SELF-GENOMENET: SELF-SUPERVISED LEARNING WITH REVERSE-COMPLEMENT CONTEXT PREDICTION FOR NUCLEOTIDE-LEVEL GENOMICS DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce Self-GenomeNet, a novel contrastive self-supervised learning method for nucleotide-level genomic data, which substantially improves the quality of the learned representations and performance compared to the state-of-the-art deep learning frameworks. To the best of our knowledge, Self-GenomeNet is the first self-supervised framework that learns a representation of nucleotide-level genome data, using domain-specific characteristics. Our proposed method learns and parametrizes the latent space by leveraging the reverse-complement of genomic sequences. During the training procedure, we force our framework to capture semantic representations with a novel context network on top of features extracted by a deep encoder network. The network is trained with an unsupervised contrastive loss. Extensive experiments show that our method with self-supervised and semi-supervised settings is able to considerably outperform previous deep learning methods on different datasets and a public bioinformatics benchmark. Moreover, the learned representations generalize well when transferred to new datasets and tasks. The source code of the method and all the experiments are available in the supplementary material.

1 INTRODUCTION

In bioinformatics, learning from unlabeled data can reduce deployment costs in many deep learning applications, since it reduces the number of required annotated samples that often require manual experimental validation, such as the functional annotation of genes (Gligorijević et al., 2021) or chromatin effects of single nucleotide polymorphisms (SNPs) (Zhou and Troyanskaya, 2015). Self-supervised learning is amongst the most promising approaches for learning from limited amounts of labeled data and is an attractive approach in bioinformatics due to the availability of unlabeled sequence data. In contrast to supervised methods, these techniques learn a representation of the data without relying on human annotation. Early self-supervised methods focused on solving a *pretext task* (Devlin et al., 2018) that allows utilizing the data itself to generate labels and use supervised methods to solve unsupervised problems. The representations learned by performing this task can be used as a starting point for *downstream* supervised tasks such as taxonomic prediction or gene annotation. Several different self-supervised representation learning methods were proposed e.g. in the fields of natural language processing (NLP) and computer vision (CV). However, self-supervised learning has not yet seen such widespread adoption in bioinformatics and remains an important and challenging endeavor for the representation of genome data.

Existing methods for representation learning for omics-data tends to use these methods from NLP or CV with slight architecture adjustments and additional preprocessing (e.g. Rives et al., 2021; Lu et al., 2020; Ciortan and Defrance, 2021). However, these methods so far do not take many biological characteristics into account. The genome sequence embodied by one strand on the DNA molecule is always accompanied by another strand going in the opposite direction with complementary nucleotides. This means that both the forward and the *reverse-complement* (RC) sequences are valid DNA sequences and equally correspond to several properties of the genome, such as regulatory and taxonomical characteristics. Therefore, machine learning models developed to distinguish different aspects of genomic sequences are ideally required to be equivariant with respect to the reverse-complement Zhou et al. (2021). Our proposed Self-GenomeNet uses a context network that learns representations that take the RC into account.

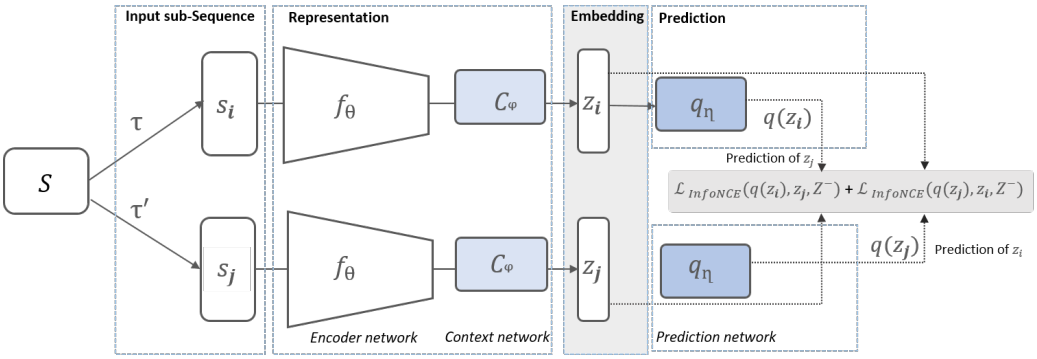

Figure 1: Conceptual illustration of our proposed Self-GenomeNet. Our method initially creates two subsequences s_i, s_j from a sequence S. Then, the subsequences are fed into a representation block composed of the encoder network f_θ, followed by the context network C_ϕ to produce the embeddings z_i, z_j. Finally, each of the two embeddings predicts the other one, against embeddings of other sequences in the same batch, using an unsupervised contrastive loss.

