MutFormer: A context-dependent transformer-based model to predict deleterious missense mutations from protein sequences in the human genome

Theodore T Jiang¹,², Li Fang¹, Kai Wang¹,³

¹ Raymond G. Perelman Center for Cellular and Molecular Therapeutics, Children’s Hospital of Philadelphia, Philadelphia, PA 19104, USA
² Palisades Charter High School, Pacific Palisades, CA 90272, USA
³ Department of Pathology and Laboratory Medicine, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA 19104, USA

*Correspondence should be addressed to wangk@chop.edu (K.W.) and fangl@chop.edu (L.F.).
Abstract

Various machine-learning models, including deep neural network models, have already been developed to predict deleteriousness of missense (non-synonymous) mutations. Still, potential improvements to the current state of the art may benefit from a fresh look at the biological problem using more sophisticated self-adaptive machine-learning approaches. Recent advances in the natural language processing field show transformer models—a type of deep neural network—to be particularly powerful at modeling sequence information with context dependence. In this study, we introduce MutFormer, a transformer-based model for the prediction of deleterious missense mutations. MutFormer uses reference and mutated protein sequences from the human genome as the primary features. It uses a combination of self-attention layers and convolutional layers to learn both long-range and short-range dependencies between amino acid mutations in a protein sequence. We pre-trained MutFormer on reference protein sequences and mutated protein sequences resulting from common genetic variants observed in human populations. Next, we examined different fine-tuning methods to successfully apply the model to deleteriousness prediction of missense mutations. Finally, we evaluated MutFormer’s performance on multiple testing data sets. We found that MutFormer showed similar or improved performance over a variety of existing tools, including those that used conventional machine-learning approaches (e.g., support vector machine, convolutional neural network, recurrent neural network). We conclude that MutFormer successfully considers sequence features that are not explored in previous studies and could potentially complement existing computational predictions or empirically generated functional scores to improve our understanding of disease variants.
Introduction

Whole-exome and whole-genome sequencing technologies are powerful tools for the detection of genetic mutations. A typical human genome has 4.1 million to 5.0 million variants when compared with the reference genome sequence (The 1000 Genomes Project Consortium, 2015), (the average exome captures genomic regions that account for 1-2% of the human genome). Therefore, distinguishing or prioritizing a small number of disease-related variants from such a large number of background variants becomes a key challenge in understanding genome and exome sequencing data. In particular, interpretation of non-synonymous single nucleotide variants (nsSNVs) is of major interest, because missense mutations in proteins account for over half of the current known variants responsible for human-inherited disorders, especially Mendelian diseases, where the mutations have high penetrance (Stenson, et al., 2020). Unlike frameshift indels or splicing mutations in canonical splice sites that have high likelihood to alter protein function, missense mutations change only a single amino acid, thus, most of them may not have significant impacts on protein function. Population-specific allele frequencies, such as those inferred from the ExAC (Karczewski, et al., 2017) and gnomAD (Karczewski, et al., 2020) databases, can be useful to filter out common missense variants that likely to be neutral, and mutation databases such as ClinVar (Landrum, et al., 2017) and HGMD (Stenson, et al., 2020) can be valuable resources to find previously reported mutations that may be deleterious. Still, a large number of missense variants from exome sequencing are not yet definitively documented; therefore, the functional interpretation of such variants remains an crucial task.

Numerous computational tools have been developed to predict the deleteriousness or pathogenicity of missense mutations (Liu, et al., 2011; Liu, et al., 2020; Liu, et al., 2016; Thusberg, et al., 2011), however, as shown by multiple recent publications, the accuracy of predictive algorithms still has room for improvement. Databases such as dbNSFP have now documented these whole-exome prediction scores for different prediction algorithms, in an effort to facilitate the development of improved functional assessment algorithms. However, depending on the evaluation data sets that were used, most algorithms for missense variant prediction are 65-80% accurate when examining known disease variants, and only approximately 43.4% of pairwise prediction correlations between different predictive algorithms are greater than 0.5 (Liu, et al., 2020). Many conflicting predictions can be made between different algorithms, which motivated the development of several “ensemble”-based scoring systems that combine multiple prediction algorithms, such as MetaSVM, REVEL, CADD and others. In fact, predictions combined from different algorithms are considered as a single piece of evidence according to the ACMG-AMP 2015 guidelines (Richards, et al., 2015). In addition, most existing computational algorithms are based on similar or related information (e.g., evolutionary conservation scores, mutation tolerance scores); potential improvements to the current state of the art could benefit from a fresh look at the biological problem using more sophisticated self-adaptive machine-learning approaches that examine additional types of information.

In other previous prediction algorithms, deep learning-based sequence models have been demonstrated as effective in modeling variant function. These existing methods primarily utilized convolutional neural networks (CNNs) to model sequences, however, recently, advances in deep learning have shown transformer models to be particularly powerful for modeling sequential data. The modern transformer model, Bidirectional Encoder Representations from Transformers (BERT) (Devlin, et al., 2019; Vaswani, et al., 2017) relies on its central mechanism, the self-attention. The usage of the self-attention allows the transformer model to achieve unprecedented ability to model relationships within individuals in a sequence – crucial in the comprehension of linear sequences. In the past three years, transformers have achieved state-of-the-art performances on a broad
range of natural language processing (NLP) tasks (Devlin, et al., 2019; Lan, et al., 2019; Liu, et al., 2019; Yang, et al., 2019), and transformers even outperformed more traditional CNN-based models on image recognition tasks (Dosovitskiy, et al., 2020). As of late, transformers have also been successfully applied for modeling protein structure in Alphafold2 (Jumper, et al., 2021). Part of the reason for the success of transformers may be due to its increased ability to handle subtle context dependency through a multi-head attention mechanism, and the ability to compute attentions in parallel to greatly speed up computation over typical RNN-based algorithms.

In biological contexts, each amino acid in a given protein sequence also exerts its function in a context dependent manner, including both local dependency (such as forming a short signal peptide that was recognized by cellular machinery) and long-distance dependency (such as being close to another amino acid in three-dimensional structure to form a binding site for ligands). Therefore, we hypothesize that transformer models would be capable of a more effective modeling of protein sequences, somewhat similar to how transformers have ‘transformed’ the field of natural language processing and language translation over the past few years.

