# *EnhancerDetector*: Enhancer Discovery from Human to Fly via Interpretable Deep Learning

**Luis M. Solis [1], Geyenna Sterling-Lentsch [2], Marc S. Halfon [2] and Hani Z. Girgis [1*]**

[1]The Bioinformatics Toolsmith Laboratory, Department of Electrical Engineering and Computer Science, Texas A&M University-Kingsville, Kingsville, 78363, Texas, USA and [2]Department of Biochemistry, University at Buffalo-State University of New York, Buffalo, 14203, New York, USA

Received YYYY-MM-DD; Revised YYYY-MM-DD; Accepted YYYY-MM-DD

## ABSTRACT

**Enhancers are essential non-coding DNA elements that regulate gene expression, yet their accurate identification remains a major challenge. We introduce *EnhancerDetector*, a convolutional neural network-based framework for cross-species enhancer prediction that combines high accuracy with biological interpretability. Trained on human data, *EnhancerDetector* achieves strong performance across human, mouse, and fly datasets, consistently outperforming existing methods in precision and F1 score. It generalizes effectively to datasets generated using diverse experimental assays. An ensemble strategy further improves prediction reliability by reducing false positives — critical for genome-wide applications. *EnhancerDetector* supports fine-tuning on new species and retains strong performance even when adapted with as few as 20,000 enhancer sequences, making it ideal for newly sequenced genomes with limited experimental data. For interpretability and visualization, we apply class activation maps to identify sequence regions predictive of enhancer activity. Experimental validation in transgenic flies confirms the predictive power of *EnhancerDetector*: five of six tested candidates drove reporter expression, and four exhibited expression patterns supported by prior literature. These analyses highlight distinct sequence and contextual features that confer what we term enhancerness: enhancer sequences possess a characteristic, identifiable signature. Together, these findings position *EnhancerDetector* as a practical, accurate, and interpretable framework for enhancer discovery across diverse species.**

## INTRODUCTION

Enhancers are regulatory elements that play a critical role in the transcriptional activation of genes. These non-coding DNA sequences can be located upstream, downstream, or even within the target gene itself. Their ability to function across long genomic distances is facilitated by the three-dimensional architecture of a genome (Hariprakash and Ferrari, 2019, Zaugg *et al.*, 2022). Enhancers are essential for processes such as cell differentiation, development, and the response to environmental signals. Moreover, enhancer dysfunction is increasingly recognized as a contributor to various medical issues, collectively termed "enhanceropathies" (Zaugg *et al.*, 2022). Thus, accurate identification is crucial for understanding gene regulation and its implications in health and disease.

Despite their biological significance, identifying enhancers remains a considerable challenge. Enhancers exhibit high locational variability and are often found within extensive regions of non-coding DNA, making them difficult to detect (Hariprakash and Ferrari, 2019). This challenge is compounded by the cell-type-specific nature of enhancers: an enhancer active in one cell type may be inactive in another (Hariprakash and Ferrari, 2019).

Traditional experimental methods for enhancer identification have included:

- Chromatin immunoprecipitation followed by sequencing (ChIP-seq): Detects protein-DNA interactions, typically involving transcription factors or histone modifications, with various computational tools used to analyze sequence motifs and genomic features (Park, 2009).
- Chromatin accessibility assays: These assays identify open regions of chromatin, often corresponding to active enhancers. Common approaches include ATAC-seq, which employs the Tn5 transposase to insert sequencing adapters into accessible regions (Buenrostro *et al.*, 2015); FAIRE-seq, which isolates nucleosome-depleted DNA fragments (Giresi *et al.*, 2007); and DNase-seq, which uses DNase I digestion to identify hypersensitive sites across a genome (Boyle *et al.*, 2008).
- High-throughput Chromosome Conformation Capture (Hi-C): Captures the interactions between different regions of the genome by cross-linking chromatin and then digesting the DNA with a restriction enzyme (Belton *et al.*, 2012).

*To whom correspondence should be addressed. Tel: +1 361 5932094; Fax: +1 361 5932981; Email: hani.girgis@tamuk.edu

© YYYY The Author(s)

- Reporter assays: Classical *in vivo* and *in vitro* methods where candidate enhancer sequences are inserted upstream of a reporter gene (e.g., luciferase or GFP) to test activity (Goto *et al.*, 1989, Harding *et al.*, 1989, Hiromi and Gehring, 1987).
- Genome-wide reporter assays: High-throughput versions of reporter assays that allow for large-scale screening of candidate enhancers in a tissue or cell-type-specific context (Arnold *et al.*, 2013, Jenett *et al.*, 2012, Tokusumi *et al.*, 2017).
- Chromatin profiling: Identifies enhancer-associated chromatin states based on histone modifications and chromatin accessibility across the genome (Klemm *et al.*, 2019).
- Enhancer RNA (eRNA) detection: Identifies active enhancers based on the presence of bidirectionally transcribed non-coding RNAs produced at enhancer loci (Kim *et al.*, 2010).
- CRISPR/Cas9-based approaches: Functionally validate enhancers by perturbing their sequences through deletion or repression to assess regulatory consequences (Catarino and Stark, 2018).
- Antibody-based approaches: Use of antibodies to target enhancer-associated proteins such as p300 or CBP in ChIP-type experiments (Visel *et al.*, 2009).

However, these approaches can be resource-intensive, may not capture all enhancer types, and often require validation.

Computational methods such as utilizing machine learning algorithms and neural networks have become regularly used and shown promise in learning more about enhancer identification. A wide variety of computational techniques have been employed in existing enhancer identification tools. Traditional machine learning approaches are exemplified by tools such as iEnhancer-2L (Liu *et al.*, 2015), which uses a two-layer model based on sequence features; EnhancerPred (Jia and He, 2016), which incorporates several machine learning algorithms; and iEnhancer-EL (Liu *et al.*, 2018), which applies ensemble learning. More recent tools have adopted deep learning strategies. A recent review (Winter *et al.*, 2025) highlights the growing importance of deep learning-based sequence-to-function models as powerful "biological oracles" — models capable of accurately predicting enhancer activity from raw DNA sequences. For instance, Enhancer-DSNet (Asim *et al.*, 2020) and iEnhancer-DCLA (Liao *et al.*, 2022) rely on deep neural networks trained on raw DNA sequences, while iEnhancer-GAN (Yang *et al.*, 2021b) leverages generative adversarial networks. Attention-based models such as ADH-Enhancer (Mehmood *et al.*, 2024) and architecture-specific variants like iEnhancer-EBLSTM (Niu *et al.*, 2021), which uses enhanced bidirectional LSTM networks, and iEnhancer-DCSV (Jia *et al.*, 2023), which employs DenseNet with convolutional block attention, have also been proposed. Additional approaches include iEnhancer-XG (Cai *et al.*, 2020), a two-layer predictor using XGBoost and multiple feature extractors, and iEnhancer-RD (Yang *et al.*, 2021a), which combines special features with a deep neural network. Enhancer-IF (Basith *et al.*, 2021) represents a hybrid ensemble-based strategy that integrates predictions from 35 baseline models — spanning five machine learning algorithms and seven feature encoding schemes — followed by meta-modeling and final ensemble aggregation, with a specific emphasis on cell-type-specific enhancer prediction.

*Despite their methodological diversity, the vast majority of these tools either are no longer functional, suffer from significant usability limitations, or lack rigorous experimental*

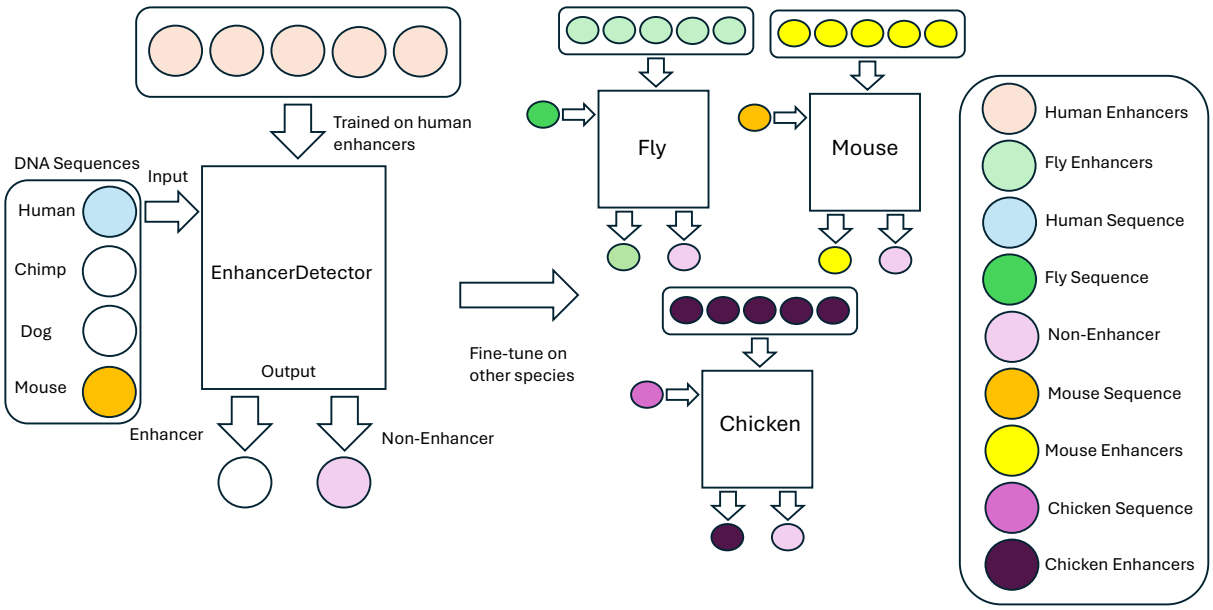



**Figure 1.** General Architecture and Cross-Species Adaptability of *EnhancerDetector*. The left panel depicts the base model trained on the human CATlas dataset, which can be directly applied to closely related vertebrate species with minimal performance degradation. The right panel highlights *EnhancerDetector*'s support for fine-tuning, enabling the human-trained model to be adapted for more distantly related species. For illustrative purposes, fine-tuning is shown for fruit fly, mouse, and chicken, though *the approach is broadly applicable to any species with sufficient enhancer data.*

*validation.* A common issue is the lack of long-term accessibility; numerous tools become unavailable over time due to inactive web servers or the absence of publicly available source code, hindering reproducibility and further research. Additionally, several tools suffer from insufficient documentation and lack user-friendly interfaces, making them difficult to implement without significant reverse engineering. Even when source code is provided, compatibility issues often arise; they fail to support common input formats containing ambiguous nucleotides (e.g., 'N'), which are prevalent in genomic data. Moreover, some tools depend on proprietary or unavailable libraries, complicating installation and execution. In our attempts, multiple experienced computational scientists were unable to successfully install or execute many of these tools, raising serious concerns about accessibility for biologists without computational expertise. *Finally, the enhancers predicted by most of these tools were never validated experimentally.*

Given the growing demand for reliable enhancer identification tools and the widespread limitations of existing ones, we developed a new approach that addresses many issues. We introduce *EnhancerDetector*, a deep convolutional neural network-based binary classifier designed to identify enhancer sequences with high accuracy and usability. *EnhancerDetector* processes input sequences of 400 base pairs and classifies them as either enhancers or non-enhancers. It relies only on common, well-maintained Python libraries, ensuring easy installation. *EnhancerDetector* exhibits broad utility for enhancer discovery across diverse biological contexts, demonstrating strong generalization across datasets, species, and experimental frameworks.

## MATERIALS AND METHODS

### Overview

*EnhancerDetector* is a flexible enhancer prediction framework that uses deep learning to classify genomic sequences as enhancers or non-enhancers. Given a sequence in FASTA format, the tool outputs a confidence score representing the likelihood that the input sequence functions as an enhancer. *EnhancerDetector* is centered on a convolutional neural network, with an optional ensemble mode to boost prediction confidence. These models are trained on curated enhancer datasets from both human and fruit fly genomes, and support fine-tuning for application to new species using limited enhancer data. *EnhancerDetector* also employs class activation maps, an explainable AI technique that highlights the most influential regions within input sequences. An overview of the architecture is shown in Figure 1.