Specifically, we propose a simple contrastive self-supervised algorithm for learning representations of nucleotide-level genomic sequences (Figure 1). Conceptually, our method divides a sequence into two complementary subsequences, with one of them transformed to its RC. Then, the contextual representations of these subsequences are required to distinguish each other from the representations of other, random sequences. We exploit the inherent symmetry in the genome sequence implied by the RC to achieve a natural simplification over possible methods that do not take it into account, all while encoding the RC awareness in our architecture. Because genome sequences can have semantic structure at varying ranges of length, representations that respect both local and global structures are preferable. Therefore, we train our network with varying-length subsequences both for the input and the target subsequence. For this, the context network makes use of a recurrent neural network layer, which makes the computationally efficient training based on all used length scales within a single step possible. Our self-supervised method utilizes contrastive loss to maximize the similarity between representations while minimizing the similarity between representations derived from unrelated sequences.

Contributions: We present the first self-supervised framework tailored specifically to DNA sequence data. Our main findings and contributions can be summarized as follows: (1) We introduce a contrastive self-supervision method that is specialized for nucleotide-level genomics data. (2) We propose a context-aware network that makes use of inherent symmetries in genomics data in the form of the reverse complement and improves the quality of learning representation. (3) We evaluate our methods for self-supervised and semi-supervised settings on bioinformatic benchmarks and datasets, achieving superior performance in several tasks. (4) We show the learned representations generalize well when transferred to a new dataset and task.

2 Related Work and Background

Self-supervised learning Models based on deep self-supervised learning have become a standard method of choice in natural language processing (NLP). Powerful models such as GPT-3 (Brown et al., 2020) and BERT (Devlin et al., 2018) are pre-training in a self-supervised manner on a large text corpus. They are then often fine-tuned for various tasks using smaller task-specific datasets, where they show performance that is superior to purely supervised methods. Recently, self-supervised visual representation learning (Chen et al., 2020) outperformed supervised methods and achieved state-of-the-art performance on ImageNet (Deng et al., 2009). Self-supervised contrastive learning is one of the most successful methods where it maximizes the agreement of learned representations between differently augmented views of the same example (Chen et al., 2020; Zbontar et al., 2021; Oord et al., 2018). Motivated by this, we propose a novel contrastive self-supervised model similar to (Oord et al., 2018) which our network differs in network architecture, representation learning strategy, and applications in bioinformatics.

Self-supervised learning in bioinformatics Existing self-supervised learning techniques applied on sequential RNA and protein data are adopted from other application fields of deep learning. Rives et al. (2021) adopted BERT (Devlin et al., 2018) as a pre-trained model for predicting masked amino acids and achieved state-of-the-art performance on some sequence predictions tasks. Similarly, Contrastive-sc (Ciortan and Defrance, 2021) is a method towards adaptation from computer vision for single-cell RNA sequencing data to perform cell clustering. CPCProt (Lu et al., 2020) applied CPC (Oord et al., 2018) model and train the network on protein data. They showed that the benefit of a trained embedding with contrastive loss can achieve comparable performance to bigger models on different datasets. Similarly (Lu et al., 2020; Rives et al., 2021), Self-GenomeNet is trained by the contrastive loss in which the positive pairs include the input sequence and a reverse-complement of the sequence. Unlike previous works, we trained our network with varying-length input to explore both local and global structures of sequences.

Deep learning methods for genome sequences In genomics, many deep learning architectures use convolutional layers to capture local patterns in data, whereas recurrent layers learn long-range dependencies in the sequences (Eraslan et al., 2019). For example, DeepTACT (Li et al., 2019) predicts the chromatin contacts between regulatory elements using bidirectional LSTM (Hochreiter and Schmidhuber, 1997) on top of merged convolutional layers. DeepDNA (Wang et al., 2019) uses an LSTM layer on top of a convolutional layer to compress human mitochondrial genome data. Zhou et al. published DeepSEA Zhou and Troyanskaya (2015) dataset and developed a model for this dataset which contains multiple convolutional layers. Later, DanQ (Quang and Xie, 2016) improved the deep learning framework of DeepSEA using a recurrent unit on top of a convolutional layer. In this paper, we also used the DeepSEA dataset and DanQ model. We compared the performance of our method with several other self-supervision techniques.

