Analysis of GPN and its applicability to distant species

Master Thesis

Author: Silvan Tristan Büdenbender (Student ID: 7412199)
Supervisor: Prof. Dr. Thomas Wiehe
Co-Supervisor: Prof. Dr. Gereon Frahling

Department of Computer Science
Faculty of Mathematics and Natural Sciences
University of Cologne

23.06.2025

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten fremden Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

Silvan Tristan Büdenbender

Köln, den 23.06.2025

Abstract

In this work the DNA language model named Genomic Pretrained Network (GPN) is studied under different aspects. The architecture of the model is examined from a computational point of view unveiling a potentially novel perspective on exponentially dilated convolutions. Further, the models inference behavior is probed with the conclusion that the model stops utilizing context beyond 1 kilo base of genomic distance. Lastly, the model is evaluated on DNA data from the beetle model organism *Tribolium castaneum* to compare the existing GPN model trained on several *Brassicales* species with two new GPN models trained on several *Cucujiformia* species. These beetle specific models outperform a plant specific model on beetle DNA data. All code used in this work can be found at `https://github.com/SilvanCodes/masterthesis-ramses`.

Disclaimer

All parts of this work might at some point have been part of conversations with Claude from Antrophic or ChatGPT from OpenAI, mostly the o3 model, which was used extensively for literature research as well as for discussion of ideas in development and to generate parts of the source code used in this project. All writing in this document has only been edited by the author without incorporating the above mentioned tools.

Contents

List of Figures

List of Tables

List of Acronyms

CDS
coding sequence 16

CNN
convolutional neural network 2

FFN
feedforward network 4

GELU
Gaussian Error Linear Unit 4

GPN
Genomic Pretrained Network . . . . . .

LLM
large language model 1

LM
language model 1

NDM
Nucleotide Dependency Map 4

RF
receptive field 5

RL
reinforcement learning 1

RLHF
reinforcement learning from human feedback 1

SNP
single nucleotide polymorphism 10

SSM
state-space model 2

VEP
variant effect prediction 2

1 Introduction

With the recent rise and success of large language models (LLMs), sometimes just language models (LMs), such as the GPT and Llama families, other fields of science apart from natural language processing and generation have started to experiment with such Transformer-based machine learning models.

LLMs have shown unprecedented capabilities in modeling conditional probability distributions, which in the setting of natural language is often translated to the following question: Given this start of a sentence, conversation, paragraph of an article, how is it likely to be continued?

Most humans with internet access these days have witnessed the remarkable power of this approach first hand with more and more capable chatbots being available on their own as well as being integrated into many products right away.

1.1 Large Language Models

The initial breakthrough was the attention mechanism (Vaswani et al., 2017) introducing the archetypal, now famous, Transformer architecture. It has to be noted that attention was developed over preceding works already (Bahdanau et al., 2014), (Wu et al., 2016).

To add some nuance to the initial description of the current success of LMs, it is important to realize observed capabilities, especially in human-LM interactions, result from more than the plain language-modeling task. Language modeling itself is only concerned with teaching the model to reconstruct sentences from the training data exactly. This initial training on a huge volume of data, often called pre-training, which currently is mostly encoder-only (masked language modeling) or decoder-only (autoregressive language modeling) results in a base model.

This base model then is usually further fine-tuned to a task at hand, benefiting from the rich internal representation the LM has learned. One such example is InstructGPT (Ouyang et al., 2022), which has been fine-tuned first on a hand-crafted dataset of text to preferably generate. Following this fine-tuning they employed reinforcement learning from human feedback (RLHF) to further align the generated text with a notion of human preference.

Reinforcement learning (RL) is starting to play an increasingly large role in pushing the capabilities of LMs as it can be applied to any verifiable task, e.g. writing software that passes a test. One recent impressive application is from DeepSeek-AI et al. (2025) where RL is used to develop reasoning capacity in the model.

1.2 DNA as language

One compelling subject to apply LLMs to is the comprehension of DNA data, for a multitude of reasons. First and most obvious is the fact that DNA, identical to natural language, is represented as a string of characters (A, C, G, T) called nucleotides. Further, some might argue that DNA itself can be interpreted as a kind of language (Dong & Searls, 1994), (Sanabria et al., 2024).

Another very important reason is that LMs are trained on unlabelled data, i.e. just the string of characters itself is required as training data. For DNA data that is extremely relevant as labelled data is very sparse and costly to obtain, mostly only via experimental means in the wet lab. Labels for DNA are usually only available for a few model species and their respective reference genomes.

Generating relatively high quality DNA sequences of individual organisms on the contrary has become comparatively cheap in the recent years (NHGRI, 2022). So due to the high availability of unlabeled data and the costly acquisition of labels an unsupervised modeling approach is highly favorable.

Furthermore, LMs are a proper fit as the raw statistics the model initially learns, without any fine-tuning, are already highly informative and applicable to predictions with biological relevance such as variant effect prediction (VEP) (Benegas et al., 2023), motif discovery and secondary as well as tertiary RNA structure prediction (da Silva et al., 2024).

Some of the first works exploring the application of LMs to DNA data are purely Transformer-based models like DNABERT (Ji, Zhou, Liu, & Davuluri, 2020), (Zhou et al., 2023) and Nucleotide Transformer (Dalla-torre et al., 2024).

As DNA data from genomes easily spans millions of nucleotides and as single nucleotide resolution, i.e. one token per nucleotide, is favorable for precise biological interpretation, the interest to work with sequences of up to millions of tokens is high, aiming to detect and learn genome wide interactions. Attention as a mechanism struggles in that regard, as its default implementation (Vaswani et al., 2017) comes with quadratic scaling of computation cost in sequence length.

Models like Hyena-DNA (Nguyen et al., 2023) and Caduceus (Schiff et al., 2024) employ state-space models (SSMs) with sub-quadratic scaling to match the needs of long sequence modelling. Other architectures like Borzoi (Linder, Srivastava, Yuan, Agarwal, & Kelley, 2023) employ a mix of convolutional neural networks (CNNs) with attention or recently Evo2 (Brixi et al., 2025) a mix of SSMs with attention.

2 Genomic Pretrained Network

In this research we follow the work from Benegas et al. (2023) employing an interesting Transformer/CNN-hybrid architecture similar to Yang, Lu, and Fusi (2024). The model was trained on several genomes of the *Brassicales* order of plants.

2.1 Architecture Overview

The GPN architecture depicted in Figure 1 generally follows the shape of a Transformer but the rather costly self-attention mechanism is substituted with a one-dimensional convolution.

The model vocabulary is constituted of four tokens, one for each nucleotide "a", "c", "g", "t" and the three special tokens "[UNK]", "[MASK]" and "[PAD]". The tokens are one-hot encoded into a 512 dimensional space as the embedding. It can be argued that this is sufficient in comparison to learnable embeddings with regard to the vocabulary size, as a single nucleotide allows for very little semantics to be embedded (Sanabria, Hirsch, & Poetsch, 2023).

The initial embedding layer is followed by 25 Transformer-like blocks, labeled Dilation Block in Figure 1. Each block begins with a layer norm before a one-dimensional convolutional layer which expects 512 channels input and constructs 512 channels output. The number of output channels is the number of filters that are applied to our input sequence. Each filter has a kernel size of 5, except for the first one which has kernel size 9, a stride of 1 and exponentially increasing dilation. The dilation follows the sequence 1, 2, 4, 8, 16, 32, 64, 128 and then repeats. In the given constellation,

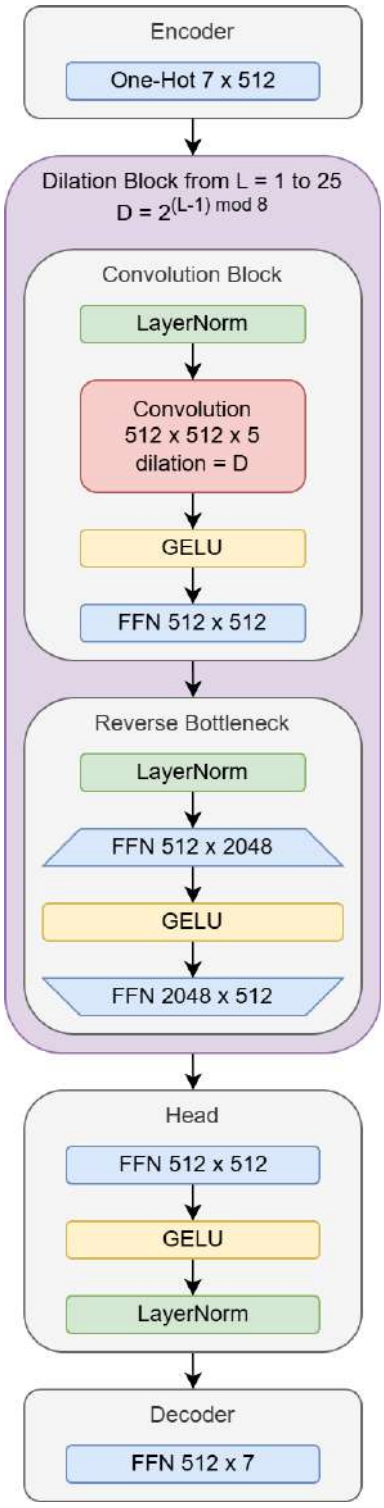


Figure 1: GPN Architecture Overview

the sequence fully repeats three times over the 25 blocks. The output of the convolution is fed through a Gaussian Error Linear Unit (GELU) activation function followed by a single layer feedforward network (FFN) concluding the first part of the block structure. Subsequently the second part of the block passes the signal again through a layer norm and a reverse-bottleneck two-layer FFN with intermediate dimension 2048 and a GELU activation in between. Over both parts of the block span residual connections and the block structure incorporates the more efficient pre-norm instead of the post-norm (Xiong et al., 2020).