In this study, we propose MutFormer, a transformer-based model to assess deleteriousness of missense mutations. MutFormer is an adaption of the BERT architecture (Devlin, et al., 2019) to protein contexts, with appropriate modifications to incorporate protein-specific characteristics. MutFormer can analyze protein sequences directly, with or without any homology information or additional data. Our experiments show that MutFormer is capable of matching or outperforming current methods in the deleteriousness prediction of missense variants.

**Methods**

**The MutFormer model**

MutFormer is based on the BERT architecture (Devlin, et al., 2019). A central component of the classical BERT model is its bidirectional self-attention. This mechanism uses a 2D matrix to calculate and model context between all positions in a given sequence, enabling the learning of long-range dependencies between individuals. The convolution, on the other hand, another mechanism capable of learning dependencies, is better suited for short-range dependencies: convolutions are more capable of prioritizing localized patterns via filters, while the repeated application of convolution filters, which are required for the relating of farther individuals in a sequence, often weakens long range dependencies. MutFormer takes advantage of both self-attention layers and convolutional layers, to effectively learn both long-range and short-range dependencies.

In language modeling tasks, words or sub-words are short-range features of the sequence. The original BERT uses a fixed WordPiece vocabulary which contains common words/sub-words in the training corpus (Wu, et al., 2016). This vocabulary cannot be tuned during the pre-training and fine-tuning process; therefore, a spelling error may introduce an out-of-vocabulary word that will hinder the model’s interpretation ability of a given sequence. In protein sequences, “words” correspond to key subsequences or patterns of amino acids. These “words” can often be changed due to mutations, and furthermore, are unknown. Recent studies showed that vocabulary-free models (e.g. byte-level models) are more robust to noise and perform better on tasks that are sensitive to spelling (Xue, et al., 2021). Therefore, instead of using a fixed vocabulary, convolutional layers placed in between the embedding layers and the transformer body, which we call its adaptive vocabulary, are used by MutFormer. MutFormer uses these convolutions to learn its own vocabulary over the course of the training process, incorporating nonlinear patterns into
its vocabulary via the convolution filters (Figure 1). The weights of the convolutional layers are tuned during both the pre-training and fine-tuning processes.

In addition to the direct implementation of the adaptive vocabulary, a model which instead featured an integrated adaptive vocabulary was also tested. This is because while in theory, the convolutions should be able to create a representation of a protein that will enable the model to best interpret the sequence, in practice, some information that is present in the original raw embedded sequence may be lost through the convolutions. To solve this, the integrated vocabulary model, instead of feeding the embeddings through the convolutions linearly, utilizes skip connections that result in the convolutions acting as if they were “integrated” into the embedding layers, allowing the transformer model to access both the convolution filtered representation of the sequence as well as the original sequence.

**Pretraining of MutFormer on human protein sequences**

We pretrained MutFormer on human reference protein sequences (all isoforms) and protein sequences caused by non-synonymous SNVs with > 1% population frequency in the gnomAD database (Karczewski, et al., 2020). The total number of protein sequences used in the pretraining step was about 120K. The maximum input length of MutFormer is 1024, where protein sequences longer than 1024 are cut into segments. During cutting, all segments were retained, except for segments less than 50 amino acids long, which were discarded. A “B” letter was added to the beginning of a sequence and a “J” letter was added to the end of a sequence so that the true start and end of a protein sequence was also indicated (“B” and “J” are not included in the current amino acid alphabet). In total, we accumulated ~150K training datapoints from these 120K protein sequences.

The original BERT model uses a self-supervised pretraining objective of recovering an original sequence from corrupted (masked) input, from which high-dimensional representations of the sequence are learned. Similar to BERT, the pretraining objective of MutFormer was to predict masked amino acid residues from altered sequences. For the masked residue prediction task, amino acid residues were randomly masked, by either replacing them with a [MASK] token or a random other amino acid. To ensure enough context was present for the model, a maximum number of 20 amino acids were masked per sequence. In addition, in order to facilitate the learning of more dependency knowledge and minimize overfitting, we utilized dynamic masking throughout the duration of training: the masked amino acids were changed randomly for each epoch of data trained on. Note that for MutFormer, the “next sentence prediction” objective in the original BERT was removed, because protein sequences in aggregate, unlike their natural language counterpart, do not form ‘paragraphs’ with logical connections between sentences. The pre-training was performed on a single cloud Tensor Processing Unit (TPU) hardware accelerator (TPU v2-8) on the Google Cloud Platform.

We pretrained MutFormer (utilizing our adaptive vocabulary) with three different model sizes (Table 1), as well as a MutFormer model with 8 transformer layers that utilized the integrated adaptive vocabulary (Figure 1). For comparison purposes, we also pretrained two models without the adaptive vocabulary. We named these models MutBERT, as indication of the usage of the original BERT architecture (Table 1). The hyperparameters used for pre-training are displayed in Table S1, and loss and accuracy for the pretraining task are listed in Table S2.
Training and Evaluation data for variant deleteriousness prediction (Fine-tuning task)

For MutFormer’s fine-tuning data, we obtained 84K manually annotated pathogenic missense SNVs from the Human Gene Mutation Database (HGMD) Pro version 2016 (Stenson, et al., 2020). We combined this set with SNPs from the gnomAD database (Karczewski, et al., 2020) with allele frequency >0.1%, the vast majority of which are assumed to be benign. Although a commonly used allele frequency threshold for benign variants is 1%, we used 0.1% instead, in order to achieve 1) the exclusion of pretraining data from the fine-tuning data (MutFormer’s pretraining data utilized the threshold of 1%); 2) the approximate balancing of the number of benign and pathogenic examples in the dataset. Any mutations that appeared in both the HGMD database and benign set were removed, and from the remaining data, all the mutations present in the original pretraining data (i.e., >1% in gnomAD) were also eliminated. The final benign set contained 61K variants. The combination of the pathogenic set and the benign set was randomly split into a training set and an independent validation set. The independent validation set was then isolated from the training set by reference sequence: to prevent memorization from the training set to the validation set, mutations were deleted from the training set if the mutated sequence or the mutation’s reference sequence was present in the validation set. Any mutations that appeared in both the HGMD database and benign set were removed, and from the remaining data, all the mutations present in the original pretraining data (i.e., >1% in gnomAD) were also eliminated. The final benign set contained 61K variants. The combination of the pathogenic set and the benign set was randomly split into a training set and an independent validation set. The independent validation set was then isolated from the training set by reference sequence: to prevent memorization from the training set to the validation set, mutations were deleted from the training set if the mutated sequence or the mutation’s reference sequence was present in the validation set. The independent validation set contains 5282 benign variants and 3145 deleterious variants. Note that this independent validation set, despite its independent selection, is still prone to bias because of its similarity with the training set. For this reason, this set is only used to internally compare the performance of models trained within these experiments, and separate testing sets compiled from various different sources are used for the comparison of MutFormer with other existing methods of deleteriousness prediction.