Reverse Complement DNA molecules in living organisms always occur as a double strand, where both strands have opposite directions and consist of nucleotides that are complementary to each other. The sequence on one strand is therefore accompanied by its *reverse complement* (RC) on the other: The sequence in reverse, and with the *A, C, G, T* nucleotides replaced by *T, G, C, A* respectively. Since both forward and reverse-complement sequences are found in the structure, they are equally responsible for several properties of the genome, such as regulatory and taxonomical characteristics. However, next-generation sequencing techniques (NGS) output sub-sequences randomly drawn from both strands, since it usually involves breaking the genome into a collection of small DNA fragments that are sequenced individually. Therefore, machine learning models developed to work on NGS datasets should identify genomic characteristics to be invariant with respect to the reverse-complement. To ensure this invariance, several techniques have been proposed (Shrikumar et al., 2017; Zhou et al., 2021) and benchmarked (Zhou et al., 2021). In our work, we will approach this technique from another perspective and exploit this genome-specific feature to tighten the lower bound between the mutual information between learned representations and valid varying-size genome sequences.

3 METHOD

We first describe the motivation of our proposed method before explaining its details in Section 3.2.

3.1 MOTIVATION

Most successful self-supervised learning approaches are built for computer vision or NLP and are therefore not optimal for bioinformatics data. For example, the self-supervision task of CPC (Oord et al., 2018) or generative language models (Dai and Le, 2015) is based on predicting the representation of a short, constant length subsequence or a single token, which could be an N-gram or a single nucleotide. We hypothesize single-length short sequence targets hinder the model from learning global semantics effectively. We, therefore, propose training representations that are based on varying length sequences – which typically *is* the case for both CPC and single character models – that are themselves trained by predicting representations of varying length sequences – which is a novelty in this area. Similar to Oord et al. (2018), we use a recurrent context network on top of an encoding network that encodes to constant-size patches. However, we use the output of the context network to predict the output of the same context network applied to other patches, as opposed to predicting only the output of the encoding network. The fact that the representation is expected to be equivariant w.r.t. the RC simplifies this approach: instead of having to fit both one forward and

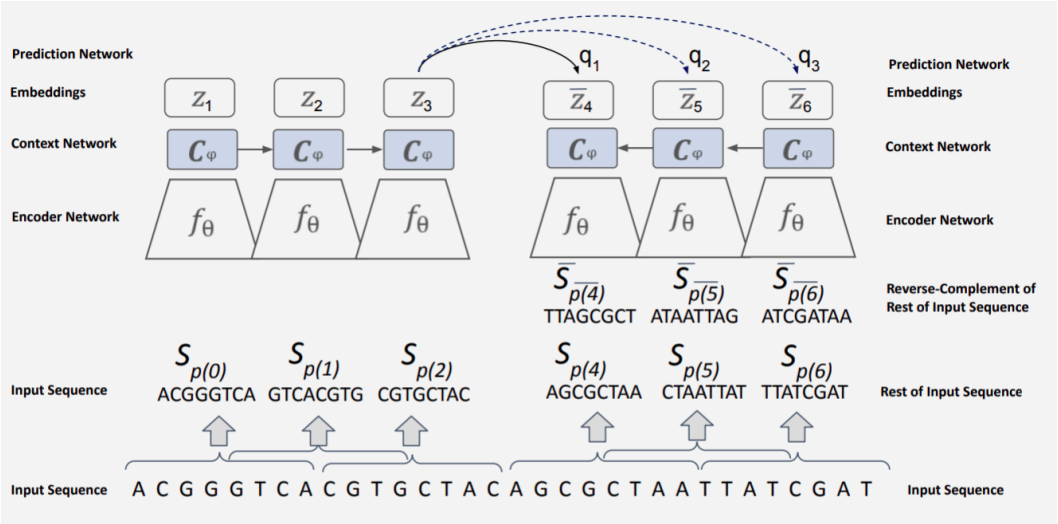

Figure 2: Implementation of Self-GenomeNet. Multiple representations $\mathbf{z}_i$ of subsequences of different lengths are created in a single step using the sequence-output of the recurrent context network $C_{phi}(\cdot)$ which follows the encoder network $f_\theta(\cdot)$. The same process happens to the reverse-complement of the given sequence, creating $\overline{\mathbf{z}}_i$. Different prediction layers q_m are used to predict representations that are a different number of patches apart.

one reverse context network that predicts each other, only a single context-network is trained that is both applied to the forward as well as to the RC sequence.