Afterwards the head of the network does one more pass through a single FFN layer, GELU and layer norm. Embedding vectors from this layer carry all the accumulated semantic information the network has learned to infer from the provided input DNA per position in the input. These vectors can be used to run downstream analysis and modeling tasks.

Finally a decoder linear layer down-projects the signals from 512 dimensions to 7, the vocabulary size. The resulting signal per position in the sequence can then be interpreted as a probability distribution over the tokens by applying a soft-max function.

2.2 Architectural Remarks

With the overall architecture laid out we now turn to its closer examination.

2.2.1 Tokenization and Embedding

The choice of the character level nucleotide vocabulary is notable for its various implications on downstream analysis. It vastly simplifies interpretability as the output of the network can be a direct distribution over the four nucleotides.

Alternatives can be k-mers or Byte-Pair-Encoding examined by (Sanabria et al., 2024) in great detail. Interestingly they found that token frequency alone is a strong signal for some downstream prediction tasks.

While such tokenization strategies in combination with a learned embedding matrix allow to imbue each token with more upfront information, it can make downstream analysis, which leverages single nucleotide changes and predictions such as computing the GPN-Score (Sec. 2.4, Eq. 6) or Nucleotide Dependency Maps (NDMs) (da Silva et al., 2024), less straight forward or barely possible.

2.2.2 Dilated Convolutions

The exponential dilation schedule is an interesting choice as it has been shown to work well in the setting of context aggregation for dense prediction (per pixel semantic labeling) (Yu & Koltun, 2015) and capturing long-range dependencies in

DNA sequences (Gupta & Rush, 2017). It allows for an exponentially increasing receptive field (RF) with a linear increase in parameters.

A dilated convolution is a kernel that looks only at every n-th datapoint after the first one in each dimension, given it is symmetrically and regularly dilated. For DNA-LMs we are in the 1-D setting of applying a kernel over a tokenized and embedded DNA sequence.

A classical dense kernel is said to have a dilation of 1, it looks at every contiguous datapoint. In contrast, a kernel with dilation of 2 would start and then only look at every other datapoint, see Figure 2.

$$\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Figure 2: 3x1 Kernel, Dilation = 1 (top), Dilation = 2 (bottom)

It is now interesting to examine the relationship between the receptive field of parts of the network and the size of the kernel and the dilation factor. The RF describes to how many inputs one output is sensitive to. We make the simplifying assumption to only work with layers of successive convolutions, i.e. disregarding non-linearities and FFN parts of the network. This simplification is justified in the light of illuminating the token interactions which only manifest in the convolutions.

Generally speaking the receptive field RF of a kernel with size k and dilation factor d can be described as in Equation 1.

$$RF(k, d) = 1 + d(k - 1) \tag{1}$$

Theoretically, a large receptive field can be achieved either with a large kernel or a large dilation factor. A large kernel implies more parameters become necessary. A large dilation factor alone only expands the furthest data points covered but does not increase the amount of data points considered.

Inside the GPN and also other preceding architectures (van den Oord et al., 2016), (Kalchbrenner et al., 2016) an exponential dilation schedule over successive layers was used. The resulting receptive field RF_{exp} with exponential base b and number of layers n can be described as follows:

$$\begin{aligned} RF_{exp}(k, b, n) &= 1 + (k - 1) \sum_{l=0}^{n-1} b^l \\ &= k + (k - 1) \sum_{l=1}^{n-1} b^l \text{ for } n \geq 2 \end{aligned} \tag{2}$$

We should pay close attention to how kernel size k and dilation base b interact. It should be a goal to avoid skipping data points with respect to the input sequence akin to just increasing dilation directly. This effect of skipping data points is sometimes also referred to as "gridding" (P. Wang et al., 2017). A dilated convolution in isolation per definition always skips datapoints given a dilation factor $d \geq 2$. With respect to the input sequence, this does not have to be the case for successive dilated convolutions. If every point in the output sequence is influenced by one continuous stretch of the input sequence, we successfully avoided gridding.

Contrary to statements in P. Wang et al. (2017), who advise against dilation factors with a common factor relation ship as created by exponential dilation, this is achieved for all settings where $b \leq k$. This follows from the observation that initially, every data point in the input sequence holds information only of itself. We have to look at each token or we ignore some data. The first layer with initial dilation $d = b^0 = 1$ then aggregates data of k successive data points, i.e. each data point in the first hidden state of the network is influenced by k data points of the preceding state. Now we are able to aggregate anywhere between every single up to every k-th data point while still being influenced by a continuous stretch of the input sequence. In the maximal case of incorporating only every k-th data point, each position in the next hidden state is influenced by $k \cdot k = k^2$ continuous input positions. This can be extended to then taking anywhere between every single up to every k^2-th datapoint to remain influenced by a continuous stretch of the input sequence. By realizing taking every k-th datapoint, then every k^2-th datapoint, is the maximal case and exactly equal to the dilation of base $b = k$ in Equation 2, the initial statement follows.

The interaction of kernel size k and dilation base b can be further examined by the associativity of convolutions. Building on our simplifying assumption we can compute the effective kernel over n layers with exponential dilation by computing the convolution over all kernels as described in Equation 3.

$$K_{eff} = K_1 * K_2 * ... * K_n \tag{3}$$

By having each K_1 to K_n represent their dilation pattern as in Figure 2 with fixed weights of 1 and then computing K_{eff} we can observe the following: The size of K_{eff} is equal to the RF defined in Equation 2. Each specific component of the kernel tells us via how many paths that respective relative position in the input sequence influences a specific value at the output. This follows from the fact that one output signal position derived by the one input signal position simply recovers the kernel weight of the input signal position and as all weights are set

to 1 only additive interactions can increase the weights.

In dense convolutional neural networks this kernel turns out be a bell shaped curve, which is part of the reason for the locality bias in CNNs (Luo et al., 2016). The shape changes drastically when introducing dilation as shown in Figure 3.

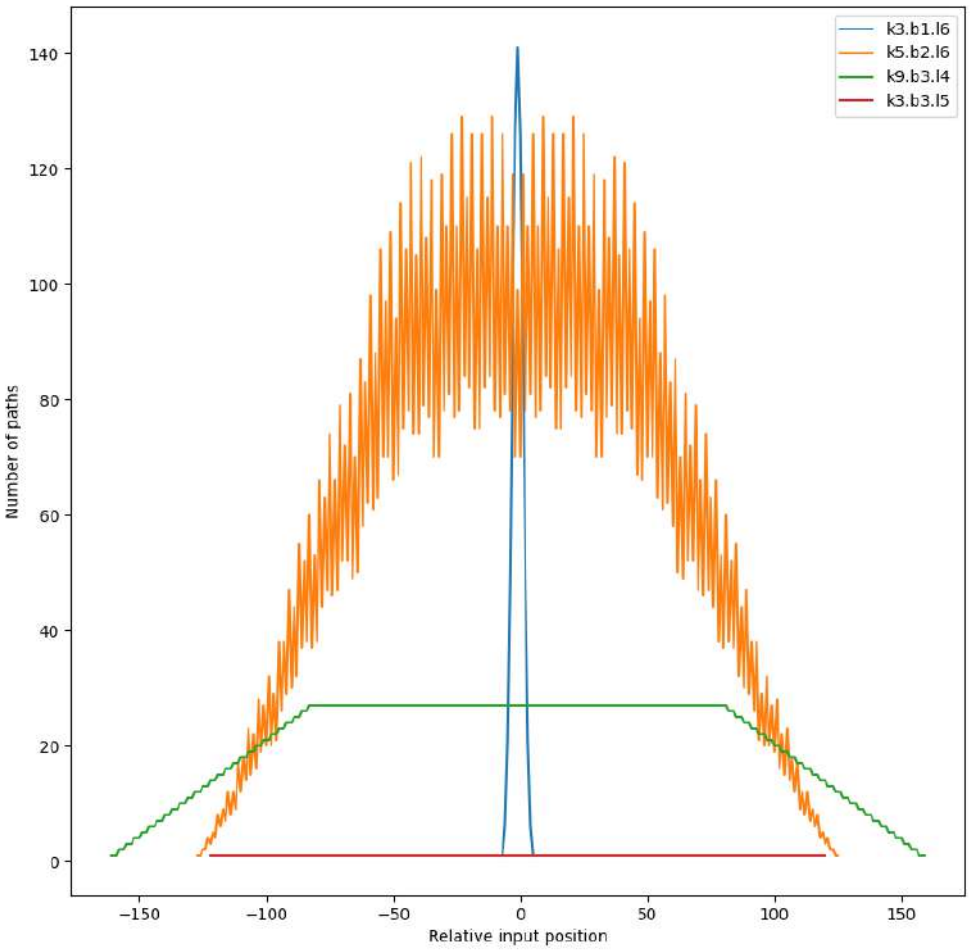


Figure 3: Path distributions for different kernel sizes k, dilation bases b and number of layers l listed as kx.by.lz, where x, y, z indicate the parameter value.

The blue line in Figure 3 shows the emerging bell shape with effectively no dilation ($b = 1$). Importantly, this highlights the limited linear growth of the RF represented by its narrow span in the graph and further the extreme focus of signal paths and thereby gradients on the center positions. The red line depicts the case of $b = k$, only the width of the RF grows. Many combinations of $b < k$ show patterns similarly to the orange line with wide RF and noisy paths distribution. An interesting case can be observed when $k = b^2$ shown as the green line with $k = 9$ and $b = 3$. The path distribution exhibits a "cut-off pyramid" shape where the central 50% of positions have equal number of path with linearly non-increasing flanks on both sides. This configuration could be of interest as it potentially helps gradient flow in contrast to the noisy cases.