Fine-tuning MutFormer for prediction of deleterious mutations

MutFormer was fine-tuned on the training data described above. Mutated protein sequences were generated using ANNOVAR (Wang, et al., 2010), with each sequence containing exactly one mutation. To obtain the best possible results in deleteriousness prediction, we tested three different fine-tuning methods (Figure 2):

1) Per residue classification. The input is a single protein sequence that may be either benign or deleterious. The model is tasked with classifying each amino acid in the protein sequence as benign or deleterious. Amino acids that are identical to the reference sequence are labeled as benign, and the mutated residue is labeled as benign or deleterious, depending on the true classification of the sequence (loss and metrics for classification are calculated on only the mutation site). This corresponds to the token classification task or named-entity recognition (NER) task in NLP.

2) Single sequence classification. The input is a single protein sequence. The model is tasked with classifying the entire sequence as deleterious or benign (via the [CLS] token). This corresponds to the sentence classification (e.g., sentiment analysis) in NLP.

3) Paired sequence classification. For each mutation, both the reference sequence and mutated sequence are given. The model classifies the aggregate of the sequences as deleterious or benign through comparison of the two sequences. This was inspired by the sentence similarity problem (e.g., the MRPC task) in NLP.
To find the best fine-tuning method, model, and sets of hyperparameters, we performed two different internal comparison tests using our independent validation set. Test 1 compared the MutFormer architecture with the classical BERT architecture, as well as the three different fine-tuning methods, using different hyperparameters. Test 2 compared the use of the integrated vocabulary against the original MutFormer architecture. Prior to both tests, some initial testing (with our independent validation set) on random hyperparameters was done to establish a set of hyperparameter values that worked well with all combinations of methods/models included in the test. For test 1, the initial set of hyperparameters was established based on the three fine-tuning methods (per residue, single sequence, paired sequence) and three different models (MutBERT\textsubscript{8L}, MutBERT\textsubscript{10L}, MutFormer\textsubscript{8L}) being tested. For test 2, the initial set of hyperparameters was established based on the four models (MutFormer\textsubscript{8L}, MutFormer\textsubscript{10L}, MutFormer\textsubscript{12L}, MutFormer\textsubscript{8L} (with integrated vocab)) being tested. Full list of hyperparameters used are detailed in Table 3, and results of test 1 and test 2 are displayed in Figure 3 and Figure 4, respectively.

To create a model capable of the best possible performance in deleteriousness prediction, in addition to utilizing protein sequence analysis, when training our final models (displayed in our final testing) (Figure 4), MutFormer also incorporated prediction values from previous methods published in literature. Previous methods’ predictions were included in the following way. First, using ANNOVAR, predicted scores for all mutations within a newly generated test set were obtained from the dbNSFPv3 database (Liu, et al., 2016). These scores were standardized between 1 and 2, and all missing predictions were assigned values of 0. A fully connected dense layer was connected to these inputs, and the output of this dense layer was concatenated with the original model output. Another dense layer after this concatenated result was then connected to the output node to produce the end prediction result. This incorporation strategy prevents the model from becoming reliant on external predictions, limiting the weighting of sequence analysis vs external predictions to about 1:1 in MutFormer’s prediction. This intended usage ratio is consistent to our findings when looking into the weights of the various finetuned models’ prediction layers, where we found, for all models, an approximate weighting of half for MutFormer’s sequence analysis, and half for external predictions.

**Testing MutFormer against existing methods of deleteriousness prediction**

To assess the performance of MutFormer against existing methods of deleteriousness prediction, a total of 6 testing datasets were used. Out of these 6 datasets, one is sourced from two disease specific but non-gene-specific studies, three are non-gene-specific and non-disease-specific datasets, and two are gene-specific mutation datasets (details for each testing dataset used are outlined in Table 2). For each dataset, filtering was performed using the reference sequences to ensure that no bias was present: all mutations that shared reference sequences with any mutation present in the pretraining data were deleted from the testing sets, and all reference sequences present in any of the independent test sets were removed from the fine-tuning training data prior to model training.

In order to allow for a more comprehensive evaluation of the performance of MutFormer with different levels of “fit” on a wide range of data (models with a higher “fit” will perform better on more similar data but worse on more dissimilar data; models with lower “fit” will have the opposite tendencies), different MutFormer models with varying hyperparameters that affected a model’s level of “fit” were trained (an analysis of MutFormer’s performance with varying levels of fit is analogous to an analysis of an ROC curve, where the performances of MutFormer on similar vs. dissimilar data is compared for different fit levels). In this test, the number of freezing layers and batch size were varied, while all other hyperparameters were set to the best ones found during our hyperparameter test 2 (see above). From these results, testing sets 4-6 showed more
variation with different fit parameters than sets 1-3 did (results from all test runs are summarized in Figure S2). All models were then tested on all testing datasets, and the overall best performing model across all testing datasets was used to represent MutFormer in our comparison. Batch sizes of 16, 32, and 64 were tested in conjunction with freezing layer numbers of 0, 5, 6, and 8 (full hyperparameter description in Table 3). Freezing layer numbers are defined as the number of transformer body layers that were frozen (for MutFormer 8L with integrated adaptive vocabulary, our current best performing model, 8 layers is the total number of transformer body layers). For any freezing layer number greater than 0, the embedding layers were frozen as well (through testing on our validation set we found that leaving the embedding layers trainable while freezing the transformer layers significantly decreased performance). Each model was trained for 14k steps, and checkpoint steps 6k, 8k, 11k, 12k, and 14k were evaluated.

When fine-tuning our final models (used in our comparison of MutFormer vs existing methods), to increase MutFormer’s overall generalization ability and limit overfitting, data augmentation was implemented for the fine-tuning training data: for all epochs of data, each datapoint had a 50% chance of being altered. Those that were selected to be altered would be trimmed down to anywhere from 50% length of the original sequence to 100% length of the original sequence.Trimming was done around the mutation site to ensure the mutation site on average stayed in the same location (in the middle) in the sequence before and after trimming. Epochs were also shuffled independently of each other.