### Enhancer Dataset

For the human classifier, we utilized a comprehensive set of putative enhancer sequences curated from the Cis-element Atlas (CATlas, which we will henceforth refer to as the CATlas dataset. This dataset was generated through the application of single-nucleus open chromatin accessibility sequencing (snATAC-seq) across 222 distinct cell types derived from both adult and fetal human tissues. The identified regions of open chromatin should encompass a diverse repertoire of regulatory elements, including transcriptional enhancers. The sequences within the CATlas dataset are uniformly 400 base pairs (bp) in length. For the *Drosophila melanogaster* classifier, we integrated distal DNase I hypersensitive sites identified by the Stamatoyannopoulos Laboratory (Thomas *et al.*, 2011), distal scATAC-seq regions from the Furlong Laboratory (Cusanovich *et al.*, 2018), and experimentally validated *cis*-regulatory modules (CRMs) sourced from REDfly (Keränen *et al.*, 2019).

### Preprocessing

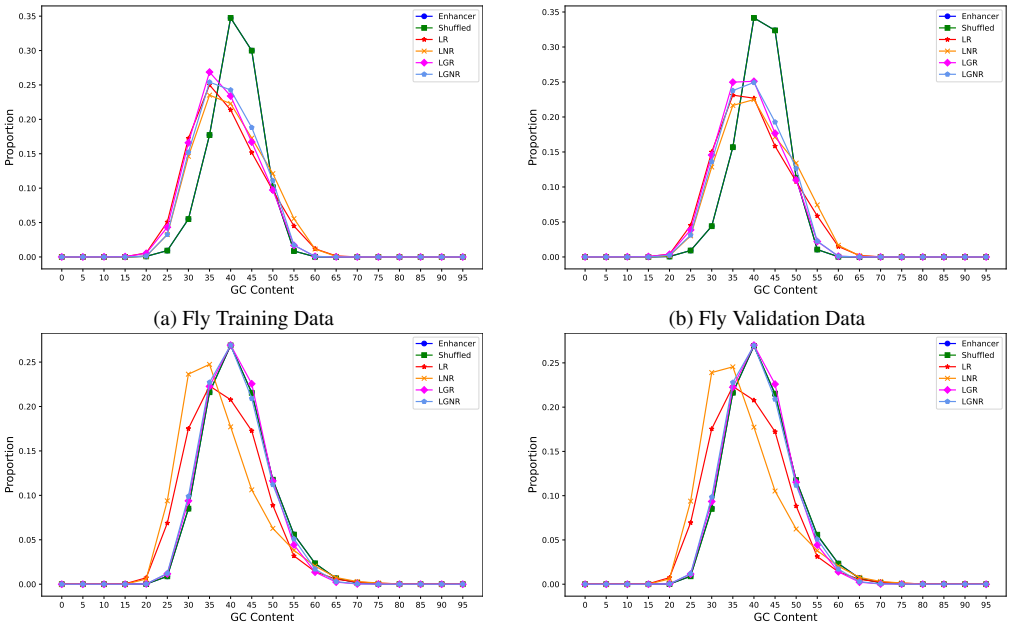
To ensure the high quality of our enhancer datasets, we implemented a stringent preprocessing pipeline to exclude potentially confounding genomic features, namely promoters, coding regions (exons), and insulators, from both the human and fly datasets. For the human data, we leveraged the RefSeq and GENCODE annotations to precisely delineate transcription start sites and the genomic coordinates of coding exons. Similarly, for the fly data, we employed the RefSeq and Ensembl databases to obtain analogous information. We defined a promoter region as a 1000 bp region centered on each annotated transcription start site. Human insulator regions were retrieved from the ENCODE Project via the UCSC Genome Table Browser, specifically selecting regions characterized by the presence of CTCF binding sites. For the fly, we incorporated experimentally validated insulator coordinates derived from a previously published study (Négre *et al.*, 2010). All instances of overlap between these potentially confounding regions and our initial enhancer datasets were systematically removed using BEDTools (Quinlan and Hall, 2010). Finally, to ensure uniformity in genomic coordinates for the fly data, all sequences were mapped to the DM6 genome assembly using the Liftover tool (Hinrichs *et al.*, 2006).

### Control Datasets

To rigorously evaluate our model's ability to discriminate true enhancers from non-functional genomic regions, we generated five distinct control datasets representing negative examples. Four of these datasets were specifically designed to control for potential confounding factors: sequence length, repetitive element content, and guanine-cytosine (GC) content. Based on these variables, we constructed the following datasets:

- Length Repeat (LR): Sequences in this control set were sampled to match the length distribution of the enhancer dataset, allowing for the presence of repetitive elements.
- Length No-Repeat (LNR): This dataset comprises sequences sampled to match the length distribution of the enhancer dataset, with the exclusion of repetitive elements.
- Length GC-Repeat (LGR): Sequences in this control set were sampled to match both the length and GC content distributions of the enhancer dataset, while permitting the inclusion of repetitive elements.
- Length GC-No-Repeat (LGNR): This dataset contains sequences sampled to match both the length and GC content distributions of the enhancer dataset, with the exclusion of repetitive elements.

In addition to these controlled datasets, we generated a fifth control set consisting of shuffled enhancer sequences.




(a) Fly Training Data (b) Fly Validation Data

(c) Human Training Data (d) Human Validation Data

**Figure 2.** Comparison of GC content in fly and human enhancer training and validation datasets. This figure illustrates the similarity in GC content distributions between control and enhancer datasets for both training and validation splits in human and fly. All four panels show closely aligned GC content distributions, demonstrating that the control sequences are well-matched to the enhancers in terms of GC content.

For each enhancer in our primary datasets, we generated shuffled sequences by randomly permuting its constituent k-mers (subsequences of length $k$), where $k$ ranged from 1 to 6. This control aims to assess the model's capacity to distinguish true enhancers from sequences that merely share the same k-mer composition. Repetitive elements for both the human and *Drosophila melanogaster* genomes were identified using Red (Girgis, 2015). Control regions for both species were selected from their respective genomes, ensuring no overlap with the enhancer datasets. When generating control sequences for the human dataset, the length was precisely matched to 400 bp. For the *Drosophila melanogaster* dataset, control sequence lengths were randomly chosen from a uniform distribution between 460 and 500 bp for each corresponding enhancer, providing a broader range for scanning. A tolerance of 0.03 was permitted for the GC content of the selected control sequences in both the human and *Drosophila melanogaster* datasets.

## Data Partitioning

Our enhancer and control datasets were split into training (70%), validation (20%), and testing (10%) datasets. Given the requirement for deep neural networks to be effectively trained with substantial amounts of data, we allocated 70% of our dataset for training purposes. An additional 20% of the data was designated as the validation dataset to detect and mitigate overfitting — the phenomenon where a model learns the training data too well but fails to generalize to new, unseen data (i.e., memorizing instead of learning). The remaining 10% of the data served as the testing dataset to ensure a comprehensive evaluation on previously unseen data. The data from both enhancers and controls in the human dataset were randomly allocated across these partitions. This random split ensures that each partition contains a representative mix of both enhancers and non-enhancers. However, for the fly dataset, a different approach was required. Since the enhancers for the fly were sampled from four different experiments there is potential overlap between the enhancers. If enhancers were split randomly, overlapping enhancers could appear across two or three partitions, leading to data leakage. As a remedy, sequences were sorted by chromosome and start position. After that, we split the data in sequential order.

## GC Distrubution

After partitioning the data, we ensured that the enhancer sequences and the control sequences were matched in terms of length and GC content. This prevents the model from oversimplifying the classification task by relying on differences in these basic sequence properties. As shown in Figure 2, the GC content distributions for both the fly and human datasets are well-matched between enhancers and controls. To quantify the similarity between the enhancer and control datasets, we used Kullback-Leibler (KL) divergence to compare the distributions of sequence length and GC content. A KL divergence of zero indicates identical distributions, with lower values reflecting greater similarity. For the fly datasets, we observed KL divergence values ranging from 0.1173 to 0.2216 for training and from 0.1272 to 0.2267 for validation. For the human datasets, values ranged from 0.0068 to 0.2233 for training and from 0.0066 to 0.2236 for validation, indicating strong similarity in both species. Shuffled control datasets yielded KL divergence values of 0.0 as they were generated directly from the enhancer sequences.

## Data Indexing

To enable the neural network to learn a representation of each nucleotide, we assigned a unique integer index to

each nucleotide. The specific values are unimportant to the network, as long as they are positive and unique. We modified the input sequences by assigning an integer to each nucleotide encountered in the genomic data. For example, the nucleotide A is assigned the index 2, with 0 and 1 reserved for padding and unknown bases, respectively. T is assigned the index 3, and so on, until every nucleotide has been mapped to a numerical value. Thus, a sequence such as ATAT is encoded as [2, 3, 2, 3], which can be used as input to the network.

### Data Augmentation

To train the network effectively, we increased the amount of training data through a data augmentation strategy. We generated additional data by creating reverse complement counterparts of the original sequences and using them as input. This approach is valid because forward and reverse complement sequences are biologically equivalent. As a result, the model was exposed to more diverse training examples without requiring additional data collection. During training, each input sequence had a 50% chance of being replaced with its reverse complement. These reverse complements were indexed into numerical representations using the same scheme as the forward sequences. We applied an additional augmentation technique when preparing the fly dataset. To ensure that all sequences fell within a uniform length range (460–500 bp), we expanded or slightly trimmed sequences as needed. This was important as we planned to scan long sequences by dividing them into segments within this range. The procedure was the same for both expansion and trimming: we determined the midpoint of each sequence, then generated five sequence copies of randomly sampled lengths between 460 and 500 bp, all centered on the midpoint.

### Convolutional Neural Network

For *EnhancerDetector*, the primary model used is a convolutional neural network. Convolutional neural networks are well-suited for processing data with a structured arrangement, such as DNA sequences, and are effective at identifying and learning hierarchical patterns. Our network (Figure 3) begins with an embedding layer, which learns numerical representations for nucleotides. The output of the embedding layer is passed to four custom convolutional blocks, each consisting of two one-dimensional convolutional layers followed by a batch normalization layer. The output of the final block is flattened and passed through two dense layers, with a batch normalization layer in between. The network ends with a single-neuron dense layer that outputs a probability score between 0 and 1, representing the likelihood that a given sequence is an enhancer. Sequences with scores greater than or equal to 0.5 are classified as enhancers, while those below 0.5 are classified as non-enhancers. We optimized the following hyperparameters:

- Number of filters: Determines how many patterns each convolutional layer learns.
- Filter size: Specifies the width of the filters, e.g., a size of 3 corresponds to patterns spanning three consecutive positions.
- Number of dense neurons: Controls the number of neurons in the dense layers used for classification.

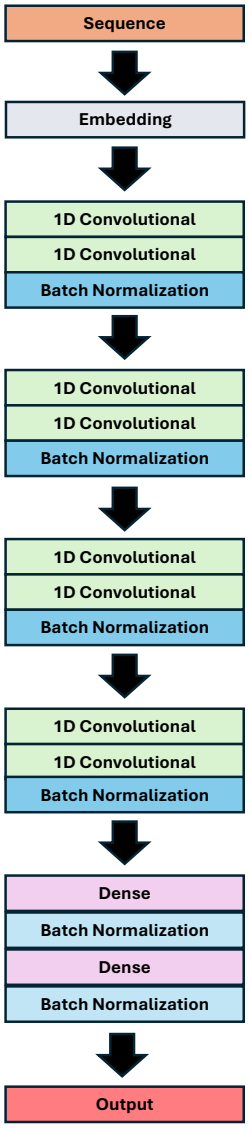



**Figure 3.** An overview of the *EnhancerDetector* architecture. The network takes a DNA sequence as input and passes it through an embedding layer, which learns a numerical representation for each nucleotide. The resulting sequence is then processed by four convolutional blocks, each consisting of two 1D convolutional layers followed by a batch normalization layer. These blocks extract patterns within the input sequences. Next, the output is passed through two dense layers, with a batch normalization layer in between, to classify whether a given sequence is an enhancer. The final output is a value between 0 and 1, where values above 0.5 indicate an enhancer.

