5.16: Last 10% surprisal profiles for HSF-1 samples 9–12. The profiles display alternating low- and high-surprisal regions with sustained positional variability, suggesting thermally influenced contextual dynamics and structurally diverse sequence patterns in the final sequence regions.

Several datasets exhibit pronounced surprisal peaks approaching or surpassing 2.5, suggesting the presence of structurally diverse or weakly conserved nucleotide subsequences near the terminal region of the gene. These elevated surprisal values indicate positions where the model encounters reduced contextual consistency, thereby lowering prediction confidence. In contrast, stretches of comparatively low surprisal values (< 1.0) correspond to conserved sequence segments that are more effectively captured by the learned contextual representations of the model.

A notable feature consistently observed across all profiles is the abrupt decline of surprisal values to nearly zero toward the extreme end of the analyzed region. This terminal flattening likely reflects sequence boundary effects or padding-related behavior in the transformer context window, where the model encounters reduced uncertainty due to limited downstream contextual dependencies. Overall, the last 10% region exhibits a heterogeneous surprisal landscape, combining conserved low-uncertainty regions with sporadic highly uncertain nucleotide positions, thereby indicating a mixture of stable and variable sequence organization near the terminal portion of the *hsf-1* gene.

5.3.5 Interpretation of Surprisal Profiles

The surprisal profiles generated from the transformer predictions provide direct insight into sequence-level contextual organization.

- Regions with consistently low surprisal correspond to stable and contextually constrained nucleotide segments.
- Regions with repeated high surprisal peaks indicate positions with weaker contextual conservation or increased sequence variability.
- Continuous moderate surprisal distributions suggest long-range contextual dependencies learned successfully by the transformer model.
- Similar surprisal trends across temperature conditions indicate preservation of genomic contextual structure despite environmental variation.

Overall, the surprisal-based framework enabled quantitative characterization of nucleotide predictability and revealed conserved contextual behavior within temperature-dependent genomic sequences.

Chapter 6

Results and Discussion

6.1 Overview

This chapter presents the consolidated results of the computational nucleotide prediction framework applied to three model organisms and gene targets: HSF1 (Heat Shock Factor 1 — a large, multi-kilobase gene), HSP-16.2 (Heat Shock Protein 16.2 — a short gene), and ISY-1 (Integrator Complex Subunit — an intermediate-length gene). The analysis spans two complementary dimensions: (i) sliding-window consistency analysis, which identifies genomic regions where predicted nucleotide positions agree across two independent transcript versions (T2 and T3) for each of four strand variants (S1–S4); and (ii) surprisal analysis at competitive positions derived from metadata, which characterises the information-theoretic uncertainty of the prediction model at positions where at least two nucleotide bases achieve near-equal probability.

Together, these analyses serve to validate the robustness and biological relevance of the prediction pipeline and to identify high-confidence genomic windows that may be of functional interest.

6.2 Sliding-Window Consistency Analysis

The sliding-window analysis compares predicted base calls at each genomic index between two transcript sources (T2 and T3) across four strand variants. A position is counted as a ‘correct index’ within a window if the predicted base at that position matches across both transcript versions, thereby serving as a proxy for cross-transcript reproducibility. The most robust window is defined as the 100-position (or 150-position, depending on gene length) sliding window containing the highest number of such concordant predictions.

6.2.1 HSF1 — Large Gene Analysis

HSF1 represents the largest gene in this study, with index ranges extending to approximately 8,476 positions. The sliding-window analysis across all four strand variant comparisons reveals a consistently high number of total correct indices, ranging from 660 (S2) to 962 (S3), reflecting a gene-wide reproducibility rate significantly above

chance. Notably, the distribution of correct indices across windows for HSF1 is bimodal in all four strand comparisons, with a prominent early peak (approximately indices 1200–2400) and a secondary elevation in the mid-to-late genome region (approximately 4500–8000).

The most robust window across all strand variants converges in the 1200–2400 index range, with the peak correct index count in a single window ranging from 52 (S4) to 77 (S3). S1 and S2 show their peaks at windows 2000–2400 and 1300–1700 respectively, while S3 and S4 peak at 1300–1700 and 1200–1600. The consistency of peak windows across three of the four strand variants (S2, S3, S4) in the 1200–1700 zone strongly suggests that this region harbours functionally constrained positions where the model assigns stable, reproducible base probabilities regardless of strand orientation.