**Results**

**Effect of the adaptive vocabulary on the pretraining task**

During pretraining, we trained models of different sizes for both the MutFormer architecture and MutBERT (the MutFormer model without the adaptive vocabulary) architecture (Table 1). The loss and accuracy on the pretraining task are shown in Error! Reference source not found.. According to our results, the accuracy of MutFormer8L on the pretraining task test set was 54.5% higher than MutBERT8L, indicating the advantage of the adaptive vocabulary. In addition, the accuracy of MutFormer8L was 27.7% higher than that of MutBERT10L, despite the latter having two more transformer layers and 10M more parameters (the subscript of the model name indicates only the number of transformer layers but does not consider the two convolutional layers). This outperformance despite smaller size verifies that the improved performance of the adaptive vocabulary was not simply due to the additional number of parameters or additional layers.

**Performance of different fine-tuning methods and hyperparameters**

As part of our internal comparison test 1, we fine-tuned MutFormer and MutBERT using three methods: per residue classification, single sequence classification and paired sequence classification (Figure 2, see Methods for details). The Receiver Operator Characteristic (ROC) curves and corresponding Area Under Curve (AUC) for deleteriousness prediction are shown in Figure 3A-C. Figure 3D shows a summary of the performance comparison of the three methods; paired sequence classification performed best, followed by per residue classification, and the optimal results were achieved by using a maximal input sequence length of 512 (usage of the paired sequence method means an aggregate sequence length of 1024) (Figure 3F). Upon examination of the performance of different model architectures, as shown in Figure 3E,
MutFormer_{8L} outperformed MutBERT_{8L} and MutBERT_{10L} for each fine-tuning method, indicating the advantage of the MutFormer architecture.

Use of the Integrated Adaptive Vocabulary

In our internal comparison test 2, MutFormer (with integrated adaptive vocabulary)'s performance in paired sequence classification of our independent validation set was compared to that of the other three original MutFormer architecture models (Figure S1). ROC curves are shown in Figure S1A, and a summary comparison histogram of the four different models tested is shown in Figure S1B. Overall, margins of difference are small, but based on the results, the performance of the MutFormer model with integrated adaptive vocabulary is higher than that of the original MutFormer model; even with only 8 transformer layers, the integrated adaptive vocabulary model outperformed MutFormer_{12L} in deleteriousness prediction of missense mutations for our independent validation set.

Comparison with existing variant prediction methods

As paired sequence classification performed best, for the comparison of MutFormer versus other methods, this fine-tuning method was used. MutFormer’s best overall performance was achieved by training our best model, MutFormer (with integrated adaptive vocab), on a batch size of 32, and 0 freezing layers (for full hyperparameter descriptions see Table 3). MutFormer’s performance was compared against a variety of existing methods of deleteriousness prediction (Adzhubei, et al., 2010; Carter, et al., 2013; Choi, et al., 2012; Chun and Fay, 2009; Davydov, et al., 2010; Dong, et al., 2015; Garber, et al., 2009; Gulko, et al., 2015; Kircher, et al., 2014; Ng and Henikoff, 2003; Pollard, et al., 2010; Qi, et al., 2021; Quang, et al., 2015; Rentzsch, et al., 2019; Reva, et al., 2011; Schwarz, et al., 2014; Schwarz, et al., 2010; Shihab, et al., 2013; Shihab, et al., 2015; Siepel, et al., 2005; Wu, et al., 2021). In our comparison, existing methods’ predictions were processed in the following way: each method’s predictions were standardized from 0-1, based on prediction values of all possible missense mutations present in the dbNSFPv3 database (Liu, et al., 2016). Missing predictions were automatically assigned a prediction value of 0. Both non-inverted and inverted prediction identities (1=deleterious, 0=benign and 0=deleterious, 1=benign) were tested across our fine-tuning training data, and inversion of scores was done accordingly for each algorithm being compared. For dataset 4, which only included rare benign examples, a threshold for each existing method, chosen by taking the point closest to the upper left corner on an ROC curve based on MutFormer’s fine-tuning training data, was used to calculate a method specificity for each method. Upon analyzing the performances of the different testing datasets, we found that the best overall performing MutFormer model outperforms previous methods of deleteriousness prediction in non-gene-specific and non-disease-specific datasets (more similar to MutFormer’s fine-tuning training dataset: sets 2-4), and is only outperformed by MVP on set 1, which is a combination of two disease-specific but non-gene-specific datasets. Of note, set 1’s original goal was to test utility in real clinical sequencing studies, and contains de novo mutations, which may or may not correspond to the disease. On a dataset scale, overall correlation is assumed to be present, this lack of direct correspondence is why all methods’ ROC AUC are close to 0.5. On the two gene-specific databases (sets 5-6), which contain data less similar to that of MutFormer’s fine-tuning data, MutFormer’s performance in comparison to other methods expectedly drops, while still matching the performance of various existing methods. Full testing results are displayed in Figure 4.
**Precomputed deleteriousness scores for all missense mutations**

To facilitate future use by other studies, we precomputed deleteriousness scores for all missense mutations using the best performing MutFormer model. The inference was done on a cloud TPU device (v2-8), which took about 11.5 hours. These scores can be directly used in the ANNOVAR software to annotate missense variants identified from genome or exome sequencing. Since the scores are organized in a flat file format, they can be also used in other functional annotation software tools to complement the dbNSFP database, which has a variety of other prediction scores for missense mutations in the human genome.

**Discussion**

In the current study, we present MutFormer, a transformer-based machine-learning model to predict the deleteriousness of non-synonymous SNVs using protein sequence as the primary feature. We pretrained MutFormer on reference protein sequences and alternative protein sequences resulting from common genetic variants in the human genome and tested different fine-tuning methods for deleteriousness prediction. During our evaluation processes, MutFormer outperformed multiple commonly used methods and had comparable performance with other methods even when tested on data sets that were less similar to MutFormer’s training data (gene-specific data, sets 5-6). Below we discuss several advantages and limitations of the MutFormer method and its computational package.