3.2 SELF-GENOMENET

The Self-GenomeNet architecture learns representations of varying-length genome sequences by encoding them in both convolutional and recurrent neural network layers, which are trained using a contrastive loss. The architecture enforces the symmetry implied by the RC while also making training more efficient. As depicted in Fig. 1, our proposed method initially creates two subsequences from an input sequence. The subsequences are taken by a representation network composed of the *encoder network* (f_θ) and the *context network* (C_ϕ). Later, on top of the embedding representation, the linear prediction layer (q_m) estimates the embedding of the other subsequence using a contrastive loss against random other subsequences.

More specifically, given are an unlabeled genome sequence of nucleotides $S_{1:N} = [s_1, s_2, \ldots, s_N]$ and its RC $\overline{S}_{N:1} = [\overline{s_N}, \overline{s_{N-1}}, \ldots, \overline{s_1}]$, where $s_i \in \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$ and $\overline{s_i}$ gives the complementary nucleotide, e.g. $\overline{\mathrm{A}} = \mathrm{T}$. We aim to produce an embedding of a subsequence $S_{1:t}$ that contrastively predicts the embedding of the RC of the remaining subsequence, $\overline{S}_{N:t'}$ for some $t' > t$.

For this, the sequence is first divided into P overlapping patches $S_{p(j)}$ with $p(j) = (j \cdot a + 1)$:$(j \cdot a + l)$, where l is the patch-length and a the patch stride length, and where $a < l$ for overlapping patches. Patches are first encoded with a convolutional neural network $f_\theta(\cdot)$. The resulting sequence of vectors $[f_\theta(S_{p(0)}), f_\theta(S_{p(1)}), \ldots, f_\theta(S_{p(P-1)})]$ is fed into a recurrent context network $C_\phi(\cdot)$, giving rise to embeddings $\mathbf{z}_i = C_\phi\big(\{f_\theta(S_{p(u)})\}_{u<i}\big)$ for $1 \leq i \leq P$. Patches of the RC, $\overline{S}_{\overline{p}(j)} = [\overline{s_{j \cdot a+l}}, \ldots, \overline{s_{j \cdot a+1}}]$ are equally fed into first f_θ then C_ϕ, giving rise to $\overline{\mathbf{z}_i} = C_\phi\big(\{f_\theta(\overline{S}_{\overline{p}(u)})\}_{u \geq i}\big)$. $\mathbf{z}_i$ is then a representation of $S_{1:((i-1) \cdot a+l)}$; likewise, $\overline{\mathbf{z}}_i$ represents $\overline{S}_{N:(i \cdot a+1)}$.

Training the encoder and context networks consists of predicting $\overline{\mathbf{z}}_{i+m}$ from $\mathbf{z}_i$, with $m > 0$, contrastively against corresponding embeddings from other, negative example sequences $S^{(k)-}$, i.e. against $\overline{\mathbf{z}}_{i+m}^{(k)-}$. For this, a linear prediction layer q_m is used, and the Noise Contrastive Estimation or InfoNCE loss (Gutmann and Hyvärinen, 2010), which maximizes the mutual information shared between forward sequence and its matching RC sequence, is used:

$$L_{i,m} = -\log \frac{\exp\left(\overline{\mathbf{z}}_{i+m}^{T} q_m(\mathbf{z}_i)\right)}{\exp\left(\overline{\mathbf{z}}_{i+m}^{T} q_m(\mathbf{z}_i)\right) + \sum_k \exp\left(\left(\overline{\mathbf{z}}_{i+m}^{(k)-}\right)^T q_m(\mathbf{z}_i)\right)}. \tag{1}$$

Negative samples $\overline{\mathbf{z}}_{i+m}^{(k)-}$ are efficiently produced by comparing against representations of other sequences loaded in the same mini-batch. Note that every sequence in the minibatch produces *two* negative examples for other sequences, giving $2(B-1)$ negative samples when using minibatch size B: The embedding of the RC of the last $(P-i-m)$ patches starting from path $(i+m)$ – i.e., the analog to $\overline{\mathbf{z}}_{i+m}$ coming from other sequences – and the embedding of the first $(P-i-m)$ patches of the normal forward-sequence, so the analog to $\mathbf{z}_{P-i-m}$. This is because of the invariance of the embedding method with respect to the RC. Embeddings are, however, always only contrasted with embeddings of sequences that have the same length as the positive example. This is to prevent the network from learning to encode the represented length directly to gain an advantage, which would not be an intrinsically interesting feature for downstream tasks.