This convergence in HSF1 has biological plausibility: the early coding region of HSF1, which encodes the DNA-binding and trimerisation domains, is expected to be under strong purifying selection, resulting in lower nucleotide entropy and more predictable base calls. The secondary elevation toward the 3' end may correspond to regulatory regions or alternatively spliced exons.

6.2.2 HSP-16.2 — Short Gene Analysis

HSP-16.2 is a substantially shorter gene (index range approximately 1–624), and the total correct indices are correspondingly lower (52–81 across strand variants). However, the concordance rates — when normalised to gene length — are comparable to or exceed those of HSF1, suggesting that the prediction model performs reliably on short genes.

The most robust windows for HSP-16.2 are located in two distinct zones. Strand variants S1 and S2 share their peak in the 350–520 index range (windows 350–450 and 420–520, with 15 and 21 correct indices respectively), whereas S3 and S4 peak earlier in the 170–280 range (14 correct indices each). This bifurcation between the early and late robust windows across strand variants is consistent with the hypothesis that different strand orientations capture different functional regions of the same gene — the 3' regulatory region being more consistent in the forward strand comparisons, and the 5' promoter-proximal region being more consistent in reverse-complement comparisons.

The relatively high peak correct index count for S2 (21 in window 420–520) is noteworthy: in a gene spanning only ~ 620 positions, achieving 21 concordant predictions in a single 100-position window represents a local concordance rate of approximately 21%, far exceeding the genome-wide rate for this gene ($81/624 \approx 13\%$). This elevated local concordance strongly implies the presence of a conserved functional

motif in this region, potentially a heat shock response element (HSE) or a 3' UTR regulatory sequence.

6.2.3 ISY-1 — Intermediate Gene Analysis

ISY-1 spans approximately 1,037 positions and exhibits total correct index counts ranging from 126 (S4) to 149 (S2). The distribution of correct indices across windows is notably flatter and more uniform compared to HSF1 and HSP-16.2, suggesting a more distributed pattern of cross-transcript reproducibility without a single dominant hotspot.

However, meaningful peaks are still identifiable. Strand variants S1 and S2 both share their most robust window at 890–1040, representing the terminal region of the gene, with 29 and 32 correct indices respectively. This is a striking result: the convergence of the peak window at the very end of the gene across two independent strand comparisons implies that the 3' end of ISY-1 carries a disproportionately high density of reproducibly predicted positions. Biologically, this could reflect a conserved 3' processing signal or polyadenylation regulatory element.

In contrast, S3 and S4 peak at 320–470 and 330–480 respectively (both with 26 correct indices), clustering around the central third of the gene. This mid-gene robustness in the reverse-complement strand comparisons may indicate the presence of an internal conserved exon boundary or splice donor/acceptor site in this region.

6.3 Comparative Summary of Window Robustness

Across all three genes, several cross-gene patterns emerge from the sliding-window analysis:

1. HSF1 consistently shows the highest absolute correct index counts (660–962), driven by its greater gene length. Normalised for length, HSP-16.2 and ISY-1 are comparably informative.
2. All three genes exhibit at least partial convergence of the most robust window across strand variants, indicating that the prediction pipeline identifies gene-specific regions of stability rather than distributing correct predictions uniformly.
3. The most robust windows in forward-strand (S1/S2) comparisons and reverse-complement (S3/S4) comparisons tend to occupy different genomic zones within the same gene, consistent with the strand-asymmetric nature of regulatory signals in eukaryotic genomes.

4. The window peak correct index values are generally consistent between S1/S2 and between S3/S4 within the same gene, indicating that paired strand variants provide mutually corroborating evidence for local reproducibility.
5. HSP-16.2 S2 achieves the highest normalised local concordance rate ($\sim 21\%$ in its peak window), making it the most locally reproducible region identified in this study.

Table 6.1: Summary of sliding-window consistency analysis across all genes and strand variants.