Although a large number of computational tools have been developed over the years on predicting the deleteriousness of non-synonymous mutations, to the best of our knowledge, MutFormer is among the first batch of tools that utilize transformer models to adapt the biological sequence analysis problem as a language analysis problem. A similar model of note is ProtBERT (Elmaggar, et al., 2021), a previous application of the BERT architecture to protein contexts. Despite both utilizing the transformer architecture, MutFormer differs from ProtBERT in several key aspects: 1) MutFormer makes use of an adaptive vocabulary via convolution layers while ProtBERT uses a fixed vocabulary; 2) MutFormer was pretrained on human protein sequences and common variants, while ProtBERT was trained on reference sequences of all species with sequence information; 3) MutFormer’s primary goal was deleteriousness prediction while ProtBERT focused on subcellular localization of proteins and secondary structure prediction.

In addition, the training process of MutFormer is simple and straightforward. MutFormer uses self-supervised pretraining strategy and therefore does not require any labeled data. For this reason, a large model with hundreds of millions of parameters can be trained on a large amount of non-curated data. On this note, while the current study focused on human genome exclusively, it is conceivable to include other well-annotated genomes from other species in future studies to see whether increased complexity in the sequence space during pre-training can further improve performance. In the fine-tuning stage, MutFormer learns the deleteriousness of mutations based on the labeled training data as well as its understanding of protein sequence already learned in the pretraining stage, allowing a small amount of fine-tuning data to be effectively used to achieve an improved result.

Furthermore, transformers consider attention, which is not only useful for understanding context in language processing problems, but could also give important insights into the deleteriousness effect amino acids can have under different sequence contexts: for example, the same amino acid motif may have completely different functionality depending on the surrounding sequence.
One potential exploration, for example, could involve the analysis of the filters in the adaptive vocabulary to identify potential amino acid patterns of interest.

There are also several limitations of the current study. First, the training data and testing data sets are still of limited size, and testing on large-scale experimentally or clinically supported datasets would result in a more effective evaluation of usability. In the future, MutFormer can be evaluated on large-scale genome sequencing data followed by manual review, to determine whether it helps prioritize deleterious variants in clinical sequencing settings. Second, because of computational limitations, we did not fully test all parameters during training. As a result, it is likely that our results are not completely optimized; larger models using longer maximum sequence lengths would also be able to outperform the current MutFormer models (for example, a 12L integrated vocab MutFormer should perform noticeably better than the current 8L model). Third, in the deleteriousness prediction of missense mutations, it is likely impossible for a given model to obtain all required evidence from sequence data alone, so incorporation of other features, such as 3D structure (i.e., analyzing 3D structure to scale attention with 3D distance, or labelling sections as belonging to beta sheets or alpha helices for better prediction of deleteriousness), methylation, clinical phenotypic information (i.e., using this knowledge to prioritize certain genes), and other features that could significantly affect proteins’ behavior, could reasonably improve overall understanding and thus performance.

In summary, MutFormer is a novel transformer-based method to predict the functional effects of missense mutations. We hope that MutFormer can bring new insights to the bioinformatics community, by being a language model capable of improving our understanding of the language of proteins. Given that MutFormer used complementary information that other bioinformatics tools developed for deleteriousness prediction, we also envision that they could be combined to reach consensus on predictions, which may be useful in implementing current clinical guidelines.

Data and Code Availability

The source code to run MutFormer, all 6 pretrained models, a reproducible workflow, and the best performing fine-tuned models are available at the GitHub repository: https://github.com/WGLab/MutFormer/.

Competing interests

The authors declare no competing interests.

Acknowledgements

We would like to acknowledge the TPU Research Cloud (TRC) program by Google, which provided us with TPUs for the duration of the project. The study is in part supported by NIH grant GM132713 (KW) and the CHOP Research Institute.
Adzhubei, I.A., et al. A method and server for predicting damaging missense mutations. *Nat Methods* 2010;7(4):248-249.

Carter, H., et al. Identifying Mendelian disease genes with the variant effect scoring tool. *BMC Genomics* 2013;14 Suppl 3:S3.

Chun, S. and Fay, J.C. Identification of deleterious mutations within three human genomes. *Genome Res* 2009;19(9):1553-1561.

Choi, Y., et al. Predicting the functional effect of amino acid substitutions and indels. *PLoS One* 2012;7(10):e46688.

Devlin, J., et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics; 2019. p. 4171-4186.

Dong, C., et al. Comparison and integration of deleteriousness prediction methods for nonsynonymous SNVs in whole exome sequencing studies. *Hum Mol Genet* 2015;24(8):2125-2137.

Dosovitskiy, A., et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In.; 2020. p. arXiv:2010.11929.

Elnaggar, A., et al. ProtTrans: Towards Cracking the Language of lifes Code Through Self-Supervised Deep Learning and High Performance Computation methods for missense variants. *Nucleic Acids Res* 2021;49(3):310-315.

Garber, M., et al. Identifying novel constrained elements by exploiting biased substitution patterns. *Bioinformatics* 2009;25(12):i54-62.

Gulko, B., et al. A method for calculating probabilities of fitness consequences for point mutations across the human genome. *Nat Genet* 2015;47(3):276-283.

Jumper, J., et al. Highly accurate protein structure prediction with AlphaFold. *Nature* 2021;596(7873):583-589.

Karczewski, K.J., et al. The mutational constraint spectrum quantified from variation in 141,456 humans. *Nature* 2020;581(7809):434-443.

Karczewski, K.J., et al. The ExAC browser: displaying reference data information from over 60 000 exomes. *Nucleic Acids Res* 2017;45(D1):D840-d845.

Kircher, M., et al. A general framework for estimating the relative pathogenicity of human genetic variants. *Nat Genet* 2014;46(3):310-315.

Lan, Z., et al. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In.; 2019. p. arXiv:1909.11942.

Landrum, M.J., et al. ClinVar: improving access to variant interpretations and supporting evidence. *Nucleic Acids Research* 2017;46(D1):D1062-D1067.

Li, J., et al. Performance evaluation of pathogenicity-computation methods for missense variants. *Nucleic Acids Research* 2018;46(15):7793-7804.

Liu, X., Jian, X. and Boerwinkle, E. dbNSFP: A lightweight database of human nonsynonymous SNPs and their functional predictions. *Human Mutation* 2011;32(8):894-899.

Liu, X., et al. dbNSFP v4.0: a comprehensive database of transcript-specific functional predictions and annotations for human nonsynonymous and splice-site SNVs. *Genome Med* 2020;12(1):103.