One loss term is introduced for each index i, denoting the number of patches represented by $\mathbf{z}_i$. Furthermore, because it is possible to skip a different number of patches m between the representations being matched, one can sum over multiple values of m, each with its individual weights for predictor q_m. Because of the symmetry of the setup, the model is both predicting $\overline{\mathbf{z}}_{i+m}$ from $\mathbf{z}_i$ as described above, as well as $\mathbf{z}_i$ from $\overline{\mathbf{z}}_{i+m}$. The corresponding loss $\overline{L}_{i,m}$ then uses negative examples with the same length as $\mathbf{z}_i$. The ultimate loss for each individual sequence S is therefore:

$$L = \sum_{i,m} \left(L_{i,m} + \overline{L}_{i,m}\right).$$

4 Experiment Setup

Datasets and Tasks We evaluate our method with the following datasets: (1) **Virus** dataset: We downloaded all publicly available viral genomes from GenBank (Sayers et al., 2021). We allocated the FASTA files into training, validation, and test sets by approximate ratios of 70%, 20%, and 10%. We divided the dataset into two taxonomical classes as a classification task: bacteriophages vs. viruses that are not bacteriophages, based on provided annotations. The bacteriophage class contained approximately 1.0 billion nucleotides, the non-phage virus dataset approximately 0.5 billion nucleotides. (2) **Bacteria** dataset: We downloaded all publicly available bacteria genomes from GenBank, comprising approximately 83 billion nucleotides, similarly to how we collected the *Virus* dataset. This dataset was only used for unsupervised pre-training for transfer-learning, so no supervised task was defined. (3) The **DeepSEA** dataset, described in Zhou and Troyanskaya (2015), contains around 5 million human genome sequences, where each data sample contains 1000 nucleotides as input and a label vector for 919 binary chromatin features. We provide more details for pre-processing of the dataset in Appendix 8.

We used both the *Virus* dataset as well as the *DeepSEA* dataset to evaluate semi-supervised learning using representation learning: Training the self-supervised model on all samples within the datasets (without labels) and using the learned representation for supervised-learning using a reduced amount of label data. We used the *Bacteria* dataset to evaluate transfer-learning of representations: Training a self-supervised model on *Bacteria* data and using the learned weights to perform supervised learning on the *Virus* dataset.

Network architecture For the self-supervised training on the *Virus* dataset, the encoder network $f_\theta(\cdot)$ is a convolution layer with 1024 filters, a kernel size of 24, and a stride of 6. The context network $C_\phi(\cdot)$ is an LSTM layer with a hidden layer size of 512. When pre-training on the *Bacteria* dataset for transfer learning, a very similar structure is used. Partly in line with an increase in the sequence length of the data samples, the convolutional kernel size and the stride values are increased to 20 and 40, respectively. In order to fit more samples into a batch, the hidden layer size of the LSTM was decreased to 256. For the supervised training task, a linear layer on top of the last output of LSTM is added.

For the *DeepSEA* benchmark, we followed the DanQ network architecture by Quang and Xie (2016). The model consists of a convolutional, a max-pooling, and a bidirectional LSTM layer, followed by a hidden layer after all neurons are flattened. Dropout is used after both the max-pooling and the

LSTM layers. For unsupervised pretraining, we used the parts of the model before the LSTM layer as encoder network $f_\theta(\cdot)$, and an unidirectional LSTM as context network $C_\phi(\cdot)$. We then transferred the learned weights to the full DanQ model architecture for supervised learning.

Although this architecture has a similar structure compared to the model we used for *Virus*, there are also differences, especially when converting the model from the self-supervised architecture to the supervised (DanQ) model, which may affect the contribution of pre-trained models to the efficient classification task. We believe the most important difference is the proportion of neurons of the supervised model that are not trained in the self-supervised stage, which is approximately 96.4%. We expected this to limit the contribution of self-supervised models notably.

For both datasets a fully connected linear layer is used as prediction layer q_m.

Optimization For pretraining on both the *Virus* and the *Bacteria* dataset we used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0001, which performed better than 0.01, 0.001 and 0.0005 in preliminary experiments. We used the default values for other Adam hyperparameters from the Keras (Chollet et al., 2015) software: $\beta_1 = 0.9$, $\beta_2 = 0.999$. For the *DeepSEA* dataset, we use the RMSprop optimizer as used by Quang and Xie (2016), with a learning rate of 0.0001. The dropout rate of 0.2 and 0.5 are used respectively for the dropout layers as in the DanQ model (Quang and Xie, 2016). For all datasets, we used the default Keras weight initialization, which is the Glorot uniform initialization (Glorot and Bengio, 2010).

Throughout the self-supervised pre-training, we used the largest fitting minibatch size (power of 2) that worked in a GeForce RTX 2080 Ti, which was 512 for the virus dataset and 128 for the *Bacteria* and *DeepSEA* datasets. For supervised training, we used the same batch sizes for each method, which is 2048 for the *Virus* dataset and 512 for the *DeepSEA* dataset.