Gene / File	Strand Pair	Total Correct Indices	Most Robust Window
HSF1	T2-S1 vs T3-S1	788	2000–2400
HSF1	T2-S2 vs T3-S2	660	1300–1700
HSF1	T2-S3 vs T3-S3	962	1300–1700
HSF1	T2-S4 vs T3-S4	733	1200–1600
HSP-16.2	T2-S1 vs T3-S1	60	350–450
HSP-16.2	T2-S2 vs T3-S2	81	420–520
HSP-16.2	T2-S3 vs T3-S3	52	180–280
HSP-16.2	T2-S4 vs T3-S4	54	170–270
ISY-1	T2-S1 vs T3-S1	135	890–1040
ISY-1	T2-S2 vs T3-S2	149	890–1040
ISY-1	T2-S3 vs T3-S3	133	320–470
ISY-1	T2-S4 vs T3-S4	126	330–480

6.4 Surprisal Analysis at Competitive Positions

The surprisal analysis was conducted on HSF1 File 11, a representative data file selected for detailed metadata examination. Surprisal, defined in the context of this study as the negative log-base-2 probability of the predicted nucleotide at a given position, quantifies the model’s uncertainty: a surprisal of 1.0 bit corresponds to a true 50/50 prediction (maximum binary uncertainty), while lower values indicate greater model confidence and higher values indicate the model assigned lower probability to the observed outcome.

Competitive positions are operationally defined as those positions where the top two nucleotide probability estimates are within a defined competitive range, reflecting genuinely ambiguous predictions.

6.5 Biological Significance of Prediction Patterns

The robust prediction windows identified across HSF1, HSP-16.2, and ISY-1 likely correspond to biologically conserved genomic regions under functional constraint. In HSF1, the strongest reproducibility occurs within regions associated with essential DNA-binding and stress-response functions, while the terminal robust regions in ISY-1 may reflect conserved transcript-processing elements. Similarly, the highly reproducible region observed in HSP-16.2 may correspond to conserved heat-shock regulatory motifs.

In contrast, competitive positions with high surprisal values are more likely associated with AT-rich, non-coding, or repetitive genomic regions where nucleotide context provides weaker predictive certainty. The predominance of near 50/50 A/T competition observed in the metadata analysis is consistent with the known AT-rich composition of many non-coding regions in *C. elegans*. These findings suggest that the transformer model captures biologically meaningful sequence constraints in addition to statistical nucleotide patterns.

Chapter 7

Conclusion

This study demonstrates that transformer-based nucleotide prediction can recover biologically meaningful sequence structure directly from genomic data without requiring explicit annotation. Across the three analysed genes — HSF1, HSP-16.2, and ISY-1 — the prediction framework consistently identified robust genomic windows with elevated cross-transcript concordance, suggesting the presence of conserved and functionally constrained regions. The alignment of these robust windows with biologically important regions, such as the DNA-binding region of HSF1 and regulatory regions of HSP-16.2 and ISY-1, indicates that the model captures genuine sequence-level conservation patterns.

The surprisal analysis further revealed that most competitive positions cluster near the theoretical 50/50 uncertainty boundary, reflecting genuine ambiguity in nucleotide prediction. The predominance of A/T competitive positions is consistent with the AT-rich nature of many non-coding genomic regions in *C. elegans*. Positions exhibiting surprisal values above 1.0 bit represent highly uncertain regions where contextual information alone is insufficient for confident nucleotide discrimination.

From a computational perspective, the integration of sliding-window consistency analysis, metadata extraction, and surprisal scoring provides a detailed framework for analysing transformer prediction behaviour beyond conventional accuracy metrics. The results show that prediction reproducibility is spatially non-uniform and strongly dependent on local genomic context, indicating that the model effectively captures contextual nucleotide dependencies.

Overall, this work establishes a biologically grounded computational framework for masked nucleotide prediction and uncertainty quantification. The findings demonstrate both the strengths and limitations of transformer-based genomic modelling and provide a foundation for future large-scale genomic prediction and annotation studies.

Appendix A

Implementation

This chapter describes the software implementation of the genomic Masked Language Model (MLM) used in this thesis. The model is based on the BERT architecture and is trained from scratch on raw DNA sequences. All code is written in Python 3 and uses the PyTorch and HuggingFace Transformers libraries.