### Recurrent Neural Network

Another model we considered was a recurrent neural network, which is particularly well-suited for sequential data, as it can capture temporal dependencies and patterns over time. This model uses a recurrent block consisting of a Gated Recurrent Unit (GRU) layer, followed by a batch normalization layer and an activation function. The architecture is similar to that of the convolutional neural network: it begins with an embedding layer, followed by two recurrent blocks, then two dense layers with a batch normalization layer in between. For this network, we optimized the following three hyperparameters:

the number of GRU neurons, the activation function used in the recurrent block, and the number of dense neurons.

### Evaluation Metrics

We evaluated the performance of *EnhancerDetector* and related tools using the following metrics (higher values indicate better performance across all metrics):

- Accuracy: Percentage of correctly predicted samples (enhancers or non-enhancers).
- Recall (Sensitivity): Percentage of true enhancers correctly identified.
- Specificity: Percentage of true non-enhancers correctly identified.
- Precision: Proportion of predicted enhancers that are correct. This is especially important for experimental validation in transgenic flies.
- F1 and Weighted F1 Scores: The F1 score is the harmonic mean of precision and recall. The weighted F1 score also incorporates precision and recall, but precision was weighted twice as much.

### Guarding Against Overfitting

In addition to the data augmentation techniques described earlier, we applied early stopping to prevent overfitting — a situation where the model performs well on training data but poorly on unseen data. Early stopping is a regularization method that halts training when performance on validation data stops improving. Specifically, we monitored the weighted F1 score on the validation set after each epoch (one complete pass through the entire training dataset). Training continued as long as the score improved. If no improvement was observed for 25 epochs, training was stopped and the model with the highest validation weighted F1 score was selected.

### Optimization

The models were evaluated using various combinations of hyperparameters to identify the optimal configuration. The following hyperparameter values were tested: number of filters (32, 40, 48, 64, and 72), filter size (3 and 5), and number of dense neurons (10, 20, and 40). The best configuration for the human enhancer classifier was a combination of 64 filters, a filter size of 3, and 20 dense neurons. Similarly, we identified two best networks for the fly enhancer classifier, which were optimized with combinations of 32 and 40 filters, a filter size of 3, and 20 dense neurons. The best models were primarily chosen based on a high weighted F1 score, due to the imbalance between confirmed enhancers and control data. Specificity was also considered to ensure the model does not incorrectly identify non-enhancers, and recall was assessed to verify the model's ability to correctly identify enhancers.

### Fine-Tuning

We observed that the human model was trained on a comparatively larger dataset than the fly model (8,954,140 vs. 6,473,088), allowing it to learn certain characteristics and patterns more effectively. We therefore decided to adapt our trained human classifier to the fly dataset using a fine-tuning approach. The rationale was to adjust the model to optimize its performance on a new but related dataset. To fine-tune the classifier on the fly data, we employed a strategy of freezing and unfreezing layers during training. Only the unfrozen layers were updated with the new data, enabling the model to adapt while preserving knowledge gained from the human data. This approach allows us to retain useful features learned from the human dataset while tailoring the model to the specific characteristics of the fly data.

### Ensemble

After training the fly networks and fine-tuning the model, we created an ensemble of these three networks. An ensemble combines multiple neural network models, each independently tasked with determining whether a given genomic region is an enhancer. The ensemble aggregates the predictions from all three models: if all agree that a region is an enhancer, it is classified as such; otherwise, it is not. This approach provides a stricter criterion for identifying enhancer regions, reducing the likelihood of false positives.

### Comparison to Related Tools

To further validate *EnhancerDetector*, we compared its performance with that of other enhancer detection tools designed for binary classification. One such tool is Enhancer-FRL (Wang *et al.*, 2023), which predicts human enhancers of 200 bp in length using feature representation learning and multi-view sequence information. We evaluated Enhancer-FRL using two datasets: the ENCODE Human dataset (obtained from the UCSC Genome Table Browser) and the ENCODE Mouse dataset. Both datasets were preprocessed as described earlier, including the removal of promoters, exons, and insulators. For the Enhancer-FRL evaluation, we selected potential enhancers that were 200 bp or shorter. Regions shorter than 200 bp were symmetrically extended in both directions to reach exactly 200 bp in length. This same procedure was applied to mouse enhancers. Because our human network expects input sequences of 400 bp, we further extended the 200-bp enhancers to 400 bp. To ensure a fair comparison, we excluded any enhancer region that overlapped by at least 50% with our CATlas training enhancer dataset. This filtering ensured that none of the evaluated enhancers had been seen during training. For both the 200-bp and 400-bp enhancers in human and mouse datasets, we also generated matched control sequences (shuffled, LR, LNR, LGR, and LGNR). Enhancer-FRL was run using its default parameters.

Another tool we compared *EnhancerDetector* to was Enhancer-LSTMAtt (Huang *et al.*, 2022). This method detects 200-bp enhancers using a deep neural network that incorporates bidirectional Long Short-Term Memory (LSTM) layers and an attention mechanism. Enhancer-LSTMAtt was evaluated using the same 200-bp human and mouse ENCODE datasets, along with their corresponding control datasets. The model was executed using its default parameters.

A third tool we evaluated was the PDCNN model (Hu *et al.*, 2024). PDCNN is a convolutional neural network specifically designed for DNA sequence analysis. Unlike the other tools, PDCNN was trained on both human and mouse enhancers. The authors used sequences ranging from 50 bp to 300 bp for testing and recommended using 200-bp sequences for optimal performance. Accordingly, we evaluated PDCNN using the

same 200-bp human and mouse enhancer datasets, along with their corresponding control datasets. PDCNN was run using its default parameters.

The last tool we assessed was gkm-SVM (Ghandi *et al.*, 2014). This sequence-based classifier uses gapped k-mer features and support vector machines to predict functional regulatory elements in genomic sequences. By allowing gaps within k-mers, gkm-SVM captures more flexible and informative sequence patterns compared to traditional k-mer methods. More specifically, we used LS-GKM (Lee, 2016), an optimized version of gkm-SVM that supports training on large-scale datasets (up to 100,000 sequences). Unlike the other tools we tested, LS-GKM uses a support vector machine–based approach instead of neural networks to detect enhancers. Additionally, this tool can be trained on a user-provided dataset, unlike the other three tools, which are distributed with pre-trained models. To train it, we randomly selected 50,000 positive sequences from the CATlas training dataset. For negative sequences, we randomly selected 10,000 sequences from each of the shuffled, LR, LNR, LGR, and LGNR training datasets, resulting in a total of 50,000 negative sequences. These 100,000 sequences were then used to train LS-GKM using its default parameters. We subsequently evaluated its performance on our testing dataset and compared the results to those of our human network. This comparison was fair, as both *EnhancerDetector* and LS-GKM were trained on the same dataset and evaluated on the same unseen testing sequences. The LS-GKM model trained on CATlas sequences was then evaluated on the 400-bp-long human and mouse ENCODE enhancers.

### Reporter Gene Analysis

Putative enhancer sequences were synthesized as gBlocks (IDT, Coralville, IA, USA) and cloned into the plasmid pattBnucGFPs (details available upon request). Transgenic flies were generated by insertion into the third chromosome attP2 site (Rainbow Transgenics, Camarillo, CA, USA). For embryo analysis, homozygous transgenic embryos were collected, fixed, and stained using standard protocols. Larval expression was assessed by direct visualization of GFP following larval dissection.

## RESULTS AND DISCUSSIONS

### Main Contributions

This study presents several key contributions that advance the state of enhancer identification:

- We introduce *EnhancerDetector*, a deep learning framework for accurate and interpretable enhancer prediction *across species*. The code is freely available for academic use through github.com/BioinformaticsToolsmith/EnhancerDetector and doi.org/10.5281/zenodo.15531293.
- The model generalizes well to enhancers identified using diverse experimental assays, including CAGE, snATAC-seq, and DNase-seq.
- It is easily fine-tuned to new species with limited data, achieving high performance with as few as 20,000 enhancer sequences.
- Experimental validation in transgenic flies confirms the predictive accuracy of *EnhancerDetector*, with five of six tested enhancers activating reporter expression.
- We apply class activation mapping to provide mechanistic insights with *in silico* perturbation experiments.
- *EnhancerDetector* consistently outperforms existing tools (Enhancer-FRL, Enhancer-LSTMAtt, PDCNN, and LS-GKM) in precision and F1 score on both human and mouse datasets.

### *EnhancerDetector* Can Discover Human Enhancers

To assess our model's ability to recognize human enhancers, we trained a convolutional neural network on the CATlas human tissue dataset, which includes 400-bp sequences identified as putative enhancers using snATAC-seq across 222 cell types. We evaluated several network architectures by varying the number of convolutional filters, filter sizes, and dense layer neurons. The best-performing model consisted of 64 filters in the first convolutional layer (doubled in subsequent layers) with a filter size of 3 and 20 dense neurons. As shown in Table 1, our network achieves a validation F1 score of 72%, indicating a good balance between precision and recall. The validation recall score is 66%, demonstrating that our model is able to identify about two-thirds of human enhancers. The validation precision score is 75%, meaning that for every 100 positive predictions, 75 are true positives. Our human network has high specificity — 89% — indicating that it can accurately identify non-enhancers about 90 out of 100 times.

To test the impact of control sequence characteristics, we evaluated the model against five types of negative control datasets that varied in GC content and sequence repetitiveness, along with fully shuffled enhancer sequences. When sequences were shuffled, our network achieved a high precision and specificity of 99% and 97%, respectively. These high scores show that our network can reliably distinguish actual enhancers from random sequences with the same k-mer compositions. We observed that the network performs best when using control sequences that do not match GC content, achieving an F1 score of 71% for both the LR and LNR datasets, with LNR having a slight advantage in precision and specificity scores of 74% and 88%, respectively, compared to LR. Conversely, the model struggles more when non-enhancer sequences match GC content but do not include repeated elements, resulting in an F1 score of 68% and the lowest precision and specificity scores of 69% and 85%, respectively. Collectively, these results show *EnhancerDetector* effectively learns sequence patterns characteristic of human enhancers and can predict them with high precision and specificity.