We provided more detail on optimization in Appendix 8.

Evaluation We use macro-averaged recall $Recall_M$ (Sokolova and Lapalme (2009)) for dataset evaluations on *Virus* because the dataset is imbalanced. We use Receiver Operating Characteristics (ROC), Area Under Curve (AUC), and Precision-Recall (PR) AUC for *DeepSEA* dataset for all 919 binary outputs, as done by Quang and Xie (2016). In this paper, it was also shown that the sparsity of positive binary targets($\sim 2\%$) in the dataset inflated the ROC AUC, and that PR AUC is a better indicator of performance since it does not take the number of true negatives into account.

5 RESULTS

To empirically compare our proposed framework to existing self-supervised models such as CPC (Oord et al., 2018), Contrastive-sc (Ciortan and Defrance, 2021) and a generative language model, we follow standard protocols for self-supervised learning and evaluate the unsupervised learned representation with a linear classification and semi-supervised tasks, as well as using transfer learning to different datasets and different tasks (Henaff, 2020).

Linear Evaluation The common evaluation protocol for self-supervised learning is based on freezing the representation networks (both base-encoder and context network) after unsupervised pretraining and then train a supervised linear classifier on top of them. The linear classifier is a fully connected layer followed by softmax-classification using cross-entropy loss, which is plugged on top of the context network after removing the prediction layer. Our linear evaluation was done on both *Virus* and *DeepSEA* with different approaches to explore and compare the effectiveness of our proposed method. Table 1 reports Macro-averaged recall on the test set of *Virus*, Table 2 shows the comparison of our model against the self-supervised baselines on *DeepSEA*. Based on the performance reported in both tables, our method obtains up to 14.2%-point and 10.4%-point improvement compared with contrastive-sc and CPC respectively. We achieve the best performance in the linear evaluation setting compared to all state-of-the-art methods.

Semi-supervised Training

We evaluate the performance of Self-GenomeNet's representation on a semi-supervised task. We pre-train the base and context models on unlabeled examples and fine-tune a classification model using a subset of dataset with labels. We follow the semi-supervised protocol explained in CPC-v2 (Henaff, 2020) and use the fixed splits of respectively 0.1%, 1%, and 10% of labeled training data. Table 3, Table 4 and Table 5 show the comparison of our performance against other deep learning

Method	$Recall_M$
CPC	58.7
Language Model	58.8
Contrastive-sc	54.9
Self-GenomeNet	**69.1**

Table 1: Macro-averaged recall (in %) under linear evaluation on *Virus*.

Method	ROC AUC				PR AUC			
	Q_1	Median	Mean	Q_3	Q_1	Median	Mean	Q_3
CPC	0.632	0.717	0.734	0.826	0.024	0.051	0.096	0.129
Language Model	0.603	0.683	0.699	0.784	0.018	0.038	0.069	0.091
Contrastive-sc	0.591	0.667	0.682	0.759	0.014	0.031	0.056	0.071
Self-GenomeNet	**0.657**	**0.745**	**0.757**	**0.847**	**0.032**	**0.078**	**0.120**	**0.161**

Table 2: Linear evaluation on the DeepSEA benchmark in terms of ROC AUC and PR AUC. We report the first quartile $Q1$, the median, the mean and the third quartile Q_3 values of the performance on the 919 outputs defined for the dataset.

methods on the virus dataset and the DeapSEA benchmark respectively. The result indicates using the proposed method we can outperform the baseline CPC and Language Models.

Transfer to Other Tasks We further assess the generalization capacity of the learned representation on virus dataset. We train our model unsupervised on the bacteria dataset and evaluate on the virus dataset. Table 6 provides a comparison of transfer learning performance of our self-supervised approach for the task of classification of viruses.

6 Ablation Studies and Discussions

We present multiple ablation studies on Self-GenomeNet to give an intuition of its behavior and performance: (i) the impact of learning representation from reverse-complement using the context network; (ii) estimating varying length of input-output by prediction network; (iii) robustness of our algorithm in the low-data regime.

Learning from the reverse complement of sequences Self-GenomeNet uses the embedding representation of a context network for downstream tasks which learns from reverse complement as a target for its predictions. We investigated the effect of using embedding targets derived by *reverse-complement* subsequences against using other potential ways of reading the target subsequences. To this end, we performed three experiments where the autoregressive model reads the target subsequences in *forward* direction, *reverse* direction, or *reverse-complement* direction. Table 7 compares the performance of our trained network in these ways.