Equipped with this knowledge we now compute the RF of the model given

Equation 2 and the model configuration. The RF of one dilation cycle in the current architecture can be computed to be of size 1021 (Eq. 4), totaling over all layers to 3069 (see Appendix 8).

$$\begin{aligned} RF_{exp}(5,2,8) &= 5 + (5-1)\sum_{l=1}^{8-1} 2^l \\ &= 5 + (4)(2+4+8+16+32+64+128) \\ &= 1021 \end{aligned} \tag{4}$$

The GPN architecture also has a "legacy" version depicted in Appendix 15. It is of interest as the GPN model for *Brassicales* has been trained with this architecture. The RF of one dilation cycle in the "legacy" version can be computed to be of size 505 (Eq. 5), totaling over all layers to 2025 (see Appendix 7).

$$\begin{aligned} RF_{exp}(9,2,6) &= 9 + (9-1)\sum_{l=1}^{6-1} 2^l \\ &= 9 + (8)(2+4+8+16+32) \\ &= 505 \end{aligned} \tag{5}$$

Figure 4 shows again the RFs, this time with actual parametrization of one dilation cycle of the "legacy" version in blue, the current default version in orange and the proposed "cut-off pyramid" version in green. We can observe the path counts varying strongly between parameterizations and a visual comparison of the RF sizes we just computed.

Relating the computed RF to the training sequence length of 512 tokens we can state that in the "legacy" architecture and parametrization one dilation cycle could not quite aggregate information from every input into every output. For that to be the case we need $RF \geq 2L$ where L denotes the input sequence length. The current architecture and parametrization is close to this condition and the first dilation cycle with differing initial kernel size exactly meets it with $RF = 1025 \geq 1024 = 2L$. From that perspective the current default parametrization can be argued to be more fit to the training procedure as each dilation cycle can "attend to" the full training sequence.

2.2.3 Concluding the Arc

The architecture based on a CNN brings the big advantage of linear scaling i.e. $\mathcal{O}(L)$ in sequence length compared the $\mathcal{O}(L^2)$ for attention and the local

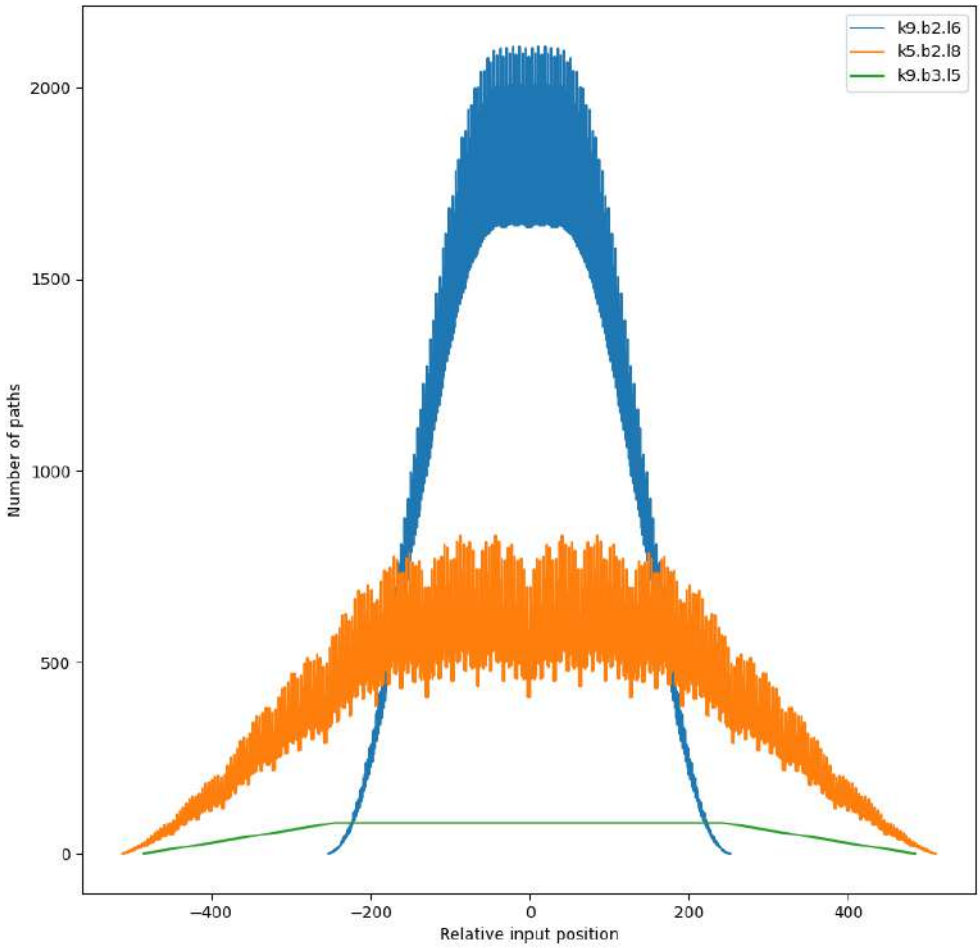


Figure 4: Path distributions for actual model configurations of kernel sizes k, dilation bases b and number of layers l listed as kx.by.lz where x, y, z indicate the parameter value.

accumulation of information in convolutions is hypothesized to be advantageous in the comprehension of DNA data (Benegas et al., 2023). The potential to create a huge receptive field is not fully leveraged as a longer dilation cycle could easily accumulate thousands of tokens.

Furthermore, attention brings in the capacity to do input-dependent computation which ordinary convolutions do not possess. SSMs are one way to bridge the gap to formulate input-dependent convolutions as outlined by Ku et al. (2025) for example.

Nonetheless, the GPN architecture has proven itself already in the "legacy" version to be capable of being able to learn from rather little genome data and successfully predict biologically meaningful results as outlined in the following section.

2.3 Application of GPN

The primary motivation of Benegas et al. (2023) is variant effect prediction (VEP). VEP aims to predict if a single nucleotide polymorphism (SNP) will result in fitness decrease of a carrying individual, i.e. the deleteriousness of a single mutation is to be anticipated.

2.4 GPN-Score

Towards this end they developed a metric called GPN-Score computed as shown in Equation 6. REF denotes the reference nucleotide at a position of interest which one would expect to find, ALT the alternative nucleotide that was actually observed. $P(.)$ is the probability assigned to a given nucleotide by the GPN model at that position given surrounding sequence context.

$$GPNScore(REF, ALT) = log\frac{P(ALT)}{P(REF)} \tag{6}$$

This metric draws from the proxy that model certainty correlates with high conservation in the genome, which in turn is weakly correlated with functional importance (Z. Wang & Zhang, 2009). It is shown that values in the extreme tail of GPN scores over a genome correlate well with known variant effect.

By the approximation of conservation via model confidence, regions in the genome predicted with high confidence and vice versa low entropy can serve as a signal for motif discovery, i.e. short conserved sequences, also shown in Benegas et al. (2023).

2.4.1 Nucleotide Dependency Maps

NDMs (da Silva et al., 2024) are an interesting new way to probe and extract the patterns learned by a DNA-LM. While the GPN was not originally used with this procedure, it is easily adaptable to any DNA-LM that outputs a per nucleotide probability distribution.

The goal is to determine dependencies between nucleotides of an input sequence by purposefully introducing SNPs at sequence position A and observe the change in model predictions at sequence position B. The results can be visualized very nicely via heatmaps, which can reveal various important DNA elements from motifs up to secondary and tertiary RNA structure.

Motivated by this type of analysis we will gauge the extend to which NDMs computed based on the GPN can be informative in the following section.

3 Model Analysis

After the initial introduction of the GPN model and some architectural remarks we now turn to its inference time behavior.

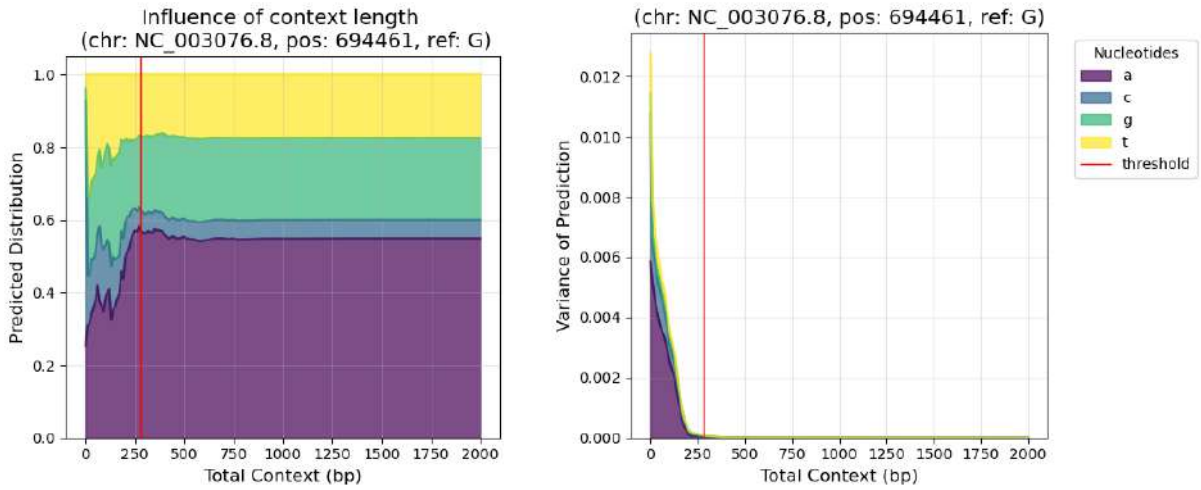
3.1 Context Utilization

As we are per se dealing with a model based on convolutions we are not structurally limited in the amount of data we can present to the model at once. The exponential dilation schedule also constructs a wide receptive field as demonstrated in Section 2.2.2. Given this potential to utilize large context windows we set out to explore how much of this potential is realized. This question also connects to the interest to compute Nucleotide Dependency Maps (NDMs) using the GPN. To properly interpret NDMs it is essential to understand up to which distance the model is capable of capturing interactions.