### *EnhancerDetector* Can Learn Patterns in Fly Enhancers

To evaluate *EnhancerDetector*'s ability to identify enhancers in *Drosophila melanogaster*, we built multiple convolutional networks with varying numbers of filters, filter sizes, and dense layer neurons. We selected the two models that gave the best results (Table 1). These networks were trained using a combination of fly enhancer datasets: distal DNase I hypersensitive sites, distal scATAC-seq peaks, and cis-regulatory modules from REDFly. As with the human

**Table 1.** Training and validation performance of different neural networks on human and fly enhancers. The models include human-trained convolutional networks, two fly-trained convolutional networks, a fine-tuned human convolutional network on fly data, an ensemble combining ab initio and fine-tuned models for discovering fly enhancers, and a recurrent model specifically designed for fly enhancer sequences. Performance was also evaluated on five control datasets: four sets consisting of genomic sequences sampled to match the enhancer datasets in length and/or GC content, with or without repetitive elements (LR: Length Repeat; LNR: Length No Repeat; LGR: Length GC Repeat; LGNR: Length GC No Repeat), and a fifth set consisting of shuffled enhancer sequences.

| | Training | | | | | Validation | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **Accuracy** (%) | **Recall** (%) | **Specificity** (%) | **Precision** (%) | **F1 Score** (%) | **Accuracy** (%) | **Recall** (%) | **Specificity** (%) | **Precision** (%) | **F1 Score** (%) |
| *Human from Scratch* | | | | | | | | | | |
| Overall | 82.83 | 69.62 | 89.44 | 76.72 | 74.18 | 81.53 | 66.34 | 89.12 | 75.30 | 72.04 |
| Shuffled | 89.24 | 69.62 | 99.05 | 97.34 | 85.91 | 88.12 | 66.38 | 98.99 | 97.05 | 84.07 |
| LR | 81.92 | 69.59 | 88.09 | 74.50 | 72.76 | 80.59 | 66.34 | 87.72 | 72.98 | 70.59 |
| LNR | 82.23 | 69.60 | 88.55 | 75.25 | 73.24 | 80.99 | 66.41 | 88.28 | 73.91 | 71.20 |
| LGR | 80.64 | 69.63 | 86.15 | 71.53 | 70.86 | 79.29 | 66.40 | 85.74 | 69.96 | 68.70 |
| LGNR | 80.09 | 69.60 | 85.33 | 70.35 | 70.07 | 78.71 | 66.34 | 84.89 | 68.70 | 67.87 |
| *Fly from Scratch (32 filters of size 3 and 20 dense units)* | | | | | | | | | | |
| Overall | 85.96 | 66.72 | 92.38 | 74.47 | 71.67 | 83.82 | 68.45 | 88.94 | 67.35 | 67.69 |
| Shuffled | 90.63 | 66.75 | 98.59 | 94.05 | 82.73 | 90.88 | 68.49 | 98.35 | 93.25 | 83.19 |
| LR | 84.25 | 66.69 | 90.10 | 69.18 | 68.29 | 80.89 | 68.51 | 85.02 | 60.38 | 62.83 |
| LNR | 85.85 | 66.73 | 92.22 | 74.09 | 71.42 | 84.04 | 68.38 | 89.27 | 67.98 | 68.10 |
| LGR | 83.59 | 66.70 | 89.23 | 67.35 | 67.09 | 79.97 | 68.46 | 83.81 | 58.49 | 61.44 |
| LGNR | 85.47 | 66.70 | 91.73 | 72.90 | 70.67 | 83.28 | 68.47 | 88.23 | 65.96 | 66.74 |
| *Fly from Scratch (40 filters of size 3 and 20 dense units)* | | | | | | | | | | |
| Overall | 84.92 | 59.21 | 93.49 | 75.21 | 68.96 | 83.94 | 62.92 | 90.94 | 69.84 | 67.34 |
| Shuffled | 89.44 | 59.22 | 99.52 | 97.61 | 80.21 | 90.26 | 62.91 | 99.37 | 97.09 | 82.16 |
| LR | 83.25 | 59.20 | 91.26 | 69.31 | 65.52 | 81.27 | 62.96 | 87.38 | 62.42 | 62.57 |
| LNR | 84.80 | 59.23 | 93.32 | 74.73 | 68.68 | 84.21 | 62.91 | 91.31 | 70.70 | 67.86 |
| LGR | 82.68 | 59.21 | 90.50 | 67.52 | 64.45 | 80.48 | 62.99 | 86.32 | 60.53 | 61.30 |
| LGNR | 84.48 | 59.28 | 92.89 | 73.53 | 68.02 | 83.54 | 62.96 | 90.40 | 68.60 | 66.59 |
| *Human Fine-tuned on Fly* | | | | | | | | | | |
| Overall | 87.68 | 73.24 | 92.49 | 76.47 | 75.13 | 83.27 | 69.36 | 87.90 | 65.64 | 66.60 |
| Shuffled | 92.14 | 73.24 | 98.44 | 93.99 | 85.85 | 90.92 | 69.34 | 98.11 | 92.45 | 83.15 |
| LR | 86.04 | 73.25 | 90.31 | 71.58 | 72.10 | 80.14 | 69.40 | 83.73 | 58.70 | 61.83 |
| LNR | 87.61 | 73.26 | 92.39 | 76.24 | 75.18 | 83.49 | 69.47 | 88.16 | 66.18 | 67.20 |
| LGR | 85.39 | 73.25 | 89.43 | 69.80 | 70.87 | 79.14 | 69.40 | 82.39 | 56.77 | 60.38 |
| LGNR | 87.25 | 73.29 | 91.90 | 75.10 | 74.45 | 82.64 | 69.42 | 87.05 | 64.11 | 65.75 |
| *Ensemble of ab initio and Fine-tuned Fly* | | | | | | | | | | |
| Overall | 85.39 | 52.14 | 96.47 | 83.13 | 64.09 | 83.85 | 52.97 | 94.15 | 75.10 | 62.13 |
| Shuffled | 87.95 | 52.14 | 99.89 | 99.35 | 68.39 | 88.12 | 52.97 | 99.84 | 99.08 | 69.04 |
| LR | 84.33 | 52.14 | 95.06 | 77.88 | 62.46 | 81.95 | 52.97 | 91.62 | 67.80 | 59.48 |
| LNR | 85.43 | 52.14 | 96.52 | 83.32 | 64.14 | 84.11 | 52.97 | 94.48 | 76.20 | 62.50 |
| LGR | 84.00 | 52.14 | 94.62 | 76.36 | 61.97 | 81.40 | 52.97 | 90.87 | 65.92 | 58.74 |
| LGNR | 85.24 | 52.14 | 96.27 | 82.33 | 63.85 | 83.68 | 52.97 | 93.92 | 74.39 | 61.88 |
| *Recurrent Model for Fly Enhancers* | | | | | | | | | | |
| Overall | 84.15 | 61.92 | 91.56 | 70.96 | 67.64 | 83.62 | 67.72 | 88.92 | 67.08 | 67.26 |
| Shuffled | 89.82 | 61.90 | 99.13 | 95.95 | 81.04 | 91.14 | 67.73 | 98.94 | 95.53 | 84.00 |
| LR | 82.27 | 61.90 | 89.04 | 65.33 | 64.11 | 80.71 | 67.70 | 85.04 | 60.14 | 62.41 |
| LNR | 83.76 | 61.89 | 91.05 | 69.74 | 66.87 | 83.71 | 67.66 | 89.05 | 67.33 | 67.38 |
| LGR | 81.52 | 61.93 | 88.04 | 63.32 | 62.81 | 79.79 | 67.77 | 83.79 | 58.23 | 61.06 |
| LGNR | 83.37 | 61.93 | 90.51 | 68.51 | 66.12 | 82.82 | 67.72 | 87.86 | 65.02 | 65.87 |

experiments, we used five negative datasets. Overall, both models had 67–68% validation F1 scores. Their validation recall scores ranged 63–68%, and specificity scores ranged 89–91%. Their precision scores were 67–70%. The scores on the training data were slightly higher than those obtained on the validation dataset but still comparable, demonstrating the lack of overfitting. The two networks performed well on the shuffled dataset. Both models showed strong specificity and precision when the control sequences did not contain repeats. Controlling for GC content made the classification task slightly more difficult: the F1 scores of the two networks on the LNR dataset (control for length only, excluding repeats) were both 68%, whereas the scores on the LGNR (length and GC-controlled, no repeats) dataset were both 67%. The same trend — that controlling for GC content makes the task slightly more difficult — was observed on the datasets that include repeats. Compared to the performance of the human network, the trend related to GC content is consistent, but the trend related to the inclusion or exclusion of repeats is not. These results reaffirm from a previous study (Li *et al.*, 2007) that there is a discernible pattern within the enhancer sequences and enhancers have a characteristic GC content.

## Fine-Tuning the Human Network Achieved Comparable Performance to Fly Networks Trained Ab Initio

We investigated whether a model trained on human enhancers could be adapted to discover fly enhancers through fine-tuning. Fine-tuning is a technique where a model that was already trained for one task is slightly adjusted to perform well on a new, related task, allowing it to keep what it has already learned while making small updates to adapt to the new data. This approach saves time and often leads to better performance, especially when the new task has less training data. Because the human training dataset was substantially larger than the fly dataset (8.9 million vs. 6.4 million samples), we hypothesized that the human-trained model had captured generalizable sequence features. To test this, we applied a fine-tuning strategy in which certain layers were frozen —

their parameters remain fixed during training — while others were retrained on the fly dataset. This approach allowed the model to retain previously learned representations while adapting to fly-specific enhancer sequence patterns. As shown in Table 1, the fine-tuned network performed comparably to the two models trained from scratch on the fly dataset. The overall validation accuracy was 83%, compared to 84% for both fly-specific models. The fine-tuned network achieved a recall of 69%, slightly outperforming the two *ab initio* models (68% and 63%), while maintaining strong specificity (88% compared to 89% and 91%). Precision (66%) and F1 score (67%) were also closely aligned with those of the fly-trained models. These results demonstrate that fine-tuning a human-trained model can be as effective as training a model *ab initio*, offering a practical and efficient strategy for adapting *EnhancerDetector* to new species without the need for extensive retraining or large datasets.

### A Network Ensemble Improves Prediction Confidence

After training three fly models — two trained from scratch on the fly dataset and one fine-tuned from the human-trained network — we observed that their individual performances were comparable, yet slightly different. To improve prediction confidence, we combined these models into an ensemble. In this approach, each model independently evaluates a sequence, and the sequence is classified as an enhancer only if all three models agree. This voting-based strategy prioritizes precision and specificity over recall by requiring unanimous consensus across models when it comes to positive predictions. As shown in Table 1, the ensemble achieved a specificity score of 94% and a precision score of 75%, outperforming the individual models (specificity scores of 89%, 91%, and 88%; precision scores of 67%, 70%, and 66%). Although recall dropped to 53%, indicating that the ensemble identified fewer true enhancers, it substantially reduced the number of false positives. These results demonstrate that the ensemble method provides a valuable option when higher confidence in predicted enhancers is needed, such as for experimental validation using transgenic animals.

### The Ensemble Can Delineate Enhancers in REDfly

We selected 18 long sequences (median length 4 kb, range 3.2–7.1 kb) from the REDfly database. Each of these sequences has demonstrated regulatory activity and thus contains one or more individual enhancers. Our goal was to determine the locations of the enhancers within these long sequences. We cleaned the sequences by removing promoters, insulators, and exons. We then split each long sequence into segments ranging from 460 to 500 base pairs and fed each segment into our ensemble of networks. Based on the confidence scores generated by the networks, we determined the locations of the highest-scoring segment(s) within each long sequence. We also shuffled the long sequences and ran them through the ensemble to ensure that our network could distinguish between real enhancer sequences and shuffled sequences. To test whether the network ensemble accurately predicted discrete enhancer locations within the longer sequences, we selected six predicted enhancers for *in vivo* testing using reporter gene analysis in transgenic *Drosophila*. Results are shown in Figure 4. Three of the putative

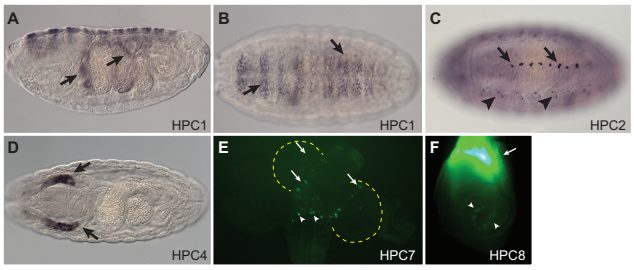



**Figure 4.** *In vivo* validation of predicted enhancers. *Drosophila* enhancers of 460–500 bp in length were predicted from longer sequences (about 4,000 bp) with known regulatory activity and tested *in vivo* in a transgenic reporter gene assay. (A) HPC1 drives expression in the embryonic midgut (arrows) as well as (B) the dorsal ectoderm (arrows), consistent with described expression for the enclosing ush_-12475/-6925 enhancer (Muratoglu *et al.*, 2006). (C) HPC2 drives reporter gene expression in the ventral midline (arrows), consistent with known expression of the *ush* gene but not the ush_-12475/-6925 enhancer. Arrowheads indicate expression driven by the plasmid vector, not the HPC2 enhancer. (D) HPC4 shows reporter gene expression in the embryonic salivary glands (arrows). Neither the larger enclosing enhancer (GMR26G07) nor its assigned target gene (*chn*) are reported to be expressed in this tissue. (E) HPC7 drives reporter expression in the larval brain (arrows; brain is outlined with yellow dotted lines) and ventral nerve cord (arrowheads). This resembles the described expression of the enclosing GMR52B12 enhancer. (F) HPC8 shows reporter gene expression in imaginal discs, including the leg discs (pictured). Very strong expression is observed in the presumptive proximal leg tissue (arrow), with weaker expression more distally (arrowheads), consistent with described expression of the *hth* target gene.