A.1 Overview of the Pipeline

The pipeline consists of five stages:

1. **Data ingestion** — FASTA files are read and RNA bases (U) are silently mapped to T, producing a plain DNA string.
2. **Tokenisation** — A lightweight character-level tokeniser converts each nucleotide to an integer index drawn from the vocabulary $\{\text{A, C, G, T, N, [MASK]}\}$.
3. **Dataset construction** — A sliding-window strategy tiles each sequence into fixed-length windows of $W = 104$ bases with stride $s = 26$.
4. **Model training** — A small BERT model is trained with a standard 15% random masking objective over 6 000 outer epochs.
5. **Evaluation** — Per-position surprisal, per-base probabilities, and masked-token accuracy are computed on held-out sequences and written to CSV files.

A.2 Environment Setup and Reproducibility

To ensure reproducibility, all stochastic components (Python `random`, NumPy, PyTorch CPU, and CUDA) are seeded at the start of the script and re-seeded at the beginning of each outer epoch.

```
1 import torch, random, numpy as np
2
3 device = "cuda" if torch.cuda.is_available() else "cpu"
4
```

```

5 def set_all_seeds(seed):
6     random.seed(seed)
7     np.random.seed(seed)
8     torch.manual_seed(seed)
9     torch.cuda.manual_seed_all(seed)

```

Listing A.1: Device selection and global seed function.

A.3 Data Ingestion

Sequences are stored in standard FASTA format. Listing A.2 shows the reader that strips header lines, concatenates sequence lines, and normalises bases to uppercase DNA.

```

1 def read_fasta_as_string(path):
2     seq_parts = []
3     with open(path, "r") as f:
4         for line in f:
5             if not line.startswith(">"):
6                 seq_parts.append(line.strip())
7     return "".join(seq_parts).upper().replace("U", "T")

```

Listing A.2: FASTA reader with RNA-to-DNA normalisation.

A.4 Tokeniser

Because the vocabulary is tiny (five nucleotides plus one special token), a hand-written character tokeniser is sufficient and avoids the overhead of a full sub-word tokeniser.

```

1 class CharTokenizer:
2     def __init__(self):
3         self.bases = ["A", "C", "G", "T", "N"]
4         self.special = ["[MASK]"]
5         self.tokens = self.bases + self.special
6         self.vocab = {t: i for i, t in
7                       enumerate(self.tokens)}
8         self.inv = {i: t for t, i in self.vocab.items()}
9         self.mask_token = "[MASK]"
10
11     def encode(self, seq):
12         # Unknown characters are mapped to N (index 4)

```

```

12         return [self.vocab.get(c, self.vocab["N"]) for c in
13                 seq]
14
15     @property
16     def vocab_size(self):
17         return len(self.vocab)

```

Listing A.3: Character-level tokeniser for DNA sequences.

A.5 Dataset and Data Collator

The `GenomeSequenceDataset` class (Listing A.4) tiles every sequence into overlapping windows and stores them as pre-tokenised tensors.

The `MLMDataCollator` (Listing A.5) implements the standard BERT masking strategy: 15% of tokens in each batch are replaced with `[MASK]`; the remaining positions receive a label of `-100` so that the cross-entropy loss ignores them.

```

1  from torch.utils.data import Dataset
2
3  class GenomeSequenceDataset(Dataset):
4      def __init__(self, sequences, tokenizer,
5                  window_size=512, stride=128):
6          self.samples = []
7          for seq in sequences:
8              for i in range(0,
9                          max(1, len(seq) - window_size +
10                             1),
11                          stride):
12                  window = seq[i:i + window_size]
13                  self.samples.append(
14                      torch.tensor(tokenizer.encode(window),
15                                  dtype=torch.long)
16                  )
17
18      def __len__(self):
19          return len(self.samples)
20
21      def __getitem__(self, idx):
22          return self.samples[idx]

```

Listing A.4: Sliding-window genome dataset.

```

1 class MLMDDataCollator:
2     def __init__(self, tokenizer, mask_prob=0.15):
3         self.mask_id =
4             tokenizer.vocab[tokenizer.mask_token]
5         self.mask_prob = mask_prob
6
7     def __call__(self, batch):
8         input_ids = torch.stack(batch)
9         labels = input_ids.clone()
10
11         mask = torch.bernoulli(
12             torch.full(labels.shape, self.mask_prob)
13         ).bool()
14
15         labels[~mask] = -100 # ignore unmasked
16             positions
17         input_ids[mask] = self.mask_id # replace with
18             [MASK]
19
20         return {"input_ids": input_ids, "labels": labels}

```

Listing A.5: MLM data collator with 15% random masking.