3.1.1 Nucleotide Level

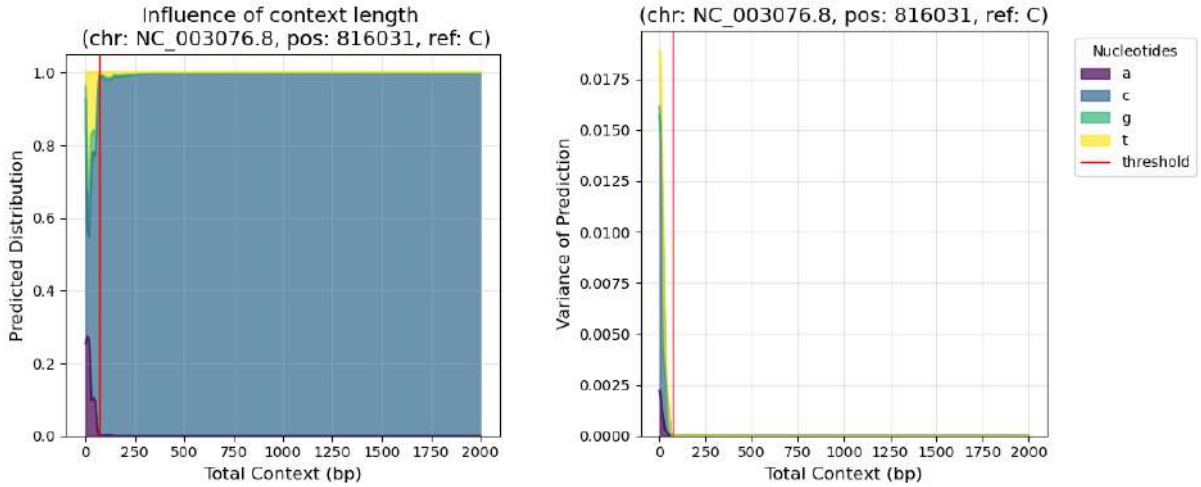
To approach this question the following experimental setup was derived: Given an arbitrary position on chromosome 5, which was explicitly excluded from the training set, of *Arabidopsis thaliana* we construct several DNA sequences with the chosen position in central place. Each sequence is progressively symmetrically larger by linear steps of 10, i.e. each sequence reveals more and more context for the chosen position both upstream and downstream of the position up to a maximum length of 2000bp. In every such sequence after tokenization the central chosen position is replaced by the [MASK] token. By running inference on all sequences and extracting the output at the chosen masked position we can then plot the change in the predicted probability distribution over increasingly larger context sizes. Three exemplary such plots can be found in Figures 5a, 6a, 7a.

To determine how much context is utilized we want to detect when the predicted probability distributions begin to stabilize, i.e. the predictions of the GPN stop to change despite continuously presenting more information. The derived measure of choice for us is to calculate the cumulative variance over the predictions for each possible nucleotide over the increasing context size using a forward looking sliding window of size 100. Once the cumulative variance falls below a threshold value of 1e-4 we call the predictions stable. The sequence size at which the threshold is crossed is our effectively utilized context size. Three exemplary plots depicting this change in cumulative variance can be found in Figures 5b, 6b, 7b.



(a) Change in nucleotide probabilities (b) Change in prediction variance

Figure 5: Model behavior w.r.t position 694461 on Chromosome 5 of *A. thaliana*



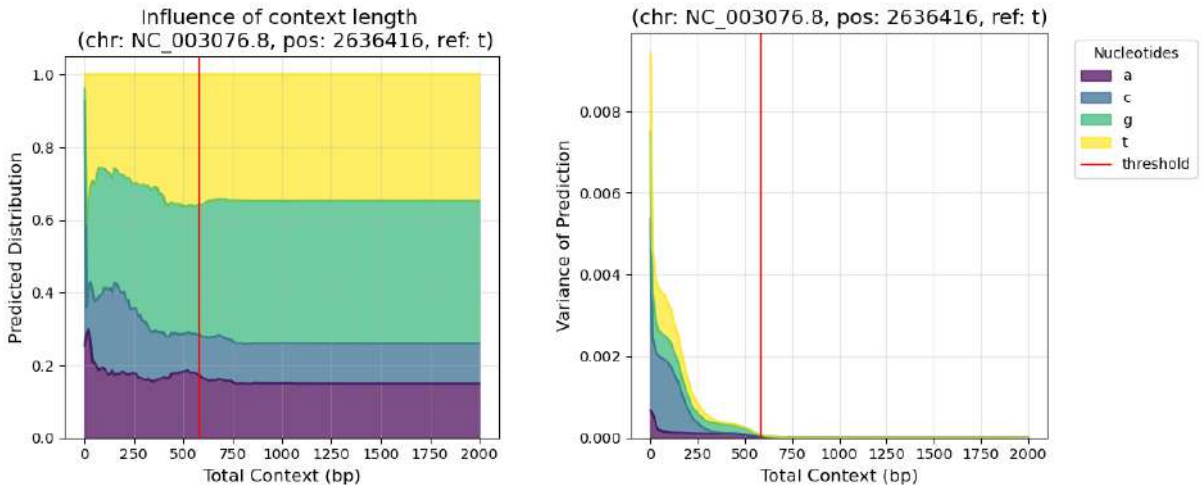
(a) Change in nucleotide probabilities (b) Change in prediction variance

Figure 6: Model behavior w.r.t position 816031 on Chromosome 5 of *A. thaliana*

The exemplary Figures 5, 6 and 7 already demonstrate diverse behaviors of the GPN predictions with regard to the genomic context of the examined positions. Importantly, they illustrate that the measure of choice meaningfully captures what a human might intuit about the stabilization of the predictions by some margin.

3.1.2 Chromosome Level

Using the approach introduced above in Section 3.1.1 we can determine the utilized context size for a single nucleotide position. In order to generalize the behavior of the GPN model to the chromosome and genome level we applied the



(a) Change in nucleotide probabilities (b) Change in prediction variance

Figure 7: Model behavior w.r.t position 2636416 on Chromosome 5 of *A. thaliana*

described analysis to 10.000 positions sampled uniformly at random on the same chromosome 5. Given the results for those positions we can examine the distribution of utilized context sizes. The different genomic settings in which we might sample can lead to varying levels of structure and biologic relevance of positions. These differences could potentially be influential in how much context GPN needs to utilize in order to settle on a prediction. Due to this circumstance we clustered the sampled positions according to their most precise genomic annotation to discover such potential differences as depicted in Figure 8.

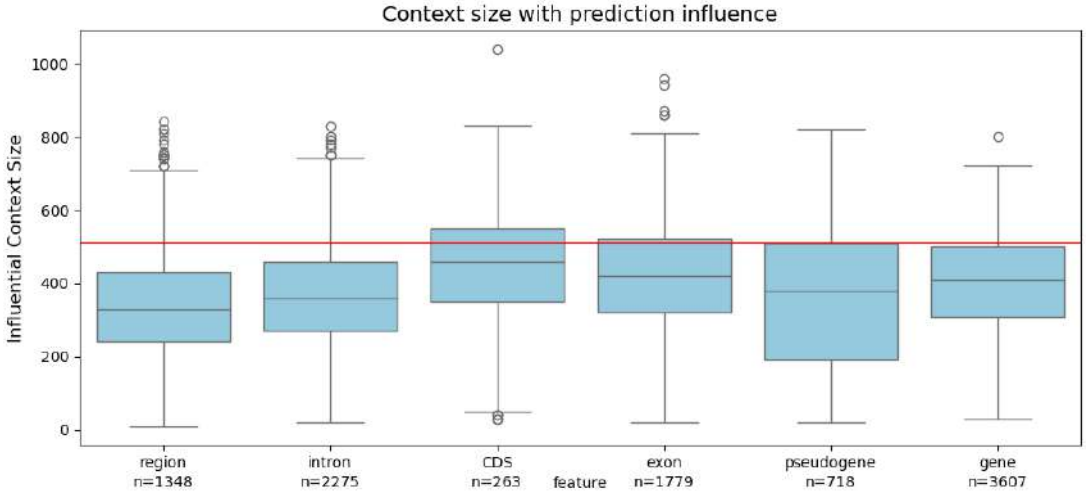


Figure 8: Distribution of influential context length by genomic annotation, n = number of sampled positions of respective region. Only regions with at least 100 sampled positions are included. The horizontal red line marks the 512bp training sequence length.

We can see only moderate differences in influential context size between genomic regions. Notably, despite having set a low threshold, i.e. being sensitive to detect small changes in predicted distribution, most predictions stabilize before utilizing the full context of the training sample size (512bp, visible as red horizontal line).

It can nonetheless not be disregarded that the upper quartile roughly coincides with the 512bp mark, i.e. about 25% of predictions do stabilize only after being presented more context than the training sequence length. The data at hand does not allow a conclusive statement what kind of effect we are observing. It could be that the GPN only learned to generalize and incorporate additional sequence information for a few inference scenarios and would learn to do so for more positions if trained on longer sequences. Alternatively, the effect could be grounded in biology in that many positions are well to be determined with little context and only some benefit from looking at a larger context. Correlating the outlier positions with known biological function could also yield additional insight.