Estimating varying length We investigated the effect of the change in length of target subsequences by comparing our method against the single-length or longer target subsequences. We divide the sequences into two nearly equal subsequences. Specifically, we split the sequences of 150 into 72 and 78, which are the two closest values that can be investigated using our previous architecture without any padding. We observed for linear evaluation setting on the virus dataset that the performance improved from 64.9% (macro-averaged recall) to 69.1%. One likely explanation is that, because we maximized the mutual information between the learned embeddings and varying-length target sequences with both short and long inputs.

Data efficient In this section, we study and compare the performance of Self-GenomeNet with other methods in the low-data regime. We train the models shown in Table 8 using 0.1%, 1%, 10%, 100% of the dataset. Based on the results reported in Table 8 and Table 5, with decreasing amounts of labeled data, supervised networks, CPC, and LM failed to generalize. When trained on unsupervised representations learned with semi-supervised settings or linear evaluation, these networks retain a much higher accuracy in this low-data regime. Indeed, the accuracy of supervised networks can be matched with significantly fewer labels.

Method	0.1%	1%	10%
Supervised	56.4	69.3	72.4
CPC	59.2	69.4	74.4
Language Model	61.8	71.5	75.5
Contrastive-sc	61.8	70.3	73.2
Self-GenomeNet	**67.2**	**75.3**	**78.2**

Table 3: Macro-averaged recall (in %) under semi-supervised learning on virus dataset using 0.1%, 1%, 10% training example.

Method	0.1%		1%		10%	
	Median	Mean	Median	Mean	Median	Mean
Supervised	0.699	0.715	0.763	0.776	**0.864**	**0.865**
CPC	0.717	0.732	0.768	0.780	0.861	0.863
Language Model	0.715	0.729	0.758	0.774	0.857	0.859
Contrastive-sc	0.711	0.726	0.751	0.764	0.847	0.848
Self-GenomeNet	**0.739**	**0.753**	**0.784**	**0.795**	0.862	**0.865**

Table 4: ROC AUC under semi-supervised learning for DeepSEA benchmark when base-network is unfreezed

7 CONCLUSION

In this paper, we presented Self-GenomeNet, a framework for contrastive learning of nucleotide-level genomic data. Self-GenomeNet is composed of an encoding network followed by a context network and uses a prediction layer for training. The network extracts compact latent representations by predicting other representations of future observations while both enforcing and exploiting the symmetry implied by the necessary reverse-complement equivariance. Self-GenomeNet is optimized end-to-end using unsupervised contrastive loss. We tested the learned representations by linear evaluation, semi-supervised learning, and transfer to other tasks and achieved state-of-the-art performance on the two public bioinformatics benchmarks. Moreover, we showed the ability of our method for training in low-data regime, estimating varying length of sequences, and learning and predicting of forward, reverse, and reverse-complement of sequences.

Reproducibility Statement: The source code is available in the supplementary material.

Method	0.1%		1%		10%	
	Median	Mean	Median	Mean	Median	Mean
Supervised	0.041	0.074	0.090	0.119	0.222	**0.239**
CPC	0.049	0.089	0.082	0.125	0.224	0.236
Language Model	0.044	0.081	0.075	0.116	0.218	0.232
Contrastive-sc	0.040	0.073	0.065	0.105	0.206	0.222
Self-GenomeNet	**0.072**	**0.115**	**0.093**	**0.138**	**0.229**	0.235

Table 5: PR AUC under semi-supervised evaluation for DeepSEA benchmark.

	Fine-tuned	Fixed
Supervised	92.0	
CPC	95.5	76.1
Language Model	92.6	67.9
Contrastive-sc	93.4	56.4
Self-GenomeNet	**96.7**	**82.1**

Table 6: Macro-averaged recall (in %) under transfer learning evaluation on virus dataset. We evaluated the results on both fine-tuned and fixed base networks, where the base network is the network trained din the self-supervision.

Forward	Reverse	Reverse-Complement
68.2	67.7	69.1

Table 7: The achieved macro-averaged recall (in%) for virus dataset under linear evaluation setting with fixed pre-trained weights and sequence length of 150.

Method	Semi-supervised Learning				Linear Evaluation			
Labeled Dataset	0.1%	1%	10%	100%	0.1%	1%	10%	100%
Supervised	56.4	69.3	72.4	78.7				
CPC	59.2	69.4	74.4	79.4	55.8	59.2	58.5	58.7
Language Model	61.8	71.5	75.5	79.4	56.6	59.2	58.8	58.8
Contrastive-sc	61.8	70.3	73.2	78.7	54.9	55.3	55.2	54.9
Self-GenomeNet	**67.2**	**75.3**	**78.2**	**80.1**	**64.3**	**68.6**	**69.0**	**69.1**

Table 8: Macro-averaged recall (in %) under semi-supervised and linear evaluation for virus dataset and sequence length 150

REFERENCES

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

François Chollet et al. Keras. `https://keras.io`, 2015.