enhancers (HPC1–HPC3) were drawn from the same initial long sequence, while three others were taken from distinct long sequences (HPC4, HPC7, and HPC8). HPC1 drives weak reporter gene expression in the embryonic midgut and dorsal ectoderm (Figure 4 A, B), consistent with the activity of its enclosing longer sequence, enhancer ush_-12475/-6925 (REDflyID: RFRC:0000003436.004; Muratoglu *et al.*, 2006). Interestingly, as reported in the original study (Muratoglu *et al.*, 2006), this expression appears to be ectopic relative to the endogenous expression of the assigned target gene, *ush*. HPC2 drives reporter gene expression in the midline of the embryonic ventral nervous system (Figure 4 C). This expression is not observed with the original longer enhancer. However, it is consistent with the known expression pattern of the *ush* gene, although a precise one-to-one correspondence between expressing cells remains to be determined (Fossett *et al.*, 2000). Whether the discrepancy between the two reporter constructs results from a mis-characterization of the original ush_-12475/-6925 reporter, is the result of a latent activity of the HPC2 fragment that is masked either *in vivo* or in the removed-from-context longer fragment, or is due to some other reason similarly remains to be explored. HPC3 failed to show reporter activity (data not shown). HPC4 exhibited strong reporter activity in the embryonic salivary gland (Figure 4 D). However, this is not a tissue where activity was observed with the original longer sequence (GMR26G07; REDflyID: RFRC:0000024999.002; Jenett *et al.*, 2012). Moreover, ectopic expression in this tissue is not uncommon due to sequences in the reporter gene vector (Zhu and Halfon, 2007). Thus, it is questionable whether this sequence should be considered a *bona fide* enhancer. HPC7 did not show significant embryonic activity but did drive limited gene expression in the larval brain and ventral nerve cord (Figure 4 E). This is consistent with the described

activity of the longer enclosing sequence (GMR52B12; REDflyID: RFRC:0000027000; Jenett *et al.*, 2012), although, again, an exact one-to-one correspondence has not been determined. HPC8 also lacked appreciable embryonic activity but was active in imaginal discs (Figure 4 F), consistent with part of the described activity of its longer enclosing sequence (GMR46A08; REDflyID: RFRC:0000030377; Jenett *et al.*, 2012). In total, five of the six tested putative enhancers (83%) drove reporter gene activity, and four (67%) exhibited expression patterns consistent with those of their larger enclosing sequences and/or their likely target genes. These results strongly demonstrate that *EnhancerDetector* can discover transcriptional enhancers.

### Recurrent and Convolutional Networks Have Comparable Performance

To evaluate alternative architectures for enhancer prediction, we trained a Recurrent Neural Network (RNN) from scratch on the fly dataset and compared its performance to that of our three Convolutional Neural Networks (CNNs): two trained directly on fly data and one fine-tuned from the human-trained model. The RNN architecture employed gated recurrent units and followed a structure similar to the CNNs, including an embedding layer, two recurrent blocks, and two dense layers. As shown in Table 1, the RNN achieved validation performance comparable to that of the CNNs. Overall, the validation accuracy scores for the four networks were approximately the same (84% vs. 84%, 84%, and 83%). Recall scores were also similar (68% vs. 68%, 63%, and 69%), as were specificity (89% vs. 89%, 91%, and 88%) and precision (67% vs. 67%, 70%, and 66%). Importantly, the F1 scores — the harmonic mean of recall and precision — were nearly identical across all models (67% vs. 68%, 67%, and 67%). These results demonstrate that the RNN achieves performance comparable to the CNNs, indicating that both architectures are suitable for recognizing transcriptional enhancers. However, training the RNN was time-consuming. Each epoch required approximately 40 minutes, and training the network on the fly data took around 65 epochs, resulting in a total training time of about two days. In contrast, each epoch of the CNN took about 8 minutes, and training typically required 22 epochs on the fly data — resulting in a total training time of approximately three hours. Thus, the CNNs trained markedly faster while achieving similar performance. When we attempted to train an RNN on the human dataset, each epoch took nearly four hours, rendering the training process impractically slow. For this reason, we did not pursue RNN training on the human data. In summary, our decision to prioritize CNNs over RNNs was clearly justified: they achieved comparable performance while drastically reducing the required training time.

### *EnhancerDetector* Detects Putative Enhancers Identified by Alternative Biotechnologies

To evaluate our human network, we used an alternative set of potential enhancers identified using different methodologies. The FANTOM5 Project identifies putative enhancers based on cap analysis of gene expression, a method distinct from the snATAC-seq technology used to identify CATlas enhancers. We applied the same preprocessing pipeline to the FANTOM dataset as we did for the fly and human datasets, including the removal of promoters, exons, and insulators. Next, we extracted regions that are 400 bp or shorter; any region shorter than 400 bp was symmetrically expanded to 400 bp. This step was necessary because the human network — trained on CATlas enhancers — expects input sequences of 400 bp. After preprocessing, the dataset contained 41,553 potential enhancers. We first performed a baseline evaluation using our human network to assess its general ability to detect enhancers. To ensure a fair assessment, we then divided the FANTOM5 regions into four datasets: training, validation, testing, and non-overlapping. A region was assigned to the training, validation, or testing set if it overlapped a CATlas region in the corresponding set by at least 50%. The non-overlapping set included FANTOM regions that did not meet this overlap threshold with any CATlas training, validation, or testing region. This partition procedure resulted in 17,306, 4,896, 2,483, and 16,868 sequences in the respective datasets.

When evaluated on the full FANTOM5 dataset, the model achieved an accuracy of 73%, a recall of 50%, a specificity of 96%, and a precision of 93%. Although the recall was lower than in the CATlas-based evaluations, the high precision and specificity suggest that the model is stringent in its predictions, favoring precision over sensitivity and thereby reducing false positives. On the training, validation, and testing subsets of the FANTOM5 data, the network correctly identified 66–67% of the putative enhancers. *Although the FANTOM5 testing subset contains some regions overlapping with CATlas, it is important to emphasize that the network had no access to this data during training or validation; hence, results on this set represent a true blind test.* Detecting 67% of the putative enhancers in such a blind evaluation underscores our model's ability to generalize enhancer patterns across datasets.

On the non-overlapping subset — which contains sequences with no or minimal (<50%) overlap with CATlas — the network classified only 28% of the sequences as enhancers. While a 28% detection rate for the non-overlapping set may initially appear low, it is important to note that approximately 25–30% of sequences in the FANTOM5 dataset are not true enhancers. As reported by the FANTOM5 authors, "67.4–73.9% of the CAGE-defined enhancers showed significant reporter activity." In another *in vivo* validation using zebrafish embryos, they observed "tissue-specific enhancer activity with 3 of 5 fragments, which corresponded to the human enhancer tissue expression." Thus, the 28% recall rate on the non-overlapping set doesn't necessarily reflect poor model performance, but rather suggests that this subset predominantly comprises the false positives within the FANTOM5 dataset, a hypothesis strongly supported by their minimal or absent overlap with the high-confidence single-cell data from CATlas.

These results demonstrate that our network effectively identifies a substantial portion of enhancers detected via both CAGE and snATAC-seq, highlighting its generalizability across datasets produced using different biotechnologies.

### *EnhancerDetector* Works on Another Vertebrate Species

To test our network's ability to discover enhancers in other species, we assembled a dataset of putative enhancers from the mouse (*Mus musculus*) genome. This dataset allows us to evaluate whether *EnhancerDetector* can generalize to

**Table 2.** Results from evaluating the human-trained network (without and with fine-tuning) on the mouse dataset. In the top half of the table, training performance is omitted because the model was not trained on mouse enhancers — it was only evaluated on mouse data.

| | Training | | | | | Validation | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **Accuracy** (%) | **Recall** (%) | **Specificity** (%) | **Precision** (%) | **F1 Score** (%) | **Accuracy** (%) | **Recall** (%) | **Specificity** (%) | **Precision** (%) | **F1 Score** (%) |
| *Human network with no fine-tuning* | | | | | | | | | | |
| Overall | — | — | — | — | — | 79.07 | 66.42 | 85.40 | 69.46 | 68.37 |
| Shuffled | — | — | — | — | — | 86.51 | 66.82 | 96.36 | 90.17 | 80.72 |
| LR | — | — | — | — | — | 77.46 | 66.79 | 82.80 | 66.01 | 66.23 |
| LNR | — | — | — | — | — | 78.15 | 66.89 | 83.77 | 67.33 | 67.14 |
| LGR | — | — | — | — | — | 77.07 | 66.95 | 82.13 | 65.19 | 65.74 |
| LGNR | — | — | — | — | — | 77.21 | 67.00 | 82.31 | 65.45 | 65.92 |
| *Human network with fine-tuning* | | | | | | | | | | |
| Overall | 84.39 | 75.05 | 89.06 | 77.42 | 76.45 | 83.02 | 71.72 | 88.67 | 75.99 | 74.35 |
| Shuffled | 88.24 | 75.09 | 94.82 | 87.88 | 83.13 | 86.92 | 71.79 | 94.49 | 86.69 | 81.05 |
| LR | 84.27 | 75.09 | 88.85 | 77.11 | 76.40 | 82.90 | 71.82 | 88.44 | 75.65 | 74.30 |
| LNR | 84.27 | 75.05 | 88.88 | 77.15 | 76.42 | 83.05 | 71.81 | 88.68 | 76.02 | 74.54 |
| LGR | 83.04 | 75.06 | 87.02 | 74.31 | 74.54 | 81.69 | 71.98 | 86.55 | 72.78 | 72.49 |
| LGNR | 82.16 | 75.12 | 85.68 | 72.40 | 73.25 | 80.82 | 71.79 | 85.34 | 71.00 | 71.24 |

another mammalian species. The Mouse ENCODE Project identifies putative enhancers based on DNase I hypersensitive sites. We obtained the mm10 ENCODE dataset from the UCSC Genome Table Browser. As with the human and fly enhancer datasets, we applied the same preprocessing procedure to the ENCODE dataset by removing promoters, exons, and insulators (regions containing CTCF binding sites). Promoters and exons were obtained from RefSeq and GENCODE (VM25) via the UCSC Genome Table Browser. After preprocessing, we selected potential enhancers shorter than 400 bp. Short regions were expanded bidirectionally to reach 400 bp in length. The final dataset contained 218,207 potential mouse enhancers. We then created five control datasets (shuffled, LR, LNR, LGR, and LGNR) using the same approach as for the human and fly controls. As shown in Table 2, the human-trained network achieved accuracy of 79%, recall of 66%, specificity of 85%, precision of 69%, and an F1 score of 68%. Using our base human network without any adjustments, we successfully identified two thirds of enhancers with approximately 70% precision. The model correctly excluded about 89% of non-enhancer regions. These results demonstrate that our network effectively generalizes enhancer patterns not only across different enhancer-detection biotechnologies but also across different vertebrate species.

To assess whether the predicted true positive enhancer sequences are conserved, we aligned them to the human genome using BLAST (Supplementary File 1). We found that only 6% of these sequences showed conservation (having at least 70% sequence identity and 70% alignment coverage). This low conservation rate demonstrates that our model identifies enhancer regions based on regulatory patterns that extend beyond simple sequence similarity.

## Fine-Tuning Improves Our Mouse Evaluation

To further enhance our human network using the mouse dataset, we applied a fine-tuning technique with the potential mouse enhancers. We divided the enhancer and control datasets into training, validation, and testing sets using a 70%, 20%, and 10% split, respectively. Similar to our approach with the fly dataset, we froze all layers during training except for the dense layers, which are responsible for making classifications rather than extracting features. This technique allowed us to train a new network in which only the classifier was updated, ensuring that we retained features learned from the human enhancers while adapting to the mouse enhancers. As shown in Table 2, fine-tuning led to performance improvements across the board, compared to using the human-trained network without any adaptation. The fine-tuned network achieved an overall validation accuracy of 83%, compared to 79% for the original human network. It also showed higher recall (71% vs. 66%) and specificity (89% vs. 85%), along with improved precision and F1 scores (76% vs. 69% and 74% vs. 68%). These results highlight the effectiveness of fine-tuning in improving enhancer detection in mice, with notable gains in accuracy, recall, specificity, precision, and F1 score.