Despite these hypothesis it also has to be stated clearly that the results depend heavily on the choices of step size to increase the context, the length of the window and the chosen threshold value. The threshold value itself directly influences the position at which we cut off, the interplay between step and window size is more delicate. A short window might miss that the distribution has not yet stabilized long term while a long window might blur changes in distribution right before stabilization. It also has to be chosen fit for the step size which governs the resolution of the distribution change.

With respect to NDMs we can state that it is unlikely to capture dependencies of nucleotides exceeding a distance of roughly 1kb, the biggest outlier in Figure 8 when computing probabilities based on the GPN model.

3.2 Applicability to different species

In order to asses the applicability of GPN to different species we have reproduced the UMAP-based (McInnes & Healy, 2018) qualitative analysis presented by the GPN authors. We analysed two different species, *Arabidopsis halleri* and *Tribolium castaneum*, the former closely related, the latter very distantly. Using their procedure, the genome of an individual was chunked into windows of 100bp without overlap which were symmetrically extended to full training sample size context of 512bp for prediction. The averaged final embedding vectors of these windows were down-projected by UMAP and labeled via respective genome annotations. Windows intersecting with more than one genomic region were ignored.

3.2.1 *Arabidopsis halleri*

Arabidopsis halleri as a close relative to *Arabidopsis thaliana* can be expected to be "understood" by the GPN rather well, i.e. the hypothesis is to find distinct clusters of different genomic regions in the resulting UMAP visualization. It was not part of the training data set. Figure 9 displays this with the limited labels that have been available for the used Lan3.1 (DOE-JGI, 2022) genome. As anticipated the coding sequence (CDS) and introns are well distinguished clusters even though some CDS labeled windows seem to be interspersed in different places. This could be rooted in the distribution shift in the genomes of the two species or it could likewise indicate a mislabeled sequence stretch. Developing the capability to determine which is the case could be of practical value.

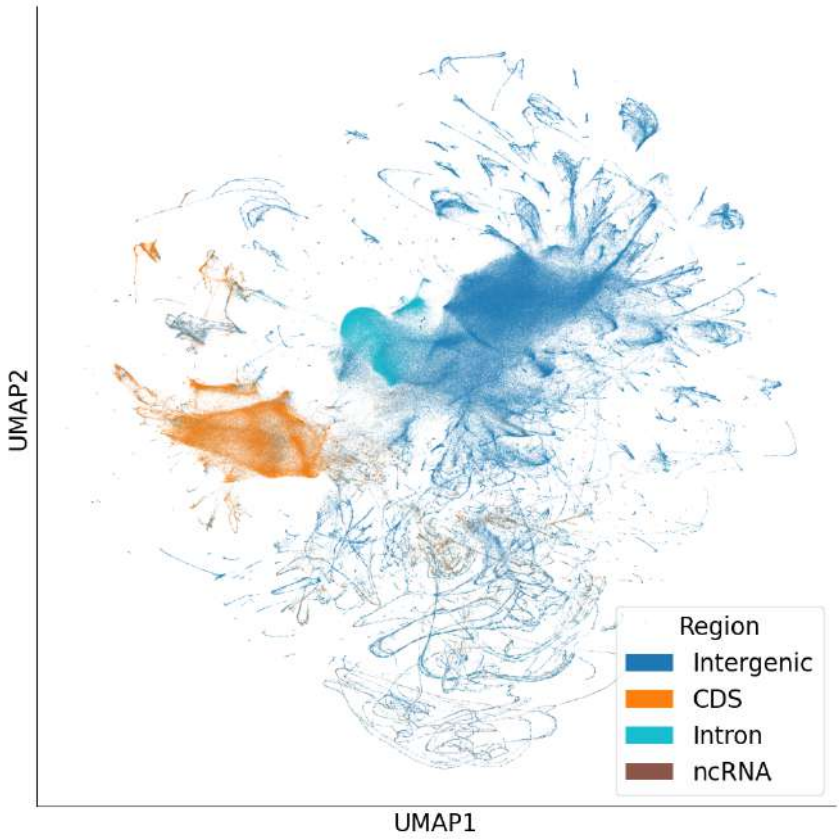


Figure 9: UMAP of GPN embeddings for *Arabidopsis halleri*

3.2.2 *Tribolium castaneum*

Tribolium castaneum as a beetle model organism (Herndon et al., 2019) is of interest to us in order to study the adaptation of three newly sequenced beetle species to deserts. The GPN could provide a basis to identify genomic regions of interest for so far unlabeled genomes. The GPN-Score itself or the adaptation of NDMs to GPN could both provide valuable hints. Despite the training on

plant data it is worth to check how well the GPN handles beetle data. Its UMAP plot on the icTriCast1.1 genome assembly from Childers et al. (2021) can be seen in Figure 10. Similarly to Figure 9 coding sequence (CDS) regions and additionally labeled repeat regions are distinctly recognizable. Finer details are not distinguishable. Finding a CDS cluster can be expected as the coding of amino acids via codon triplets of nucleotides is universal among biological live on earth even though minor differences in interpretation of the codons exist (Elzanowski & Ostell, 2024).

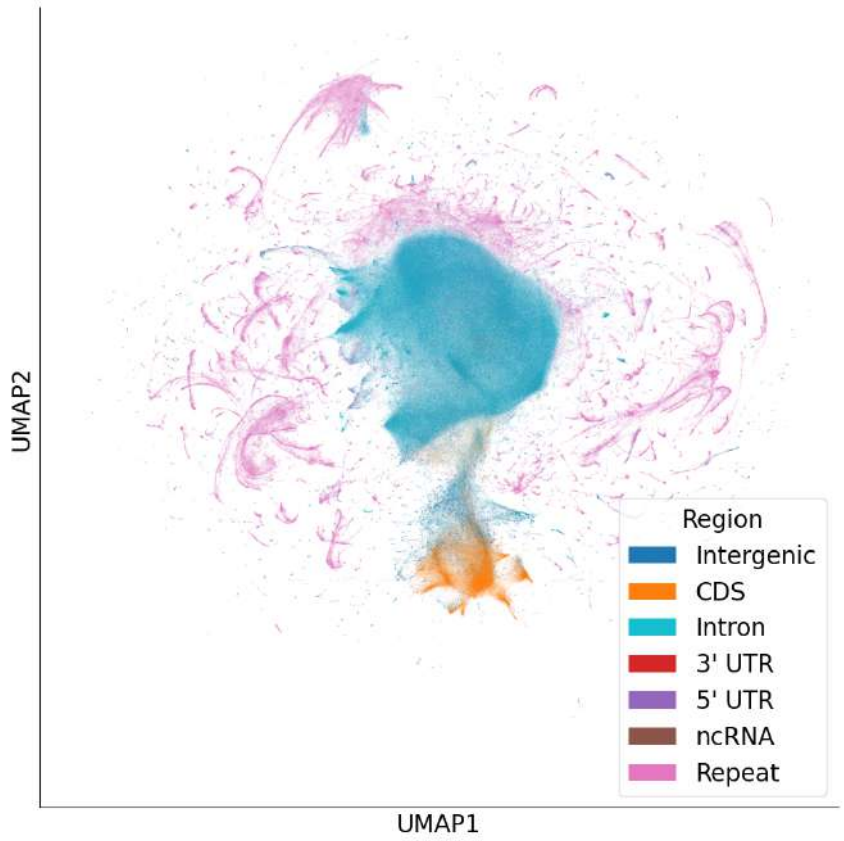


Figure 10: UMAP of GPN embeddings for *Tribolium castaneum*

4 Beetle GPN

Motivated by the qualitative differences in results of Section 3.2 the following section will deal with the training of a new model based on the GPN architecture fit for dealing with beetle data. The existing GPN model trained on *Brassicales* and the new model trained on *Cucujiformia* will be referred to as GPN-B and GPN-C, respectively.

4.1 Data Preparation

In order to acquire a data set centered around species like *Tribolium castaneum* we traversed the taxonomic ranks starting from it until more than ten annotated NCBI reference sequences where available (see Appendix Table 3 for included species). All belong to the infraorder *Cucujiformia* representing most plant-eating beetles (Robertson et al., 2015). NCBI reference sequences have soft-masked repeat regions recognizable as lower case letters in the sequence (NIH, 2025). This is important as such repetitive regions can compromise model performance.

The data preparation procedure follows Benegas et al. (2023) exactly. Of the chosen species all gene and promoter regions were selected into the data set and an equal amount of randomly sampled sequences. The resulting data set (Büdenbender, 2025) contains on the order of 4.5M sequences of length 512bp equating to about 2.3B nucleotide tokens.

4.2 Model Architecture

We train two models, one with default parameters of the current GPN architecture (see Section 2.1 and Figure 1) and one with parametrization for the "cut-off pyramid" with the kernel size and dilation schedule according to our observations from Section 2.2.2.

4.2.1 GPN-C1

This model adheres to the default current architecture of the GPN with first layer kernel size 9, then kernel size 5 over a total of 25 layers and a Transformer-style reverse bottleneck. It employs a dilation base of 2 and dilation cycle of 8. The overall configuration results in about 93M parameters.

4.2.2 GPN-C2

For this model we apply our observations from Figure 3 and choose kernel size 9 everywhere with dilation base 3 and dilation cycle 5 resulting in a receptive field of

969, falling little short of $2L = 1024$. To preserve comparability between the two models, we reduce the layers to 20 to accommodate for the additional parameters from larger kernels. This configuration results in almost 95M parameters equal to a 1.7% increase in parameter count.