A.6 Model Architecture

A compact BERT model is initialised from scratch using the HuggingFace `BertForMaskedLM` class (Listing A.6). The hyper-parameters are chosen to keep the model small enough to train on a single CPU or modest GPU while still capturing local sequence context within the 104-token window.

Hyper-parameter	Value
Hidden size	64
Number of layers	6
Attention heads	8
Intermediate (FFN) size	256
Max position embeddings	104
Vocabulary size	6

Table A.1: BERT model configuration used in this work.

```

1 from transformers import BertConfig, BertForMaskedLM
2

```

```

3 def make_bert_mlm(vocab_size, max_len):
4     return BertForMaskedLM(
5         BertConfig(
6             vocab_size          = vocab_size,
7             hidden_size        = 64,
8             num_hidden_layers  = 6,
9             num_attention_heads = 8,
10            intermediate_size   = 256,
11            max_position_embeddings = max_len,
12        )
13    )

```

Listing A.6: BERT model factory function.

A.7 Training Loop

Training follows a two-level loop structure. An *outer loop* runs for 6 000 iterations and re-seeds the random number generators at each step; an *inner loop* (delegated to HuggingFace Trainer) performs one full pass through the dataset per outer iteration. This design makes it straightforward to checkpoint or inspect the model at any coarse epoch boundary.

```

1 from transformers import TrainingArguments, Trainer
2
3 OUTER_EPOCHS = 6000
4 INNER_EPOCHS = 1
5 BATCH_SIZE   = 8
6 LR           = 4e-4
7
8 for outer_epoch in range(OUTER_EPOCHS):
9     set_all_seeds(42 + outer_epoch) # reproducible
10                                     shuffling
11
12     training_args = TrainingArguments(
13         output_dir          = "./mlm_reference",
14         num_train_epochs    = INNER_EPOCHS,
15         per_device_train_batch_size = BATCH_SIZE,
16         learning_rate       = LR,
17         weight_decay        = 0.01,
18         logging_strategy    = "epoch",
19         report_to           = "none",

```

```

19     )
20
21     trainer = Trainer(
22         model          = model ,
23         args           = training_args ,
24         train_dataset= train_dataset ,
25         data_collator= collator ,
26     )
27     trainer.train()

```

Listing A.7: Outer training loop with per-epoch re-seeding.

A.8 Evaluation Metrics

Three evaluation functions are used after training.

A.8.1 Per-Position Surprisal

Surprisal at position i is the negative log-probability assigned by the model to the true nucleotide when that position is masked:

$$S_i = -\log P(x_i \mid x_1, \dots, x_{i-1}, [\text{MASK}], x_{i+1}, \dots).$$

Because each position may be covered by multiple overlapping windows, the final surprisal is the mean over all covering windows (Listing A.8).

```

1  import torch.nn.functional as F
2
3  def surprisal_per_position_windowed(
4      model, sequence, tokenizer,
5      window_size, stride, device):
6      model.eval()
7      mask_id = tokenizer.vocab[tokenizer.mask_token]
8      encoded = torch.tensor(tokenizer.encode(sequence),
9                             dtype=torch.long)
10     seq_len = len(encoded)
11     surprisal_sum = np.zeros(seq_len)
12     surprisal_count = np.zeros(seq_len)
13
14     with torch.no_grad():
15         for start in range(0,

```

```

16             max(1, seq_len - window_size +
17                 1),
18             stride):
19         window =
20             encoded[start:start+window_size].to(device)
21         window_len = len(window)
22
23         for pos in range(window_len):
24             global_pos = start + pos
25             original_id = window[pos].item()
26
27             masked_window = window.clone()
28             masked_window[pos] = mask_id
29
30             logits =
31                 model(masked_window.unsqueeze(0)).logits[0,
32                     pos]
33             log_probs = F.log_softmax(logits, dim=-1)
34
35             surprisal_sum[global_pos] +=
36                 -log_probs[original_id].item()
37             surprisal_count[global_pos] += 1
38
39     valid = surprisal_count > 0
40     mean_surprisal = np.zeros(seq_len)
41     mean_surprisal[valid] = (surprisal_sum[valid]
42                             / surprisal_count[valid])
43     return mean_surprisal

```

Listing A.8: Windowed per-position surprisal computation.