*However, it is important to emphasize that even without fine-tuning, the human network remains a robust and reliable baseline.* While fine-tuning provides substantial benefits when additional data is available, the human network alone offers a strong foundation for enhancer detection, delivering valuable results even in data-limited scenarios. Next, we asked: How much data are needed for effective fine-tuning?

## The Human Network Can Be Successfully Fine-Tuned with Just 20K Mouse Enhancers

With the rise of large-scale genomic initiatives such as the 10,000 Vertebrate Genomes Project (Genome 10K Community of Scientists, 2009) and the Earth BioGenome Project (Lewin *et al.*, 2018), there is a growing demand for computational methods capable of predicting enhancers in newly sequenced genomes. Experimental approaches to enhancer identification are costly, labor-intensive, and time-consuming, making efficient *in silico* alternatives increasingly valuable. We have shown that our human-trained network generalizes reasonably well to the mouse genome without fine-tuning. However, performance improves remarkably with fine-tuning on mouse enhancer data. Additionally, certain species — whether new or well-studied — may lack a sufficient number of confirmed enhancers for effective fine-tuning. This raises a practical question: How many enhancers are needed to effectively fine-tune the human network for

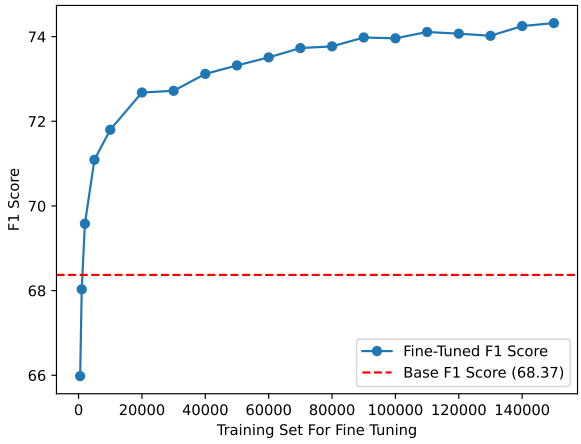



**Figure 5.** Effect of training set size on fine-tuning performance. This plot shows the F1 score achieved by the human-trained network fine-tuned on mouse enhancer data using training subsets of increasing sizes. The x-axis indicates the number of mouse enhancer sequences used for fine-tuning, while the y-axis reports the corresponding F1 score on the validation set. Increasing the number of fine-tuning samples improves enhancer detection performance, with performance gains plateauing at larger dataset sizes.

use in a new species? To address this question, we tested multiple datasets with varying numbers of potential enhancers required for effective fine-tuning of the human network. We used potential enhancers from the mouse training dataset, which comprised 152,744 potential enhancers, and randomly selected subsets ranging from 500 to 150,000 enhancers for training. For each subset of enhancers, we also randomly selected twice as many control sequences from the training dataset. These subsets of enhancers and control sequences allowed us to evaluate the impact of varying numbers of potential enhancers on the fine-tuning procedure.

F1 scores on the validation set for each model are shown in Figure 5. The baseline (using the human network without fine-tuning) achieved an F1 score of 68%. Models fine-tuned on just 500 or 1,000 mouse enhancers performed comparably to or slightly worse than the baseline (66% and 68%). Marked improvements were observed when the training set included 2,000 to 20,000 enhancers, reaching up to a 73% F1 score. These gains were achieved using only about 20% of the available mouse enhancer data. While increasing the dataset size beyond 20,000 led to marginal additional gains (e.g., up to 74%), the improvements became less pronounced, suggesting that about 20,000 enhancers provide a strong foundation for fine-tuning our human network to a new genome. These results demonstrate the potential of our fine-tuning approach for broader cross-species enhancer prediction.

## CAM-Based Knockout Experiments Reveal Biologically Significant Enhancer Regions

Class Activation Maps (CAMs) are visual tools that highlight the regions of an input sequence that are most influential in a neural network's classification decision. They help interpret which features the model has learned to associate with a given class. This technique outputs a score for each nucleotide in the sequence, typically ranging from 0 (least important) to 1 (most important). These scores can be visualized as a heat map: nucleotides with scores closer to 1 appear reddish,

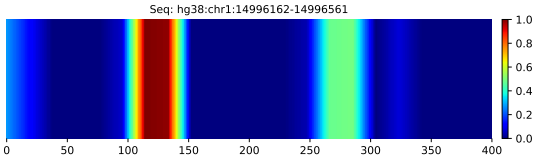



**Figure 6.** Class Activation Map (CAM) visualization for a true enhancer — a sequence labeled as an enhancer and correctly classified as such by the neural network. CAM is a technique that assigns importance scores to each nucleotide in the sequence based on how much it contributed to the model's classification. These scores range from 0 (least important) to 1 (most important) and are visualized as a heat map: nucleotides with scores closer to 1 appear reddish, while those closer to 0 appear bluish. In this example, the dark red region spanning positions 104 to 142 indicates a strong enhancer signal, suggesting that this portion of the sequence played a key role in the network's decision. The blue regions were less or not influential at all.

indicating high importance, while those closer to 0 appear bluish, indicating low importance. An example CAM is shown in Figure 6. In this figure, the dark red region spanning positions 104 to 142 highlights a portion of the sequence that strongly influences the network's classification. In contrast, the blue regions indicate areas with little to no relevance to the network's decision-making process.

To evaluate the biological significance of the regions highlighted by CAMs, we conducted a knockout experiment. The key idea was to disrupt ("knock out") the most important region of an enhancer sequence — identified using CAM — and assess whether the sequence was still classified as an enhancer by the network. We began with 25,000 true enhancer sequences and computed their CAM scores to identify regions of high importance. We defined important regions using a thresholding system with values of 0.2, 0.3, 0.4, and 0.5. These thresholds represent the minimum average CAM score required for a region to be considered important. Based on this criterion, we selected sequences containing only one important region between 25 and 100 bp in length. This constraint was critical: destroying long important regions would disrupt a large portion of the sequence and likely lead to a negative classification, while regions shorter than 25 bp would be too small to meaningfully impact classification. For each selected sequence, we shuffled the nucleotides within the important region, effectively removing its recognizable pattern. As a control, we shuffled a region of the same length elsewhere in the sequence — ensuring it did not overlap with the important region. The classifier was then applied to modified sequences to observe the effect of the knockouts.

Finally, we assessed the significance of the results using a binomial test, which calculates p-values based on the number of sequences tested. This statistical evaluation helped confirm the observed differences in classification outcomes were not due to random chance, but rather to the disruption of biologically relevant regions identified by CAMs.

Shuffling the regions identified as important by CAMs significantly reduced the number of sequences classified as enhancers across all thresholds. At a threshold of 0.2, 1,409 sequences (94% of true enhancers with important regions averaging an intensity of 0.2) were reclassified as non-enhancers. This effect persisted across higher thresholds: 2,251 sequences (92%) at 0.3, 3,070 (90%) at 0.4, and 4,345 (87%) at 0.5. In total, disrupting CAM-identified regions led to loss of enhancer classification in 87–94% of cases. As a control, we shuffled a region outside the CAM-highlighted

area within the same sequences. This had a smaller but still substantial effect: at thresholds of 0.2, 0.3, 0.4, and 0.5, the number of sequences reclassified as non-enhancers was 1,281 (85%), 1,928 (79%), 2,443 (72%), and 3,371 (68%). Next, we computed p-values comparing the proportion of negative classifications resulting from shuffling important versus non-important regions. The p-values for thresholds 0.2 through 0.5 were $6.87 \times 10^{-26}$, $8.94 \times 10^{-71}$, $1.34 \times 10^{-151}$, and $1.08 \times 10^{-225}$ — all highly significant. We also evaluated two alternative knockout strategies: reverse sequencing (e.g., AGCT $\rightarrow$ TCGA) and transversion mutation, where each nucleotide (A, C, G, T) is randomly replaced with either of the two remaining non-complementary bases (e.g., A becomes G or C, determined by chance). Both approaches yielded reclassification rates comparable to the shuffling approach. At a threshold of 0.2, the reclassification rates to non-enhancer were 92% for the mutation approach and 95% for reverse sequencing vs. 94% for the shuffled knockout. At 0.3, the rates were 89% and 92% vs. 92%. For 0.4, we observed 86% and 91% vs. 90%. Finally, at 0.5, the rates were 84% and 88% vs. 87%. All of these percentages were statically significant.

These results demonstrate that the regions identified by CAMs are not only predictive but also biologically meaningful. Disrupting the important regions has a greater impact on enhancer classification than disturbing other regions. *Notably, the experimental reversal of key enhancer regions strongly suggests a directional nature, independently confirming recent observations (Agrawal* et al.*, 2025).*

### Insertion Experiment Further Supports the Biological Relevance of CAM-Identified Regions

To further demonstrate the importance of the CAM-identified regions, we conducted an insertion experiment. The goal was to test whether inserting a high-scoring region from a true enhancer into a non-enhancer sequence could cause the network to reclassify it as an enhancer. We began with 25,000 true enhancer sequences and applied the same selection criteria as in the knockout experiment: each sequence had to contain exactly one important region identified by CAM, with a length between 25 and 100 bp. For each selected enhancer, we paired it with a true negative sequence — one that was labeled and correctly classified as a non-enhancer. We then inserted the important region from the enhancer into the corresponding negative sequence at the same relative position. After this modification, the new sequence was passed through the network for reclassification. As a control, we performed a parallel experiment in which a randomly selected region of the same size — but from outside the CAM-highlighted area — was inserted into the negative sequence. This ensured that any observed changes in classification were specifically due to the informative region identified by CAM, rather than arbitrary sequence changes. Finally, we evaluated the significance of the observed classification changes using a binomial test, demonstrating that the effects were not due to random chance.

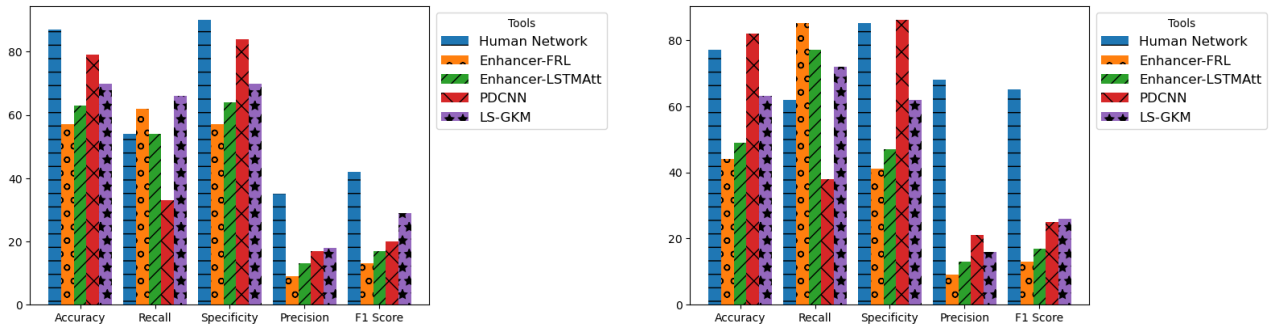
For CAM thresholds of 0.2, 0.3, 0.4, and 0.5, the number of sequences reclassified as enhancers was 1,132 (76%), 1,937 (79%), 2,745 (80%), and 4,030 (81%). As a control, we inserted regions outside of the CAM-identified important areas into true negatives. The number of sequences reclassified as enhancers in this case was 877 (59%), 1,485 (61%), 2,042 (60%), and 2,990 (60%) for the respective thresholds. We then calculated p-values comparing the reclassification rates resulting from inserting important versus non-important regions. The p-values for thresholds 0.2 through 0.5 were $1.70 \times 10^{-43}$, $4.01 \times 10^{-85}$, $3.87 \times 10^{-146}$, and $2.65 \times 10^{-220}$ — all highly significant. Insertion of CAM-identified regions consistently increased enhancer classification rates across all thresholds, with stronger effects at higher CAM intensities. In contrast, insertion of non-important regions had moderate but notably weaker impact. These findings confirm the biological relevance of CAM-identified regions in enhancer sequences.