Madalina Ciortan and Matthieu Defrance. Contrastive self-supervised clustering of scrna-seq data. *BMC bioinformatics*, 22(1):1–27, 2021.

Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28:3079–3087, 2015.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Gökcen Eraslan, Žiga Avsec, Julien Gagneur, and Fabian J Theis. Deep learning: new computational modelling techniques for genomics. *Nature Reviews Genetics*, 20(7):389–403, 2019.

Vladimir Gligorijević, P Douglas Renfrew, Tomasz Kosciolek, Julia Koehler Leman, Daniel Berenberg, Tommi Vatanen, Chris Chandler, Bryn C Taylor, Ian M Fisk, Hera Vlamakis, et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):1–14, 2021.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Wenran Li, Wing Hung Wong, and Rui Jiang. Deeptact: predicting 3d chromatin contacts via bootstrapping deep learning. *Nucleic acids research*, 47(10):e60–e60, 2019.

Amy X Lu, Haoran Zhang, Marzyeh Ghassemi, and Alan M Moses. Self-supervised contrastive learning of protein representations by mutual information maximization. *BioRxiv*, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Daniel Quang and Xiaohui Xie. Danq: a hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic acids research*, 44(11):e107–e107, 2016.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15), 2021.

Eric W Sayers, Mark Cavanaugh, Karen Clark, Kim D Pruitt, Conrad L Schoch, Stephen T Sherry, and Ilene Karsch-Mizrachi. Genbank. *Nucleic acids research*, 49(D1):D92–D96, 2021.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Reverse-complement parameter sharing improves deep learning models for genomics. *bioRxiv*, page 103663, 2017.

Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437, 2009.

Rongjie Wang, Tianyi Zang, and Yadong Wang. Human mitochondrial genome compression using machine learning techniques. *Human genomics*, 13(1):1–8, 2019.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction, 2021.

Hannah Zhou, Avanti Shrikumar, and Anshul Kundaje. Towards a better understanding of reverse-complement equivariance for deep learning models in regulatory genomics. *bioRxiv*, pages 2020–11, 2021.

Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10):931–934, 2015.

8 APPENDIX

Processing of Files For virus-bacteria dataset, data samples are created by reading consecutive samples taken from random regions of FASTA files. Compared to division of FASTA files into same-length non-intersecting sequences, this technique increases number of samples that can be created out of FASTA files. In order to provide randomness, either maximum 90% and 10% of the file is processed for virus files and bacteria files respectively, or maximum 64 data samples are created from a file, before next FASTA file is processed. For phage/non-phage classification on virus dataset, we set the sequence length to 150. For the transfer learning setting, the sequence length was set to 1000, in order to compare the capabilities of the methods for a different sequence length.

Optimization of Self-Supervised Methods
For all architectures and all our experiments, we used the same encoder and context models for a fair comparison.

For CPC, we adopted the Henaff (2020) implementation of CPC for 1-dimensional data and use an RNN (LSTM) and a single fully connected layer on top of LSTM layer for prediction as in Oord et al. (2018). For pretraining on virus dataset, every second feature from 4th to 10th derived by encoder network is predicted with different linear layers. Thus, the representations derived from the following 60 nucleotides are predicted. For bacteria pretraining, the following 2nd to 10th features are predicted, making the prediction span of 200 nucleotides. For DeepSEA dataset, every second feature from 3rd to 20th features are predicted, making the span up to 260 nucleotides.

For Contrastive-sc (Ciortan and Defrance, 2021), we used the masking rate of 0.9 as recommended by the paper and did not use additional linear layer on top of the final outputs of LSTM.

For the language modelling, we followed different strategy for different datasets. For virus and bacteria dataset, the model predicts the following 60 and 200 nucleotides respectively, which are taken from outside of the sequence. For DeepSEA, the nucleotides outside the sequence is not provided within the dataset. Thus, the last 260 nucleotides are estimated from the first 740 nucleotides during pretraining.

In the supervised training of virus datasets, we firstly train only the final linear layer while the pretrained base model is frozen. Once this layer is optimized, starting from this point, we tune the whole network.

For DeepSEA dataset, we directly optimize the whole model for both freezed and unfreezed base-networks.