Liu, X., et al. dbNSFP v3.0: A One-Stop Database of Functional Predictions and Annotations for Human Nonsynonymous and Splice-Site SNVs. *Human Mutation* 2016;37(3):235-241.

Liu, Y., et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. In.; 2019. p. arXiv:1907.11692.

Ng, P.C. and Henikoff, S. SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Res* 2003;31(13):3812-3814.

Pollard, K.S., et al. Detection of nonneutral substitution rates on mammalian phylogenies. *Genome Res* 2010;20(1):110-121.

Qi, H., et al. MVP predicts the pathogenicity of missense variants by deep learning. *Nat Commun* 2021;12(1):510.

Quang, D., Chen, Y. and Xie, X. DANN: a deep learning approach for annotating the pathogenicity of genetic variants. *Bioinformatics* 2015;31(5):761-763.

Rentzsch, P., et al. CADD: predicting the deleteriousness of variants throughout the human genome. *Nucleic Acids Res* 2019;47(1):D886-D894.

Reva, B., Antipin, Y. and Sander, C. Predicting the functional impact of protein mutations: application to cancer genomics. *Nucleic Acids Res* 2011;39(17):e118.

Richards, S., et al. Standards and guidelines for the interpretation of sequence variants: a joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology. *Genet Med* 2015;17(5):405-424.

Sasidharan Nair, P. and Vihinen, M. VariBench: a benchmark database for variations. *Hum Mutat* 2013;34(1):42-49.
Schwarz, J.M., et al. MutationTaster2: mutation prediction for the deep-sequencing age. *Nat Methods* 2014;11(4):361-362.

Schwarz, J.M., et al. MutationTaster evaluates disease-causing potential of sequence alterations. *Nat Methods* 2010;7(8):575-576.

Shihab, H.A., et al. Predicting the functional, molecular, and phenotypic consequences of amino acid substitutions using hidden Markov models. *Hum Mutat* 2013;34(1):57-65.

Shihab, H.A., et al. An integrative approach to predicting the functional effects of non-coding and coding sequence variation. *Bioinformatics* 2015;31(10):1536-1543.

Siepel, A., et al. Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. *Genome Res* 2005;15(8):1034-1050.

Stenson, P.D., et al. The Human Gene Mutation Database (HGMD((R))): optimizing its use in a clinical diagnostic or research setting. *Hum Genet* 2020;139(10):1197-1207.

The 1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature* 2015;526(7571):68-74.

Thusberg, J., Olatubosun, A. and Vihinen, M. Performance of mutation pathogenicity prediction methods on missense variants. *Hum Mutat* 2011;32(4):358-368.

Vaswani, A., et al. Attention is All you Need. *ArXiv* 2017;abs/1706.03762.

Wang, K., Li, M. and Hakonarson, H. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Res* 2010;38(16):e164.

Wu, Y., et al. Improved pathogenicity prediction for rare human missense variants. *Am J Hum Genet* 2021.

Wu, Y., et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. In.; 2016. p. arXiv:1609.08144.

Xue, L., et al. ByT5: Towards a token-free future with pre-trained byte-to-byte models. In.; 2021. p. arXiv:2105.13626.

Yang, Z., et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In.; 2019. p. arXiv:1906.08237.
The **MutFormer model architecture.** A system of positional, label, and token embeddings is used to first vectorize the input tokens. Two convolution layers then learn the adaptive vocabulary (for the integrated adaptive vocabulary model, skip connections are used). A bidirectional transformer body with self-attention considers context and learns patterns in protein sequences. The output embeddings are used for token or sequence level classification.
Different fine-tuning methods tested in this study. A) Per residue classification. The input is a protein sequence that may contain a variant. Each residue (amino acid) is given a label of benign/deleterious. Benign variants and residues that are identical to the reference sequence are labeled as benign. The fine-tuning task is to predict the label of each amino acid. This is similar to token classification problems (e.g., named-entity recognition) in NLP. B) Single sequence classification. The input is a protein sequence that may contain a deleterious variant. The embedding of the [CLS] token in the last layer is used to predict whether the sequence contains a deleterious variant. This is similar to sentence classification problems (e.g., sentiment analysis) in NLP. C) Sequence pair classification. The input is a pair of two sequences: a reference protein sequence and a mutated protein sequence (with a benign or deleterious variant in the center). The embedding of the [CLS] token in the last layer is used to predict whether the mutated sequence contains a deleterious variant. This is similar to sentence pair classification problems (e.g., sentence similarity) in NLP.
Figure 3

Fine-tuning internal comparison test 1: Performance comparison of different fine-tuning methods and MutFormer architecture/MutBERT architecture. A-C) ROC curves for two different model architectures (MutFormer/MutBERT) and three fine-tuning methods (per residue, single sequence, paired sequence). Labels are in the following format: "mn_{model name}_sl_{max input sequence length}: {ROCAUC}". D) Performance comparison of three different fine-tuning methods, using AUC scores shown in panels A-C. Whiskers indicate min and max values. E) Performance comparison of three pretrained models: MutBERT8L, MutBERT10L and MutFormer8L. F) Performance of different max input sequence lengths. E, F) The results are mean ± Standard Error of the Mean (SEM).
Figure 4

Performance comparison with existing methods. ROC curves and accuracy / AUC scores of MutFormer and different existing methods of deleteriousness prediction evaluated on 6 different databases. Labels are formatted in the following way: {Method}: ROC=(ROC AUC) A: MVP_set – dataset compiled by MVP, a previous method of deleteriousness prediction, containing 450 negative examples and 1874 positive examples. B: Meta_SVM_LR_set_1 – dataset compiled by a previous paper that originally outlined the MetaSVM and MetaLR methods, containing 56 negative examples and 35 positive examples. C: MetaSVM_LR_set_2 – same source as Meta_SVM_LR_set_1, containing 5866 negative examples and 4115 positive examples. D: MetaSVM_LR_set_3 – same source as Meta_SVM_LR_set_1 and set_2, containing 2422 negative examples (because only negative examples are present, ROC is invalid in this case; instead, specificity is used for comparison). E: Varibench_PPARG – dataset from Varibench for the peroxisome proliferator activated receptor (gamma) gene, containing 4671 negative examples and 3428 positive examples. F: Varibench_TP53 – dataset from Varibench for the TP53 gene which codes for the tumor suppressor P53 protein, containing 3444 negative examples and 4505 positive examples.
**Tables**