### Sequence Context Surrounding Enhancer Regions Contributes to Classification

While our knockout and insertion experiments demonstrated the functional significance of CAM-identified regions, we next examined whether the surrounding sequence context also influences enhancer classification. Specifically, we asked: Is preserving only the important region sufficient for the model to recognize a sequence as an enhancer? We designed the experiment to be similar to the knockout experiment, except instead of shuffling the important region itself, we shuffled the adjacent context of same length as the important region. For example, if the important region is 100 base pairs long, we shuffle the 50 base pairs directly before and after it, while keeping the important region unchanged. This allowed us to test whether the network relies on the broader sequence context, beyond the specific important region, for enhancer classification. As with the previous experiments, we selected 25,000 true enhancer sequences, applying the same criteria for selection: each sequence contained one important region between 25 and 100 bp in length. Once the sequences were chosen, we took the length of the important region and shuffled the regions on both adjacent sides of the important region to match. After modifying the sequences, we passed them through the network again to determine whether they would still be classified as enhancers.

At a CAM threshold of 0.2, 1,277 sequences (85% of the samples) were reclassified as non-enhancers. For thresholds of 0.3 and 0.4, the number of reclassified sequences was 1,959 (80%) and 2,524 (74%). At threshold 0.5, 3,553 sequences (71%) were reclassified as non-enhancers. Similar to the knockout experiment, we applied the following three disturbance approaches: (i) k-mer shuffling, (ii) transverse mutation, and (iii) sequence reverse. The results were comparable to those of the shuffled approach. At the 0.2 threshold, reclassification rates were 82% for mutation, 86% for reverse, and 85% for shuffled. At 0.3, the rates were 76% and 81% vs. 80%. For 0.4, we observed 71% and 75% vs. 74%. Finally, at 0.5, the rates were 69% and 74% vs. 71%.

These results indicate that preserving only the most predictive region is insufficient for accurate enhancer classification. Disrupting the surrounding sequence context often leads the model to reject the sequence as an enhancer, even when the core region remains intact. This finding is consistent with biological evidence suggesting that enhancer function depends not only on core transcription factor binding motifs but also on adjacent nucleotides that




(a) ENCODE Human Enhancers

(b) ENCODE Mouse Enhancers

**Figure 7.** Performance comparison of enhancer classification tools across human and mouse ENCODE datasets. Bar plots display accuracy, recall, specificity, precision, and F1 score for five enhancer classification tools: our human network, Enhancer-FRL, Enhancer-LSTMAtt, PDCNN, and LS-GKM. All tools were evaluated on the same test sets for both human and mouse enhancers. Our human network consistently outperforms the others in precision and F1 score, while maintaining competitive accuracy, recall, and specificity.

influence chromatin accessibility and higher-order regulatory architecture (Grossman *et al.*, 2017).

A comparison with the knockout experiments reveals that both the core region and its context contribute to enhancer activity. However, perturbing the core region resulted in a higher rate of loss-of-function predictions, indicating it plays a more central role. Together, these findings suggest that enhancer function emerges from an interplay between essential sequence motifs and their genomic context — both of which are effectively captured by our model.

## Comparing *EnhancerDetector* to Related Tools

To assess the performance of our human-trained network, we compared it to several other binary classification tools for human enhancer prediction: Enhancer-FRL, Enhancer-LSTMAtt, PDCNN, and LS-GKM. All tools were evaluated using both human and mouse ENCODE datasets.

We first compared our network to LS-GKM trained on the CATlas dataset, using the same unseen test set for evaluation. Our model outperformed LS-GKM in overall accuracy (82% vs. 69%). While LS-GKM achieved higher recall (72% vs. 66%), our model showed superior specificity (89% vs. 69%), precision (75% vs. 19%), and F1 score (72% vs. 30%). These results suggest that LS-GKM tends to over-classify sequences as enhancers, whereas *EnhancerDetector* achieves a more balanced performance, making it better suited for genome-scale enhancer discovery.

We next evaluated all tools on the ENCODE datasets (Figure 7). We began by comparing our human network to LS-GKM trained on the CATlas dataset. Our model consistently outperformed LS-GKM across all metrics except recall. On the human dataset, our network achieved higher accuracy (87% vs. 70%), specificity (90% vs. 70%), precision (35% vs. 18%), and F1 score (42% vs. 29%). LS-GKM showed better recall (66% vs. 54%), detecting more enhancers but at the cost of lower specificity and precision — again suggesting a tendency to over-predict enhancer regions. A similar trend was observed on the mouse dataset: LS-GKM had lower accuracy (63% vs. 77%) and specificity (62% vs. 85%), with higher recall (72% vs. 62%) but markedly lower precision (16% vs. 68%) and F1 score (26% vs. 65%). These results confirm that LS-GKM over-calls enhancers, increasing false positives, while our model achieves more balanced and reliable predictions.

On the human ENCODE data, our human network achieved markedly higher accuracy than Enhancer-FRL (87% vs. 57%). While Enhancer-FRL showed higher recall (62% vs. 52%), it suffered from much lower specificity (57% vs. 90%), suggesting a tendency to over-classify sequences as enhancers. This pattern is further reflected in its substantially lower precision (9% vs. 35%) and F1 score (13% vs. 42%) compared to our model. These results highlight our network's stronger overall performance and better balance between recall and precision. A similar trend was observed on the mouse data: our model achieved higher accuracy (77% vs. 44%), while Enhancer-FRL maintained higher recall (85% vs. 62%) but showed lower specificity (41% vs. 85%). Once again, our model outperformed Enhancer-FRL in precision (9% vs. 68%) and F1 score (13% vs. 65%). *Notably, although trained only on human enhancers, our network generalized better to mouse data, highlighting its strong cross-species applicability.*

When compared to Enhancer-LSTMAtt, our human network demonstrated superior performance across nearly all metrics. On the human dataset, our model achieved higher accuracy (87% vs. 63%), specificity (90% vs. 64%), precision (35% vs. 13%), and F1 score (42% vs. 17%). Both models exhibited similar recall (54%). Enhancer-LSTMAtt's low precision and specificity suggest a tendency to over-predict enhancers by misclassifying many non-enhancer sequences. On the mouse dataset, our human network again outperformed Enhancer-LSTMAtt. Our model achieved higher accuracy (77% vs. 49%), specificity (85% vs. 47%), precision (68% vs. 13%), and F1 score (65% vs. 17%). Although Enhancer-LSTMAtt had higher recall (77% vs. 62%), its poor precision and specificity reinforce its tendency to over-classify sequences as enhancers.

Finally, we evaluated PDCNN on the ENCODE human enhancer dataset and found that our human network outperformed it across all metrics. PDCNN achieved an accuracy of 79%, compared to 87% for our network. Our model also showed higher recall (54% vs. 33%), precision (35% vs. 17%), and F1 score (42% vs. 20%). Both tools exhibited high specificity, with PDCNN at 84% and our

model at 90%. PDCNN's combination of low recall and high specificity suggests it does not over-predict enhancers to the extent observed in Enhancer-FRL and Enhancer-LSTMAtt. However, its low precision and F1 score indicate that its enhancer predictions are less reliable than those of our model. *PDCNN was the only tool trained on both human and mouse enhancers*. When evaluated on the mouse dataset, PDCNN achieved an accuracy of 82% and a specificity of 86%, comparable to our model's 77% accuracy and 85% specificity, even though our model was trained solely on human data. Further, our network performed substantially better in recall (62% vs. 38%), precision (68% vs. 21%), and F1 score (65% vs. 25%). Similar to its performance on human data, PDCNN showed a lower tendency to over-predict enhancers. The markedly higher precision and F1 score of our human network underscore its true potential for accurate enhancer prediction across vertebrate species.

In sum, our human network emerges as a more effective enhancer classification tool than Enhancer-FRL, Enhancer-LSTMAtt, PDCNN, and LS-GKM. It consistently achieves higher accuracy, specificity, precision, and F1 score on both human and mouse datasets. These results highlight the strength of our model in delivering reliable enhancer predictions in humans and other vertebrate species.

## CONCLUSION

In this study, we introduced *EnhancerDetector*, a convolutional neural network-based framework for accurate and interpretable identification of transcriptional enhancers across species. *EnhancerDetector* consistently outperforms existing enhancer prediction methods, achieving higher accuracy, precision, and F1 score across both human and mouse datasets. The tool maintains high specificity and precision — crucial for genome-wide applications where minimizing false positives is essential. It also generalizes well to independent datasets generated using different experimental assays (e.g., snATAC-seq, CAGE, and DNase-seq). A key strength of *EnhancerDetector* is its cross-species transferability: the model trained on human enhancers performs robustly on mouse and fly data and can be effectively fine-tuned with as few as 20,000 enhancers from a new species. This feature makes *EnhancerDetector* particularly valuable for enhancer discovery in newly sequenced genomes with limited experimental data. We further leveraged model interpretability through class activation maps to identify sequence regions most predictive of enhancer activity. Using these maps, we conducted *in silico* experiments — including knockout, insertion, and context disturbance — which confirmed the biological relevance of the model's predictions. These analyses revealed that enhancers typically possess a core region essential for function and that the surrounding sequence context also contributes to regulatory activity. This property, which we refer to as *enhancerness*, reflects the intrinsic sequence and contextual features that confer functional enhancer identity. To validate our tool's predictive power experimentally, we tested six putative enhancers in transgenic flies. Five (83%) drove reporter gene expression, and four (67%) exhibited expression patterns supported by prior literature or consistent with those of their putative target genes. In sum, *EnhancerDetector* offers a high-performing, biologically grounded, and easily adaptable framework for enhancer discovery. It lays a powerful foundation for the next generation of computational regulatory genomics, enabling accurate enhancer discovery at scale — precisely when the need is greatest, as the field enters an era of hundreds of thousands of newly sequenced genomes.

## ACKNOWLEDGEMENTS

Research reported in this publication was supported by the National Human Genome Research Institute of the National Institutes of Health under award number: R21HG011507. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. We would like to thank Jack Leatherbarrow and James Weiser for their help with the experimental validations.

## DATA AVAILABILITY

The code is freely available for academic use through github.com/BioinformaticsToolsmith/EnhancerDetector and doi.org/10.5281/zenodo.15531293 under the Affero General Public License version 1.

Restrictions on Use for Profit or Non-Academics: An alternative commercial license is required.

Supplementary File 1: BLAST output of the true positive mouse enhancers versus the human genome.

## CONFLICT OF INTEREST STATEMENT.

None declared.

## REFERENCES

Agrawal, S., Kanamaru, E., Saito, Y., Ishikawa, F., and de Hoon, M. (2025). Cell type-dependent directional transcription at enhancers. *NAR Genomics and Bioinformatics*, **7**(1), lqaf007.

Arnold, C. D., Gerlach, D., Stelzer, C., Boryń, Ł. M., Rath, M., and Stark, A. (2013). Genome-wide quantitative enhancer activity maps identified by STARR-seq. *Science*, **339**(6123), 1074–1077.

Asim, M. N., Ibrahim, M. A., Malik, M. I., Dengel, A., and Ahmed, S. (2020). Enhancer-dsnet: A supervisedly prepared enriched sequence representation for the identification of enhancers and their strength. In H. Yang, K. Pasupa, A. C.-S. Leung, J. T. Kwok, J. H. Chan, and I. King, editors, *Neural Information Processing*, pages 38–48, Cham. Springer International Publishing.

Basith, S., Hasan, M. M., Lee, G., Wei, L., and Manavalan, B. (2021). Integrative machine learning framework for the identification of cell-specific enhancers from the human genome. *Brief Bioinform*, **22**(6), bbab252.