4.3 Model Training

Both models are trained using an identical schedule using the Adam optimizer (Kingma & Ba, 2014) with default parameters and batch size 256. They are trained for 240k steps with constant learning rate of 1e-3 and an additional 30k steps with cosine learning rate akin to the original GPN training procedure. The initial 240k steps are double that of the GPN-B model trained by Benegas et al. (2023). Mirroring them, the loss and thereby the gradients resulting from repeat regions during training are down-weighted with a factor of 0.1. Figure 11 and 12 show the respective training and evaluation loss curves of the two training runs. The training took place on the RAMSES cluster of the University of Cologne on four H100 NVIDIA GPUs.

4.3.1 Further Explored Ideas

We explored training a network with the custom wide parametrization but larger RF (2913 per dilation cycle) by increasing the dilation cycle to 6 and reducing the layers to 18 to encompass three full cycles seen in Figure 11 and 12 as GPN-CL. It was trained on the same species reference genomes but cut into sequences of 2048bp with 1024bp overlap and and cosine learning rate schedule with learning rate 2e-3.

We observed a faster decreasing and lower loss during training on its own dataset, but on the 512bp dataset we do not (eval loss 1.08 / test loss 1.00) and comparability suffers. As the models with three full dilation cycles turns out to be smaller with about 85M total parameters and training sequences are 2048bp long, we can effectively show twice the amount of tokens per batch with batch size 128 on our available GPUs. Thereby the model effectively trains at double the speed and with more stable gradients in addition to the different learning rate schedule.

A weak signal exists that longer sequences allow for better performance in MLM tasks on DNA. So while we might improve the results on longer sequences, the other models could possibly have achieved lower losses on there dataset as well if trained for an equal amount of tokens or given the more stable gradients via gradient accumulation. Furthermore, the evaluation data set for the 2048bp sequences is likely to be compromised either by its small size or unlucky sampling,

but the observed results are extremely unlikely to be representative. Also the training run is incomplete as allocated resources were time limited.

The model briefly introduced here is excluded from the discussion of results in Section 4.4 due to the lack of comparability, incomplete training and performance increase.

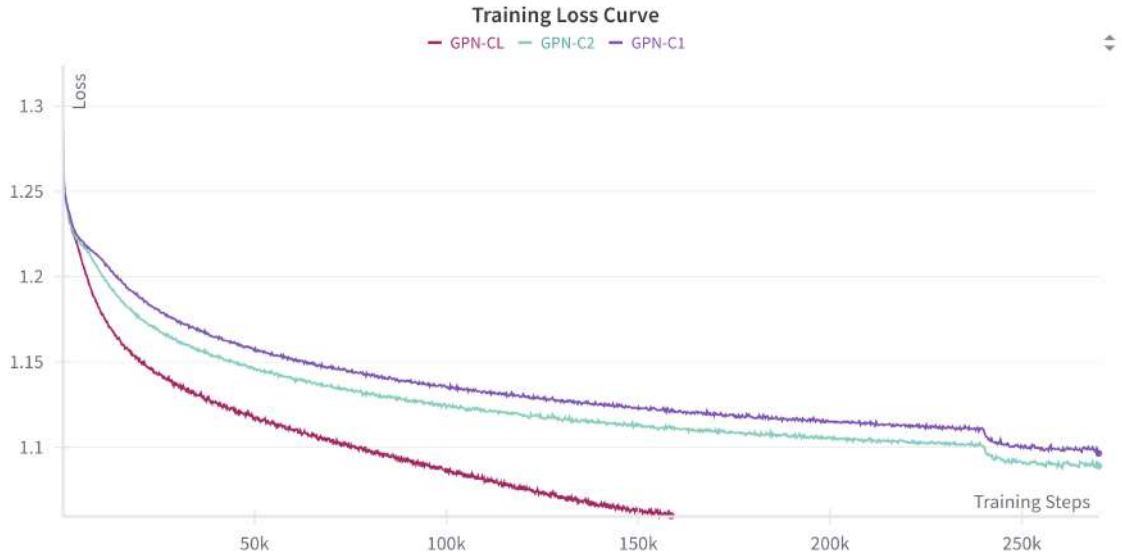


Figure 11: Training Loss

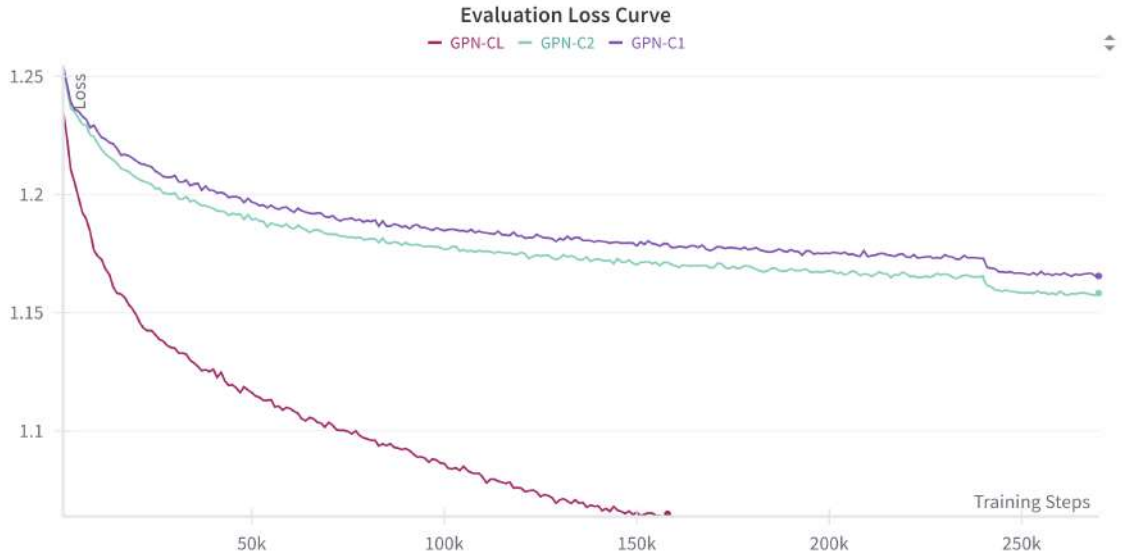


Figure 12: Evaluation Loss

4.4 Results

To compare the new models to the GPN-B we decided to compare top-1-accuracy as a quantitative metric and the resulting UMAP plots of averaged embeddings as a qualitative metric.

4.4.1 Accuracy

Table 1 shows the accuracy of the models. Most notably, we observe an improvement of accuracy between 8-13% when comparing the GPN-B to GPN-C1 and GPN-C2. The difference in confidence intervals between eval and test accuracy is due to an unintended data set size ratio of 4:1. While GPN-B is still much better compared to baseline of random choice (25%) indicating some cross-species generalization, GPN-C1/2 show the aforementioned improvement. The 99% confidence intervals of the two GPN-C models overlap, so between the two parametrizations no measurable performance effect can be concluded. Nonetheless, we can observe a differentiation in the training and evaluation loss curves after 7k steps as depicted in Figures 11 and 12 demonstrating a small but measurable difference in training behavior. Table 2 also reflects the small loss difference at least in the evaluation data set.

It has to be acknowledged that both runs were performed just once and with an identical seed. To further investigate the re-parametrization at a minimum the same training should be run with different seeds to gain confidence of observing a real opposed to a random effect. Furthermore, the increase in accuracy and decrease in loss from eval to test set hints at a possible distribution shift between the two sets. Both were build with forced inclusion of specific chromosomes (Chr. 10, Chr. 11 respectively) from *Tribolium castaneum* to have these available as test fields for downstream analysis. A difference in biological function or complexity between the chromosomes could help explain the shift. Also the unintended size difference which was recognized only later during analysis could play a role as results could be less statistically robust.

Table 1: Accuracy on *Cucujiformia* data set with 99%-confidence interval

Model	**Accuracy (eval)**	**Accuracy (test)**
GPN-B	42.78 ±0.62%	42.95 ±1.25%
GPN-C1	51.48 ±0.63%	56.03 ±1.26%
GPN-C2	**51.89** ±0.63%	**56.25** ±1.26%

Table 2: Loss on *Cucujiformia* data set

Model	Loss (eval)	Loss (test)
GPN-B	1.24	1.24
GPN-C1	1.09	**0.99**
GPN-C2	**1.08**	**0.99**

4.4.2 Structure

We now turn to investigate the structure of the learned embeddings of the two GPN-C variants. The procedure is taken almost exactly from Benegas et al. (2023) but for the choice to use cosine similarity as a distance metric when generating the UMAP projection inspired by Ethayarajh (2019). Also the preparation of the annotation file had to be adjusted to the data at hand. Figure 13 and 14 display results of similar global structure. Identical to Section 3.2 each point in the plots represents an averaged embedding at the final layer over 100bp with an unambiguous annotation of the entire *Tribolium castaneum* reference genome.