A.8.2 Unified Metadata (Probabilities + Surprisal)

The `metadata_per_position_windowed` function (Listing A.9) returns, for every position, the mean softmax probability for each nucleotide, the argmax prediction, and the surprisal. Results are written to a CSV with columns `Index`, `True_Base`, `P_A`, `P_C`, `P_G`, `P_T`, `P_N`, `Predicted_Base`, `Surprisal`.

```

1 def metadata_per_position_windowed(
2     model, sequence, tokenizer,
3     window_size, stride, device):
4     model.eval()

```

```

5     mask_id = tokenizer.vocab[tokenizer.mask_token]
6     encoded = torch.tensor(tokenizer.encode(sequence),
7                             dtype=torch.long)
8     seq_len = len(encoded)
9     bases = ["A", "C", "G", "T", "N"]
10    base_ids = [tokenizer.vocab[b] for b in bases]
11
12    prob_sum = np.zeros((seq_len, len(bases)))
13    surprisal_sum = np.zeros(seq_len)
14    count = np.zeros(seq_len)
15
16    with torch.no_grad():
17        for start in range(0,
18                            max(1, seq_len - window_size +
19                                1),
20                            stride):
21            window =
22                encoded[start:start+window_size].to(device)
23            window_len = len(window)
24
25            for pos in range(window_len):
26                global_pos = start + pos
27                original_id = window[pos].item()
28
29                masked_window = window.clone()
30                masked_window[pos] = mask_id
31
32                logits =
33                    model(masked_window.unsqueeze(0)).logits[0,
34                        pos]
35                probs = F.softmax(logits, dim=-1)
36                log_probs = F.log_softmax(logits, dim=-1)
37
38                for k, bid in enumerate(base_ids):
39                    prob_sum[global_pos, k] +=
40                        probs[bid].item()
41
42                surprisal_sum[global_pos] +=
43                    -log_probs[original_id].item()
44                count[global_pos] += 1

```

```

40     results = []
41     for idx in range(seq_len):
42         true_base = tokenizer.inv[encoded[idx].item()]
43         if count[idx] > 0:
44             mean_probs = prob_sum[idx] / count[idx]
45             mean_surprisal = surprisal_sum[idx] / count[idx]
46         else:
47             mean_probs = np.zeros(len(bases))
48             mean_surprisal = 0.0
49
50         predicted_base = bases[int(np.argmax(mean_probs))]
51         row = {"Index": idx + 1, "True_Base": true_base}
52         for k, b in enumerate(bases):
53             row[f"P_{b}"] = round(float(mean_probs[k]), 6)
54         row["Predicted_Base"] = predicted_base
55         row["Surprisal"] = round(mean_surprisal, 6)
56         results.append(row)
57
58     return results

```

Listing A.9: Unified per-position metadata function.

A.8.3 Masked Prediction Accuracy

Overall accuracy is estimated by randomly masking 15% of tokens in each window and measuring how often the model's argmax prediction matches the true nucleotide (Listing A.10).

```

1  def masked_prediction_accuracy(
2      model, sequence, tokenizer,
3      mask_prob, window_size, stride, device):
4      model.eval()
5      mask_id = tokenizer.vocab[tokenizer.mask_token]
6      encoded = torch.tensor(tokenizer.encode(sequence),
7                             dtype=torch.long)
8      seq_len = len(encoded)
9      correct, total = 0, 0
10
11     with torch.no_grad():
12         for start in range(0,
13                               max(1, seq_len - window_size +

```

```

14         stride):
15         window
            =
            encoded[start:start+window_size].to(device)
16         window_len      = len(window)
17         mask_positions  = torch.rand(window_len) <
            mask_prob
18
19         masked_window   = window.clone()
20         masked_window[mask_positions] = mask_id
21
22         logits          =
            model(masked_window.unsqueeze(0)).logits[0]
23         predictions     = torch.argmax(logits, dim=-1)
24
25         for i in range(window_len):
26             if mask_positions[i]:
27                 total    += 1
28                 correct += int(predictions[i].item()
29                               == window[i].item())
30
31         return correct / total if total > 0 else 0.0

```

Listing A.10: Masked-token prediction accuracy.