Belton, J.-M., McCord, R. P., Gibcus, J. H., Naumova, N., Zhan, Y., and Dekker, J. (2012). Hi–c: A comprehensive technique to capture the conformation of genomes. *Methods*, **58**(3), 268–276.

Boyle, A. P., Davis, S., Shulha, H. P., Meltzer, P., Margulies, E. H., Weng, Z., Sunyaev, S. R., Bernstein, I. D., Crawford, G. E., and Furey, T. S. (2008). High-resolution mapping and characterization of open chromatin across the genome. *Cell*, **132**(2), 311–322.

Buenrostro, J. D., Wu, B., Chang, H. Y., and Greenleaf, W. J. (2015). Atac-seq: A method for assaying chromatin accessibility genome-wide. *Curr Protoc Mol Biol*, **109**(1), 21.29.1–21.29.9.

Cai, L., Ren, X., Fu, X., Peng, L., Gao, M., and Zeng, X. (2020). iEnhancer-XG: interpretable sequence-based enhancers and their strength predictor. *Bioinformatics*, **37**(8), 1060–1067.

Catarino, R. R. and Stark, A. (2018). Assessing sufficiency and necessity of enhancer activities for gene expression and the mechanisms of transcription activation. *Genes Dev*, **32**(3-4), 202–223.

Cusanovich, D. A., Reddington, J. P., Garfield, D. A., Daza, R. M., Aghamirzaie, D., Marco-Ferreres, R., Pliner, H. A., Christiansen, L., Qiu, X., Steemers, F. J., Trapnell, C., Shendure, J., and Furlong, E. E. M. (2018). The cis-regulatory dynamics of embryonic development at single-cell resolution. *Nature*, **555**(7697), 538–542.

Fossett, N., Zhang, Q., Gajewski, K., Choi, C. Y., Kim, Y., and Schulz, R. A. (2000). The multitype zinc-finger protein U-shaped functions in heart cell specification in the *Drosophila* embryo. *Proc Nat Acad Sci USA*, **97**(13), 7348–7353.

Genome 10K Community of Scientists (2009). Genome 10k: A proposal to obtain whole-genome sequence for 10000 vertebrate species. *J Hered*, **100**(6), 659–674.

Ghandi, M., Lee, D., Mohammad-Noori, M., and Beer, M. A. (2014). Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS Comput Biol*, **10**(7), e1003711. Co-first authors: Mahmoud Ghandi and Dongwon Lee.

Giresi, P. G., Kim, J., McDaniell, R. M., Iyer, V. R., and Lieb, J. D. (2007). Faire (formaldehyde-assisted isolation of regulatory elements) isolates active regulatory elements from human chromatin. *Genome Res*, **17**(6), 877–885.

Girgis, H. Z. (2015). Red: an intelligent, rapid, accurate tool for detecting repeats de-novo on the genomic scale. *BMC Bioinformatics*, **16**(1), 227.

Goto, T., Macdonald, P., and Maniatis, T. (1989). Early and late periodic patterns of even skipped expression are controlled by distinct regulatory elements that respond to different spatial cues. *Cell*, **57**(3), 413–422.

Grossman, S. R., Zhang, X., Wang, L., Engreitz, J., Melnikov, A., Rogov, P., Tewhey, R., Isakova, A., Deplancke, B., Bernstein, B. E., Mikkelsen, T. S., and Lander, E. S. (2017). Systematic dissection of genomic features determining transcription factor binding and enhancer function. *Proc Nat Acad Sci USA*, **114**(7), E1291–E1300.

Harding, K., Hoey, T., Warrior, R., and Levine, M. (1989). Autoregulatory and gap gene response elements of the even-skipped promoter of *Drosophila*. *EMBO J*, **8**, 1205–1212.

Hariprakash, J. M. and Ferrari, F. (2019). Computational biology solutions to identify enhancers-target gene pairs. *Comput Struct Biotechnol J*, **17**, 821–831.

Hinrichs, A. S., Karolchik, D., Baertsch, R., Barber, G. P., Bejerano, G., Clawson, H., Diekhans, M., Furey, T. S., Harte, R. A., Hsu, F., *et al.* (2006). The ucsc genome browser database: update 2006. *Nucleic Acids Res*, **34**(Database issue), D590–D598.

Hiromi, Y. and Gehring, W. J. (1987). Regulation and function of the *Drosophila* segmentation gene *fushi tarazu*. *Cell*, **43**, 603–613.

Hu, W., Li, Y., Wu, Y., Guan, L., and Li, M. (2024). A deep learning model for dna enhancer prediction based on nucleotide position aware feature encoding. *iScience*, **27**(6), 110030.

Huang, G., Luo, W., Zhang, G., Zheng, P., Yao, Y., Lyu, J., Liu, Y., and Wei, D.-Q. (2022). Enhancer-lstmatt: A bi-lstm and attention-based deep learning method for enhancer recognition. *Biomolecules*, **12**(7).

Jenett, A., Rubin, G. M., Ngo, T.-T., Shepherd, D., Murphy, C., Dionne, H., Pfeiffer, B. D., Cavallaro, A., Hall, D., Jeter, J., Iyer, N., Fetter, D., Hausenfluck, J. H., Peng, H., Trautman, E. T., Svirskas, R. R., Myers, E. W., Iwinski, Z. R., Aso, Y., DePasquale, G. M., Enos, A., Hulamm, P., Lam, S. C. B., Li, H.-H., Laverty, T. R., Long, F., Qu, L., Murphy, S. D., Rokicki, K., Safford, T., Shaw, K., Simpson, J. H., Sowell, A., Tae, S., Yu, Y., and Zugates, C. T. (2012). A GAL4-driver line resource for *Drosophila* neurobiology. *Cell Rep*, **2**(4), 991–1001.

Jia, C. and He, W. (2016). Enhancerpred: a predictor for discovering enhancers based on the combination and selection of multiple features. *Sci Rep*, **6**(1), 38741.

Jia, J., Lei, R., Qin, L., Wu, G., and Wei, X. (2023). ienhancer-dcsv: Predicting enhancers and their strength based on densenet and improved convolutional block attention module. *Front Genet*, **14**.

Keränen, S. V. E., Rivera, J., Gallo, S. M., and Halfon, M. S. (2019). Redfly: the transcriptional regulatory element database for drosophila. *Nucleic Acids Res*, **47**(D1), D828–D834.

Kim, T.-K., Hemberg, M., Gray, J. M., and et al. (2010). Widespread transcription at neuronal activity-regulated enhancers. *Nature*, **465**(7295), 182–187.

Klemm, S. L., Shipony, Z., and Greenleaf, W. J. (2019). Chromatin accessibility and the regulatory epigenome. *Nat Rev Genet*, **20**(4), 207–220.

Lee, D. (2016). Ls-gkm: A new gkm-svm for large-scale datasets. *Bioinformatics*, **32**(14), 2196–2198.

Lewin, H. A., Robinson, G. E., Kress, W. J., Baker, W. J., Coddington, J., Crandall, K. A., Durbin, R., Edwards, S. V., Forest, F., Gilbert, M. T. P., Goldstein, M. M., Grigoriev, I. V., Hackett, K. J., Haussler, D., Jarvis, E. D., Johnson, W. E., Patrinos, A., Richards, S., Castilla-Rubio, J. C., van Sluys, M.-A., Soltis, P. S., Xu, X., Yang, H., and Zhang, G. (2018). Earth BioGenome Project: Sequencing life for the future of life. *Proc Nat Acad Sci USA*, **115**(17), 4325–4333.

Li, L., Zhu, Q., He, X., Sinha, S., and Halfon, M. S. (2007). Large-scale analysis of transcriptional cis-regulatory modules reveals both common features and distinct subclasses. *Genome Biol*, **8**(6), R101.

Liao, M., ping Zhao, J., Tian, J., and Zheng, C.-H. (2022). ienhancer-dcla: using the original sequence to identify enhancers and their strength based on a deep learning framework. *BMC Bioinformatics*, **23**(1), 480.

Liu, B., Fang, L., Long, R., Lan, X., and Chou, K.-C. (2015). iEnhancer-2L: a two-layer predictor for identifying enhancers and their strength by pseudo k-tuple nucleotide composition. *Bioinformatics*, **32**(3), 362–369.

Liu, B., Li, K., Huang, D.-S., and Chou, K.-C. (2018). iEnhancer-EL: identifying enhancers and their strength with ensemble learning approach. *Bioinformatics*, **34**(22), 3835–3842.

Mehmood, F., Arshad, S., and Shoaib, M. (2024). ADH-Enhancer: an attention-based deep hybrid framework for enhancer identification and strength prediction. *Brief Bioinform*, **25**(2), bbae030.

Muratoglu, S., Garratt, B., Hyman, K., Gajewski, K., Schulz, R. A., and Fossett, N. (2006). Regulation of *Drosophila* friend of GATA gene, u-shaped, during hematopoiesis: A direct role for Serpent and Lozenge. *Dev Biol*, **296**(2), 561–579.

Négre, N., Brown, C. D., Shah, P. K., Kheradpour, P., Morrison, C. A., Henikoff, J. G., Feng, X., Ahmad, K., Russell, S., White, R. A. H., Stein, L., Henikoff, S., Kellis, M., and White, K. P. (2010). A comprehensive map of insulator elements for the drosophila genome. *PLoS Genet*, **6**(1), 1–13.

Niu, K., Luo, X., Zhang, S., Teng, Z., Zhang, T., and Zhao, Y. (2021). ienhancer-eblstm: Identifying enhancers and strengths by ensembles of bidirectional long short-term memory. *Front Genet*, **12**.

Park, P. J. (2009). ChIP–seq: advantages and challenges of a maturing technology. *Nature Rev Genet*, **10**(10), 669–680.

Quinlan, A. R. and Hall, I. M. (2010). Bedtools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**(6), 841–842.

Thomas, S., Li, X.-Y., Sabo, P. J., and et al. (2011). Dynamic reprogramming of chromatin accessibility during *Drosophila* embryo development. *Genome Biol*, **12**(5), R43.

Tokusumi, T., Tokusumi, Y., Brahier, M. S., and et al. (2017). Screening and analysis of Janelia FlyLight Project enhancer-gal4 strains identifies multiple gene enhancers active during hematopoiesis in normal and wasp-challenged *Drosophila* larvae. *G3*, **7**(2), 437–448.

Visel, A., Blow, M. J., Li, Z., Zhang, T., and et al. (2009). ChIP-seq accurately predicts tissue-specific activity of enhancers. *Nature*, **457**(7231), 854–858.

Wang, C., Zou, Q., Ju, Y., and Shi, H. (2023). Enhancer-frl: Improved and robust identification of enhancers and their activities using feature representation learning. *IEEE/ACM Trans Comput Biol Bioinform*, **20**(2), 967–975.

Winter, S. D., Konstantakos, V., and Aerts, S. (2025). Modelling and design of transcriptional enhancers. *Nat Rev Bioeng*, **3**(5), 374–389.

Yang, H., Wang, S., and Xia, X. (2021a). ienhancer-rd: Identification of enhancers and their strength using rkpk features and deep neural networks. *Anal Biochem*, **630**, 114318.

Yang, R., Wu, F., Zhang, C., and Zhang, L. (2021b). iEnhancer-GAN: A deep learning framework in combination with word embedding and sequence generative adversarial net to identify enhancers and their strength. *Int J Mol Sci*, **22**(7).

Zaugg, J. B., Sahlén, P., Andersson, R., Alberich-Jorda, M., de Laat, W., Deplancke, B., Ferrer, J., Mandrup, S., Natoli, G., Plewczynski, D., Rada-Iglesias, A., and Spicuglia, S. (2022). Current challenges in understanding the role of enhancers in disease. *Nat Struct Mol Biol*, **29**(12), 1148–1158.

Zhu, Q. and Halfon, M. S. (2007). Vector-dependent gene expression driven by insulated p-element reporter vectors. *Fly*, **1**(1), 55–56.