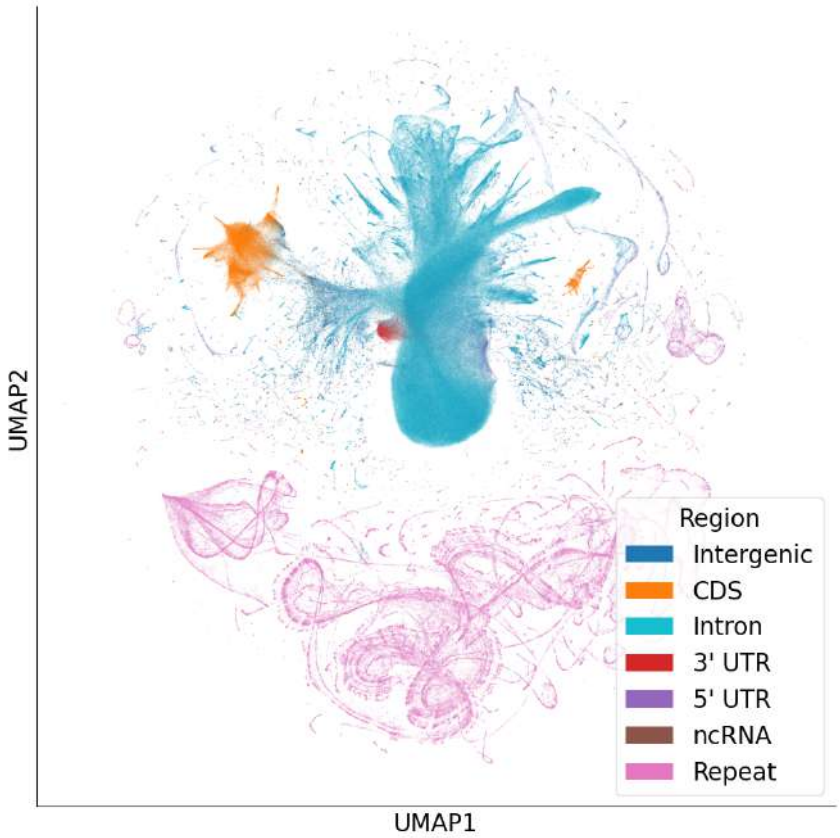


Figure 13: UMAP GPN-C1

Coding sequences, labeled CDS, represent arguably the most structured regions of DNA and are well separated in the embedded space. With regard to

CDS it is notable that a second small cluster exists in Figure 13. Following which DNA sequences it is made up of could be interesting but similarly it could be an arbitrary artefact of the model. Repeats, which are repetitive regions in the DNA seem to be well separable along either UMAP1 or UMAP2. The 3' and 5' UTRs (untranslated regions) are recognizable unlike the GPN-B embeddings of *Tribolium castaneum* in Figure 10 but not as clearly separated as in the results of Benegas et al. (2023) on *Arabidopisis thaliana*.

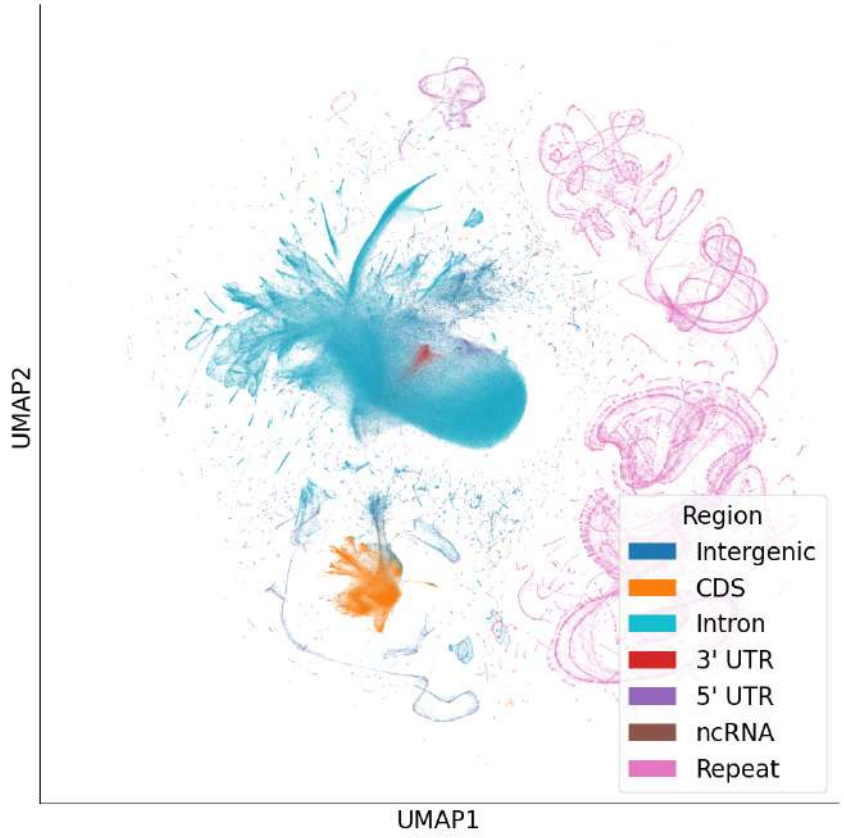


Figure 14: UMAP GPN-C2

4.4.3 Summary

Overall the model embeddings display semantically relevant structure and an improvement with regard to embeddings of *Tribolium castaneum* taken from GPN-B displayed earlier in Figure 10. Drawing from the presented metrics both trained models seem fit for applications with beetle genomic data. It would be of further interest to repeat the context utilization analysis from Section 3 with the two new models to check for differences in inference behavior between the two parameterizations. All published datasets and models can found at `https://huggingface.co/sbuedenb`.

5 Conclusion

In the presented work we have studied the Genomic Pretrained Network in aspects of its theoretical and architectural composition as well as its training and inference time behavior.

Closer examination of its internal workings lead, to the authors best knowledge, to a new perspective on exponentially dilated convolutions. The entire niche of network architectures incorporating these seems to be under-explored in the regime of very long sequence comprehension (thousands to millions of tokens). Several ideas for DNA-LM architectures inspired e.g. by Benegas et al. (2023), Yu and Koltun (2015) and Duta, Georgescu, and Ionescu (2021) are waiting to be explored.

The introduced behavioral analysis of utilized context size can be build upon to extract new information from DNA-LMs and it can be extended to investigate different properties of models in dependence on presented context length like the cross-entropy loss or the accuracy to discover sweet spots of model capabilities.

With regard to information extraction, Nucleotide Dependency Maps have captured the authors interest as they show that innovative ways to probe language models can reveal new insight. Connecting these with measures from information theory like mutual entropy seems interesting.

Further, despite being rather glimpsed over, the smaller model with larger receptive field did seem to learn better predictions when being trained on and presented with longer sequences. This could indicate the training regime of short sequences used in the majority of this work can be improved upon. Coupled with the behavioral analyses, training sequence length and inference capabilities could be jointly illuminated.

The new models trained on *Cucujiformia* open up the application of the GPN-Score and NDMs to plant eating beetle species, which will be of future interest for a project to study beetle adaptations to desert environments.

A last avenue, that should not go unmentioned, is the comparison of bigger, more general DNA-LMs with regard to their training data corpus and parameter count to smaller ones, like trained in this work. Investigating prediction performance deltas on more general and more specific DNA datasets will help to illuminate what scales of DNA pattern can be learned and to which detail under which circumstances.

6 Acknowledgements

I want to thank Prof. Dr. Wiehe and his group for there trust and continued support through this project. Further, I want to thank Prof. Dr. Frahling, who, with his lecture on Deep Learning, has prepared me exceptionally well for this work.

Additionally, I want to thank Dr. Benegas for making time to answer my questions, the Regionale Rechenzentrum Köln (RRZK) who generously supported this project with compute from the RAMSES cluster and the team behind the Snakemake project (Mölder et al., 2021) for bringing much needed tools for reproducible digital scientific workflows.

Lastly, I thank Lorinda and Lisa for proof reading, your unsolicited support is deeply appreciated.

A Appendix

A.1 GPN Legacy Architecture

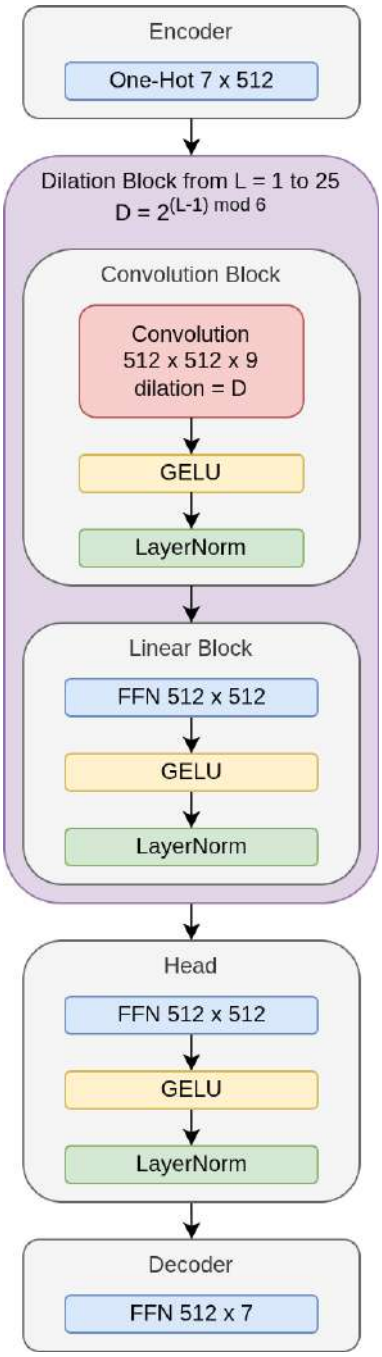


Figure 15: GPN Legacy Architecture

A.2 Data Set

Table 3: *Cucujiformia* Data Set Composition

Assembly Accession	**Assembly Name**	**Organism Name**
GCF_917563875.1	PGI_DIABVI_V3a	*Diabrotica virgifera virgifera*
GCF_024364675.1	icAetTumi1.1	*Aethina tumida*
GCF_031307605.1	icTriCast1.1	*Tribolium castaneum*
GCF_914767665.1	icHarAxyr1.1	*Harmonia axyridis*
GCF_963966145.1	icTenMoli1.1	*Tenebrio molitor*
GCF_907165205.1	icCocSept1.1	*Coccinella septempunctata*
GCF_040115645.1	ASM4011564v1	*Euwallacea fornicatus*
GCF_022605725.1	icAntGran1.3	*Anthonomus grandis grandis*
GCF_026250575.1	icDioCari1.1	*Diorhabda carinulata*
GCF_026230105.1	icDioSubl1.1	*Diorhabda sublineata*
GCF_040954645.1	icDiaUnde3	*Diabrotica undecimpunctata*
GCF_039881205.1	ESF131.1	*Euwallacea similis*