**Table 1**

Model sizes of the pretrained models (subscripts in the models’ names denote the number of self-attention layers in the respective transformer bodies) (“-“ indicates same as above).

| Model Name    | Hidden Layers | Hidden Size | Intermediate Size | Input length | # of parameters |
|---------------|---------------|-------------|-------------------|--------------|-----------------|
| MutBERT<sub>8L</sub> | 8             | 768         | 3072              | 1024         | 58M             |
| MutBERT<sub>10L</sub> | 10            | 770         | -                 | -            | 72M             |
| MutFormer<sub>8L</sub> | 8             | 768         | -                 | -            | 62M             |
| MutFormer<sub>10L</sub> | 10            | 770         | -                 | -            | 76M             |
| MutFormer<sub>12L</sub> | 12            | 768         | -                 | -            | 86M             |
| MutFormer<sub>6L</sub> (with integrated adap vocab) | 8             | 768         | -                 | -            | 64M             |

Note: MutFormer<sub>12L</sub> has the same size as BERT<sub>Base</sub>

**Table 2**

Details for each testing dataset:

| Testing dataset | Testing dataset composition                                                                 | Compilation year |
|-----------------|---------------------------------------------------------------------------------------------|------------------|
| Set 1: MVP_set  | Dataset compiled by a previous method, MVP (Qi, et al., 2021). Used by MVP to assess its own utility in real genetic studies. | 2021             |
|                 | Composition after filtering (see methods for filtering description):                         |                  |
|                 | • 1101 de novo mutations from 2 autism spectrum disorder (ASD) studies that were associated with ASD in patients. |                  |
|                 | • 778 de novo mutations from a congenital heart disease (CHD) study that were associated with CHD in patients. |                  |
|                 | • 459 de novo mutations of non-ASD-associated mutations from unaffected siblings in the Simons Simplex Collection, sourced from an ASD study. |                  |
| Set 2: Meta_SVM_LR_set_1 | Dataset compiled by a previous method, Meta_SVM/Meta_LR (Dong, et al., 2015). Used to assess Meta_SVM and Meta_LR’s performance against other methods. | 2015             |
|                 | Composition after filtering:                                                                 |                  |
|                 | • 56 pathogenic examples compiled from recent Nature Genetics publications at the time      |                  |
|                 | • 35 benign examples from the CHARGE (Cohorts for Heart and Aging Research in Genetic Epidemiology) database, which focuses on identifying genes underlying heart, lung, and blood diseases. |                  |
| Set 3: Meta_SVM_LR_set_2 | Dataset from same source as set 2 (Meta_SVM_LR_set_1)                                         | 2015             |
|                 | Composition after filtering:                                                                 |                  |
|                 | • 4135 pathogenic examples from Varibench testing dataset II for missense mutations (Sasidharan Nair and Vihinen, 2013) (Varibench is a dataset designed |                  |
specifically for the testing of prediction methods for pathogenicity).

- 5884 benign examples also from Varibench testing dataset II

Set 4: Dataset from same source as set 2 (Meta_SVM_LR_set_1) set 3 (Meta_SVM_LR_set_2).

Composition after filtering:
- 2422 benign examples from the CHARGE database.

Set 5: Dataset from Varibench (Sasidharan Nair and Vihinen, 2013).

Compiled by a study that specifically aimed to create datasets for assessing computational models’ performance in pathogenicity prediction of missense mutations (Li, et al., 2018). Focused on the PPARG gene which codes for the gamma member of the PPAR (Peroxisome Proliferator-activated Receptor) family of nuclear receptors, which can be linked to the pathology of diseases including diabetes, atherosclerosis, and cancer.

Composition after filtering:
- 145 pathogenic variants from the experimentally validated Missense InTerpretation by Experimental Response (MITER) database.
- 2207 benign variants from the same source.

Set 6: Dataset from same source as Set 5 (Varibench_PPARG).

Focused on the TP53 gene which codes for tumor protein p53, a tumor suppressor gene.

Composition after filtering:
- 608 pathogenic examples from the IARC database (database specific for TP53), labeled for significantly changing the gene expression level of the TP53 gene.
- 531 benign examples also from IARC which did not change expression level significantly.
Table 3
MutFormer Fine-tuning hyperparameter specifications (1 of 2) ("-" indicates same as above):

| Test / model | Model Architecture | Fine-tuning method | Initial / end Learning Rate | Training steps |
|--------------|--------------------|--------------------|----------------------------|----------------|
| Internal comparison test 1 | MutFormer8L, MutBERT8L, MutBERT10L | Per residue, Single Sequence, Paired Sequence | 1e-5 / (2e-7 to 1e-6) | • 4k (fine-tuning method 1 and 2), □ 10k (fine-tuning method 3) |
| Internal comparison test 2 | MutFormer8L, MutFormer10L, MutFormer12L (with integrated adap vocab) | Paired Sequence | 1e-5 / 1.4e-6 | 12k |
| MutFormer Comparison with others | MutFormer8L (with integrated adap vocab) | - | 1e-5 / 3e-9 | 14k (evaluated on 6k, 8k, 11k, and 12k) |
| MutFormer final model | MutFormer8L (with integrated adap vocab) | - | - | 12k |

MutFormer Fine-tuning hyperparameter specifications (2 of 2):

| Test / model | Max Input Sequence Length | Batch Size | Gradient Clip amount | Weight Decay (For Adam Optimizer) | Freezing layers | External predictions? |
|--------------|---------------------------|------------|----------------------|----------------------------------|----------------|---------------------|
| Internal comparison test 1 | 64, 128, 256, 512 | 16 | No clipping | 0.01 | 0 | No |
| Internal comparison test 2 | 256, 512 | 16, 32, 64 | - | - | 0 | No |
| MutFormer Comparison with others | 512 | 16, 32, 64 | - | - | • 0, • 5, • 6, • 8 | Yes |
| MutFormer final model | 512 | 32 | - | - | 0 | Yes |
MutFormer Supplementary Materials

Figures

Figure S1

Fine-tuning internal comparison test 2: Performance comparison of MutFormer vs MutFormer (with integrated adaptive vocab). A) ROC curves for two different model architectures (MutFormer / MutFormer (with integrated adaptive vocab)) tested on varying sequence lengths and batch sizes. The labels are in the following format: "mn_{model name}_sl_{max input sequence length}_bs_{batch size}: ROC={ROCAUC}". B) Performance comparison of the four different models: MutFormer_{8L}(with integrated adaptive vocab), MutFormer_{8L}, MutFormer_{10L}, and MutFormer_{12L}.
Figure S2