A.9 Configuration and Output

Listing A.11 summarises the key hyper-parameters and Listing A.12 shows the inference and file- writing loop that produces a `_metadata.csv` and an `_accuracy.txt` file for each test sequence.

```

1 DATA_FOLDER    = "data"
2 OUTPUT_FOLDER   = "batch_outputs"
3
4 TRAIN_STEM      = "HSF-1 (1)"
5 TEST_STEMS      = ["HSF-1 (5)", "HSF-1 (9)"]
6
7 WINDOW_SIZE     = 104    # nucleotides per window
8 STRIDE          = 26     # overlap step
9 BATCH_SIZE      = 8
10 LR              = 4e-4
11 INNER_EPOCHS   = 1

```

```
12 OUTER_EPOCHS = 6000
```

Listing A.11: Experiment configuration constants.

```
1 import csv, os
2
3 for stem in TEST_STEMS:
4     file_path = find_fasta_file(DATA_FOLDER, stem)
5     if file_path is None:
6         print(f"Skipping {stem}: file not found.")
7         continue
8
9     sequence = read_fasta_as_string(file_path)
10
11     # --- per-position metadata CSV ---
12     metadata = metadata_per_position_windowed(
13         model, sequence, tokenizer, WINDOW_SIZE, STRIDE,
14         device
15     )
16     fieldnames = ["Index", "True_Base",
17                  "P_A", "P_C", "P_G", "P_T", "P_N",
18                  "Predicted_Base", "Surprisal"]
19     out_path = os.path.join(OUTPUT_FOLDER,
20                             f"{stem}_metadata.csv")
21     with open(out_path, "w", newline="") as f:
22         writer = csv.DictWriter(f, fieldnames=fieldnames)
23         writer.writeheader()
24         writer.writerows(metadata)
25
26     # --- overall masked accuracy ---
27     accuracy = masked_prediction_accuracy(
28         model, sequence, tokenizer,
29         mask_prob=0.15,
30         window_size=WINDOW_SIZE,
31         stride=STRIDE,
32         device=device,
33     )
34     acc_path = os.path.join(OUTPUT_FOLDER,
35                             f"{stem}_accuracy.txt")
36     with open(acc_path, "w") as f:
37         f.write(f"Masked Prediction Accuracy:
38                 {accuracy:.4f}\n")
```

Listing A.12: Test-set inference and result export.

A.10 Summary

This chapter presented a complete, self-contained implementation of a BERT-based masked language model trained on genomic sequences. The pipeline covers data loading, tokenisation, dataset construction, model definition, training, and evaluation, with all random states controlled for reproducibility. The resulting per-position surprisal and nucleotide probability estimates are used in Chapter 5 to analyse sequence conservation and structural constraints.

References

- [1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). *Attention Is All You Need*. Advances in Neural Information Processing Systems, 30.
- [2] Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Proceedings of NAACL-HLT 2019.
- [3] Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., and Ahmed, A. (2020). *Big Bird: Transformers for Longer Sequences*. Advances in Neural Information Processing Systems, 33, 17283–17297.
- [4] Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. (2021). *DNABERT: Pre-trained Bidirectional Encoder Representations from Transformers for DNA-language in Genome*. Bioinformatics, 37(15), 2112–2120.
- [5] Dalla-Torre, H., et al. (2023). *Nucleotide Transformer: A Pretrained Model for Genomics*. Nature Methods, 20, 1703–1710.
- [6] Dalla-Torre, H., Gonzalez, L., Mendoza-Revilla, J., Carranza, N. L., Grzywaczewski, A. H., Oteri, F., Dallago, C., Trop, E., de Almeida, B. P., Sirelkhatim, H., Richard, G., Skwark, M., Beguir, K., Lopez, M., and Pierrot, T. (2023). *The Nucleotide Transformer: Building and Evaluating Robust Foundation Models for Human Genomics*. bioRxiv.
- [7] The C. elegans Sequencing Consortium (1998). *Genome Sequence of the Nematode C. elegans: A Platform for Investigating Biology*. Science, 282(5396), 2012–2018.
- [8] Morimoto, R. I. (1998). *Regulation of the Heat Shock Transcriptional Response: Cross Talk Between Heat Shock Factors, Molecular Chaperones, and Negative Regulators*. Genes & Development, 12(24), 3788–3796.