A.3 Computing Receptive Field

From the default parametrization in the "legacy" version we can describe the RF per repeating dilation cycle by Equation 5. Over the the 25 layers this cycle repeats four total times. The total RF of this model then is:

$$RF_{GPN_l} = 505 + 3 * (505 - 1) + (9 - 1) = 2025 \tag{7}$$

For the current architecture with default parametrization with a dilation cycle of 8, repeating 3 times fully over 25 layers, taking into account that the first layer uses kernel size 9, using Equation 4 we can infer the following RF:

$$RF_{GPN_c} = (9 - 5) + 1021 + 2 * (1021 - 1) + (5 - 1) = 3069 \tag{8}$$

References

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, *abs/1409.0473*. Retrieved from https://api.semanticscholar.org/CorpusID:11212020

Benegas, G., Batra, S. S., & Song, Y. S. (2023). Dna language models are powerful predictors of genome-wide variant effects. *Proceedings of the National Academy of Sciences of the United States of America*, *120*. Retrieved from https://api.semanticscholar.org/CorpusID:251911014

Brixi, G., Durrant, M. G., Ku, J., Poli, M., Brockman, G., Chang, D., ... Hie, B. L. (2025). Genome modeling and design across all domains of life with evo 2. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:276569131

Büdenbender, S. T. (2025). *Cucujiformia dataset.* https://huggingface.co/datasets/sbuedenb/big_beetle_dataset. (Accessed: 2025-06-12)

Childers, A. K., Geib, S. M., Sim, S. B., Poelchau, M. F., Coates, B. S., Simmonds, T. J., ... Scheffler, B. (2021). The usda-ars ag100pest initiative: High-quality genome assemblies for agricultural pest arthropod research. *Insects*, *12*. Retrieved from https://api.semanticscholar.org/CorpusID:235896348

Dalla-torre, H., Gonzalez, L., Mendoza-Revilla, J., Carranza, N. L., Grzywaczewski, A. H., Oteri, F., ... Pierrot, T. (2024). Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, *22*, 287 - 297. Retrieved from https://api.semanticscholar.org/CorpusID:274369582

da Silva, P. T., Karollus, A., Hingerl, J. C., Galindez, G., Wagner, N., Hernández-Alias, X., ... Gagneur, J. (2024). Nucleotide dependency analysis of dna language models reveals genomic functional elements. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:271544513

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J.-M., Zhang, R., ... Zhang, Z. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *ArXiv*, *abs/2501.12948*. Retrieved from https://api.semanticscholar.org/CorpusID:275789950

DOE-JGI. (2022). *Arabidopsis halleri v2.03.* http://phytozome.jgi.doe.gov/. (Accessed: 2025-06-20)

Dong, S., & Searls, D. B. (1994). Gene structure prediction by linguistic methods. *Genomics*, *23 3*, 540-51. Retrieved from https://api.semanticscholar.org/CorpusID:16742430

Duta, I. C., Georgescu, M.-I., & Ionescu, R. T. (2021). Contextual convolutional neural networks. *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 403-412. Retrieved from https://api.semanticscholar.org/CorpusID:237142548

Elzanowski, A., & Ostell, J. (2024). *The genetic codes.* https://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprintgc.cgi. (Accessed: 2025-06-19)

Ethayarajh, K. (2019). How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Conference on empirical methods in natural language processing.* Retrieved from https://api.semanticscholar.org/CorpusID:202120592

Gupta, A., & Rush, A. M. (2017). Dilated convolutions for modeling long-distance genomic dependencies. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:28321443

Herndon, N., Shelton, J. M. G., Gerischer, L., Ioannidis, P., Ninova, M., Dönitz, J., ... Bucher, G. (2019). Enhanced genome assembly and a new official gene set for tribolium castaneum. *BMC Genomics*, *21*. Retrieved from https://api.semanticscholar.org/CorpusID:210195064

Ji, Y., Zhou, Z., Liu, H., & Davuluri, R. V. (2020). Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:221823863

Kalchbrenner, N., Espeholt, L., Simonyan, K., van den Oord, A., Graves, A., & Kavukcuoglu, K. (2016). Neural machine translation in linear time. *ArXiv*, *abs/1610.10099*. Retrieved from https://api.semanticscholar.org/CorpusID:13895969

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980*. Retrieved from https://api.semanticscholar.org/CorpusID:6628106

Ku, J., Nguyen, E., Romero, D. W., Brixi, G., Yang, B., Vorontsov, A., ... Poli, M. (2025). Systems and algorithms for convolutional multi-hybrid language models at scale. *ArXiv*, *abs/2503.01868*. Retrieved from https://api.semanticscholar.org/CorpusID:276774804

Linder, J., Srivastava, D., Yuan, H., Agarwal, V., & Kelley, D. R. (2023). Predicting rna-seq coverage from dna sequence as a unifying model of gene regulation. *Nature Genetics*, *57*, 949 - 961. Retrieved from https://api.semanticscholar.org/CorpusID:261528750

Luo, W., Li, Y., Urtasun, R., & Zemel, R. S. (2016). Understanding the effective receptive field in deep convolutional neural networks. *ArXiv*, *abs/1701.04128*. Retrieved from https://api.semanticscholar.org/CorpusID:5665033

McInnes, L., & Healy, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *ArXiv*, *abs/1802.03426*. Retrieved from https://api.semanticscholar.org/CorpusID:3641284

Mölder, F., Jablonski, K. P., Letcher, B., Hall, M. B., Tomkins-Tinch, C. H., Sochat, V. V., ... Köster, J. (2021). Sustainable data analysis with snakemake. *F1000Research*, *10*. Retrieved from https://api.semanticscholar.org/CorpusID:234357363

Nguyen, E. D., Poli, M., Faizi, M., Thomas, A. W., Birch-Sykes, C., Wornow, M., ... Ré, C. (2023). Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *ArXiv*. Retrieved from https://api.semanticscholar.org/CorpusID:259274952

NHGRI. (2022). *The cost of sequencing a human genome.* https://www.genome.gov/about-genomics/fact-sheets/Sequencing-Human-Genome-cost. (Accessed: 2025-05-21)

NIH. (2025). *Ncbi genomes ftp.* https://www.ncbi.nlm.nih.gov/datasets/docs/v2/policies-annotation/genomeftp/#are-repetitive-sequences-in-eukaryotic-genomes-masked. (Accessed: 2025-05-24)

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., ... Lowe, R. J. (2022). Training language models to follow instructions with human feedback. *ArXiv*, *abs/2203.02155*. Retrieved from https://api.semanticscholar.org/CorpusID:246426909

Robertson, J. A., Ślipiński, A., Moulton, M. J., Shockley, F. W., Giorgi, A. J., Lord, N. P., ... McHugh, J. V. (2015). Phylogeny and classification of cucujoidea and the recognition of a new superfamily coccinelloidea

(coleoptera: Cucujiformia). *Systematic Entomology*, *40*. Retrieved from https://api.semanticscholar.org/CorpusID:55206626

Sanabria, M., Hirsch, J., Joubert, P. M., & Poetsch, A. R. (2024). Dna language model grover learns sequence context in the human genome. *Nat. Mac. Intell.*, *6*, 911-923. Retrieved from https://api.semanticscholar.org/CorpusID:271411089

Sanabria, M., Hirsch, J., & Poetsch, A. R. (2023). Distinguishing word identity and sequence context in dna language models. *BMC Bioinformatics*, *25*. Retrieved from https://api.semanticscholar.org/CorpusID:259925286

Schiff, Y., Kao, C.-H., Gokaslan, A., Dao, T., Gu, A., & Kuleshov, V. (2024). Caduceus: Bi-directional equivariant long-range dna sequence modeling. *ArXiv*, *abs/2403.03234*. Retrieved from https://api.semanticscholar.org/CorpusID:268253280

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. In *Speech synthesis workshop.* Retrieved from https://api.semanticscholar.org/CorpusID:6254678

Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Neural information processing systems.* Retrieved from https://api.semanticscholar.org/CorpusID:13756489

Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., & Cottrell, G. (2017). Understanding convolution for semantic segmentation. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1451-1460. Retrieved from https://api.semanticscholar.org/CorpusID:4599765

Wang, Z., & Zhang, J. (2009). Why is the correlation between gene importance and gene evolutionary rate so weak? *PLoS Genetics*, *5*. Retrieved from https://api.semanticscholar.org/CorpusID:6413822

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, *abs/1609.08144*. Retrieved from https://api.semanticscholar.org/CorpusID:3603249

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., ... Liu, T.-Y. (2020). On layer normalization in the transformer architecture.

ArXiv, *abs/2002.04745*. Retrieved from `https://api.semanticscholar.org/CorpusID:211082816`

Yang, K. K., Lu, A. X., & Fusi, N. (2024). Convolutions are competitive with transformers for protein sequence pretraining. *bioRxiv*. Retrieved from `https://api.semanticscholar.org/CorpusID:248990392`

Yu, F., & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *CoRR*, *abs/1511.07122*. Retrieved from `https://api.semanticscholar.org/CorpusID:17127188`

Zhou, Z., Ji, Y., Li, W., Dutta, P., Davuluri, R. V., & Liu, H. (2023). Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *ArXiv*, *abs/2306.15006*. Retrieved from `https://api.semanticscholar.org/CorpusID:259262243`