Summary of all testing dataset runs for MutFormer: All MutFormer test runs for varying levels of “fit” are displayed as solid lines; the chosen best performing overall run, which represented MutFormer in our comparison vs other existing methods, is bolded and dashed. Note that for dataset 4, which contains only negative examples, the y-coordinate in the graph corresponds to specificity instead of ROCAUC.
Tables

Table S1

Hyperparameters during pretraining for each model ("-" indicates same as above).

| Model Name         | Initial Learning Rate | Learning Rate Decay Per Step | Batch Size | Steps per epoch | Gradient Clip amount | Weight Decay |
|--------------------|-----------------------|------------------------------|------------|-----------------|----------------------|--------------|
| MutBERT$_{8L}$     | 2e-5                  | 1.33e-11                     | 64         | 2300            | 1.0                  | 0.01         |
| MutBERT$_{10L}$    | -                     | -                            | -          | -               | -                    | -            |
| MutFormer$_{8L}$   | -                     | -                            | -          | -               | -                    | -            |
| MutFormer$_{10L}$  | -                     | -                            | -          | -               | -                    | -            |
| MutFormer$_{12L}$  | -                     | 1.00e-11                     | 32         | 4600            | -                    | -            |
| MutFormer$_{8L}$   | -                     | 1.33e-11                     | 64         | 2300            | -                    | -            |
| (with integrated adap vocab) | - | - | - | - | - | - |

Table S2

Loss and accuracy on the pretraining task (masked residue prediction)

| Model Name         | Training split | Test split |
|--------------------|----------------|------------|
|                    | Loss | Accuracy | Loss | Accuracy |
| MutBERT$_{8L}$     | 1.4641 | 0.5538 | 2.0297 | 0.4021 |
| MutBERT$_{10L}$    | 1.1360 | 0.6504 | 1.7248 | 0.4863 |
| MutFormer$_{8L}$   | 0.9872 | 0.6984 | 1.2274 | 0.6212 |
| MutFormer$_{10L}$  | 0.8560 | 0.7384 | 1.0961 | 0.6631 |
| MutFormer$_{12L}$  | 0.8338 | 0.7460 | 1.0727 | 0.6730 |
| MutFormer$_{8L}$   | 0.8305 | 0.7475 | 1.0590 | 0.6930 |
| (with integrated adap vocab) | - | - | - | - |
Table S3

The numbers of SNVs in the test sets that were missing from each method

| Method              | Reference                                         | Set 1 | Set 2 | Set 3 | Set 4 | Set 5 | Set 6 |
|---------------------|---------------------------------------------------|-------|-------|-------|-------|-------|-------|
| MutFormer           |                                                   | 0     | 0     | 0     | 0     | 0     | 0     |
| SIFT                | (Ng and Henikoff, 2003)                           | 16    | 2     | 203   | 72    | 102   | 104   |
| PolyPhen2-HDIV      | (Adzhubei, et al., 2010)                         | 5     | 0     | 100   | 22    | 21    | 15    |
| PolyPhen2-HVAR      | (Adzhubei, et al., 2010)                         | 5     | 0     | 100   | 22    | 21    | 15    |
| LRT                 | (Chun and Fay, 2009)                             | 130   | 10    | 701   | 22    | 501   | 256   |
| MutationTaster      | (Schwarz, et al., 2014)                          | 8     | 0     | 65    | 25    | 14    | 14    |
| MutationAssessor    | (Reva, et al., 2011)                             | 27    | 2     | 228   | 23    | 64    | 69    |
| FATHMM              | (Shihab, et al., 2013)                           | 142   | 2     | 484   | 50    | 186   | 207   |
| PROVEAN             | (Choi, et al., 2012)                             | 12    | 1     | 160   | 67    | 46    | 51    |
| VEST3               | (Carter, et al., 2013)                           | 1     | 0     | 74    | 22    | 13    | 13    |
| CADD                | (Kircher, et al., 2014; Rentzsch, et al., 2019)  | 0     | 0     | 46    | 22    | 0     | 0     |
| DANN                | (Quang, et al., 2015)                            | 0     | 0     | 46    | 22    | 0     | 0     |
| FATHMM-MKL          | (Shihab, et al., 2015)                           | 0     | 0     | 46    | 22    | 0     | 0     |
| MetaSVM             | (Dong, et al., 2015)                             | 1     | 0     | 46    | 22    | 3     | 3     |
| MetaLR              | (Dong, et al., 2015)                             | 1     | 0     | 46    | 22    | 3     | 3     |
| fitCons             | (Gulko, et al., 2015)                            | 87    | 3     | 1195  | 159   | 433   | 433   |
| GERP++              | (Davydov, et al., 2010)                          | 0     | 0     | 50    | 22    | 7     | 7     |
| PhyloP-7way-vertebrate| (Pollard, et al., 2010)                         | 0     | 0     | 47    | 22    | 1     | 1     |
| PhyloP-20way-mammalian| (Pollard, et al., 2010)                         | 0     | 0     | 47    | 22    | 0     | 0     |
| PhastCons-7way-vertebrate| (Siepel, et al., 2005)                        | 0     | 0     | 47    | 22    | 1     | 1     |
| PhastCons-20way-mammalian| (Siepel, et al., 2005)                        | 0     | 0     | 47    | 22    | 0     | 0     |
| SiPhy-29way (log odds)| (Garber, et al., 2009)                         | 2     | 0     | 68    | 22    | 12    | 9     |
| VARITY_ER           | (Wu, et al., 2021)                               | 95    | 6     | 516   | 77    | 428   | 200   |
| VARITY_ER_LOO       | (Wu, et al., 2021)                               | 95    | 6     | 516   | 77    | 428   | 200   |
| VARITY_R            | (Wu, et al., 2021)                               | 95    | 6     | 516   | 77    | 428   | 200   |
| VARITY_R_LOO        | (Wu, et al., 2021)                               | 95    | 6     | 516   | 77    | 428   | 200   |
| MVP                 | (Qi, et al., 2021)                               | 0     | 4     | 969   | 6     | 228   | 171   |

Note: the total numbers of SNVs in the test sets are outlined in Table 2