



Palestine Polytechnic University

Deanship of Graduate Studies and Scientific Research

Master of Intelligent Systems

Multimodal Convolutional Neural Network for Pathogenicity Classification of Missense Variants

Submitted by:

Eng. Lana Sharabati

Supervisors:

Dr. Alaa Halawani

Dr. Yaqoub Ashhab

Thesis submitted in partial fulfillment of requirements of the degree

Master of Science in Intelligent Systems

September, 2026

The undersigned hereby certify that they have read, examined and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the approval of a thesis entitled: *Multimodal Convolutional Neural Network for Pathogenicity Classification of Missense Variants*, submitted by Eng. Lana Sharabati in partial fulfillment of the requirements for the degree of Master of Science in Intelligent Systems.

Graduate Advisory Committee:

Dr. Alaa Halawani (Supervisor), Palestine Polytechnic University.

Signature: _____ Date: _____

Dr. Yaqoub Ashhab (Supervisor), Palestine Polytechnic University.

Signature: _____ Date: _____

Dr. Hashem Tamimi (Internal committee member), Palestine Polytechnic University.

Signature: _____ Date: _____

Dr. Mohamed Alshalalfa (External committee member), Veracyte.

Signature: _____ Date: _____

Thesis Approved

Dr. Mahmoud Al-Haddad

Dean of Graduate Studies and Scientific Research

Palestine Polytechnic University

Signature: _____ Date: _____

DECLARATION

I declare that the Master Thesis entitled “Multimodal Convolutional Neural Network for Pathogenicity Classification of Missense Variants” is my original work, and hereby certify that unless stated, all work contained within this thesis is my own independent research and has not been submitted for the award of any other degree at any institution, except where due acknowledgement is made in the text.

Artificial intelligence tools were used to improve the language of this thesis and to assist with parts of the code. The research design, data collection, analysis, results, and conclusions are my own work, and I take full responsibility for the content.

Eng. Lana Sharabati

Signature: _____ Date: _____

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for the master degree in Intelligent Systems at Palestine Polytechnic University, I agree that the library shall make it available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of the source is made.

Permission for extensive quotation from, reproduction, or publication of this thesis may be granted by my main supervisor, or in his absence, by the Dean of Graduate Studies and Scientific Research when, in the opinion of either, the proposed use of the material is for scholarly purposes.

Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Eng. Lana Sharabati

Signature: _____ Date: _____

DEDICATION

To my parents, Mowaffaq and Abeer, thank you for your endless love, sacrifices, and support. Thank you for always believing in me and encouraging me, especially when things were difficult. Everything I have achieved is because of your love and support.

To Dema, Dania, Batool, and Zaid, thank you for always being there for me, supporting me, and making this journey easier and happier.

To my family, this work is dedicated to all of you. I am truly grateful for your love, support, and for always being by my side. This achievement belongs to all of us.

ACKNOWLEDGEMENT

First, I would like to thank Allah for giving me the strength, patience, and ability to complete this work.

I would like to express my sincere thanks to my supervisors, Dr. Alaa Halawani and Dr. Yaqoub Ashhab, for their guidance, support, and valuable advice throughout this research. I am truly grateful for their time, patience, and encouragement, which helped me complete this thesis.

I would also like to thank all the professors and staff who supported me during my studies and shared their knowledge and experience with me.

To my friends and colleagues, thank you for your support, encouragement, and for always being there when I needed help. Your support made this journey easier.

Finally, I would like to thank everyone who helped, supported, or encouraged me throughout this journey. I am truly grateful to all of you.

Abstract

Interpreting genetic missense variants is critical for diagnosing inherited disease. In the ClinVar missense data used here, 80% of variants were classified as Variants of Uncertain Significance (VUS), which leaves patients without clear diagnostic guidance. This thesis combines ClinVar with DescribePROT, a database of per-residue structural and functional descriptors, to classify missense variants by their clinical significance. These descriptors have not been applied to variant classification before. Each variant is represented by a 31×15 matrix of the protein window around the mutation, holding 14 structural and functional amino acid features, together with an eight-feature vector of population allele frequencies and physicochemical substitution scores. Five models were compared on 117,843 variants: three main architectures (VGG, VGG CNN + MLP, and DeepInsight with DenseNet121) and two single-input ablations. Once VUS are removed, ClinVar labels the remaining variants in four categories, and the models were trained on those. The VGG CNN + MLP model performed best, with a Macro F1-score of 0.6179, an accuracy of 66.77%, and a Macro ROC-AUC of 0.8910, and it held a Macro ROC-AUC of 0.8811 on an independent external ClinVar validation set of 34,146 variants. Grouped into the binary benign against pathogenic split that clinical practice acts on, the same architecture reached a ROC-AUC of 0.972. The ablations located the predictive signal: the eight variant-level features alone recovered 92% of the best model’s Macro F1, while the DeepInsight image alone performed near chance. Feature integration and variant-level annotation, rather than added network complexity, drive performance on this task.

Contents

Committee Approval	i
Declaration	ii
Statement of Permission to Use	iii
Dedication	iv
Acknowledgement	v
1 Introduction	2
1.1 Problem Statement	2
1.2 Research Objectives	3
1.3 Thesis Structure	4
2 Background	5
2.1 Human Genome and Genetic Variation	5
2.2 Clinical Variant and Protein Databases	7
2.2.1 ClinVar	7
2.2.2 UniProt	7
2.2.3 DescribePROT	8
2.2.4 Population Frequency Databases	8
2.3 Protein Structure and Function Features	10
2.3.1 Primary and Secondary Structure	10
2.3.2 Intrinsic Disorder	10
2.3.3 Solvent Accessibility	11
2.3.4 Molecular Binding and Interaction Sites	11
2.3.5 Post-Translational Modifications	12
2.3.6 Evolutionary Conservation	12
2.3.7 Physicochemical Properties of Amino Acids	13
2.4 Machine Learning	13
2.4.1 Supervised Classification	13
2.5 Neural Networks and Deep Learning	15
2.5.1 Multilayer Perceptron	15
2.5.2 Loss Functions and Optimization	15

2.5.3	Regularization and Overfitting	17
2.6	Convolutional Neural Networks (CNN)	17
2.6.1	Convolution, Filters, and Feature Maps	17
2.6.2	Stride and Padding	18
2.6.3	Activation Functions	18
2.6.4	Pooling Operations	19
2.6.5	From Images to Protein Sequences	19
2.6.6	Multimodal Fusion	20
2.7	Advanced CNN Architectures	20
2.7.1	VGG	20
2.7.2	DenseNet	20
2.8	DeepInsight	22
2.9	Hyperparameter Optimization	22
2.10	Evaluation Metrics	23
3	Literature Review	25
3.1	Traditional and Machine Learning Approaches	25
3.1.1	SIFT	25
3.1.2	PolyPhen-2	26
3.1.3	MutationTaster	26
3.1.4	FATHMM	26
3.1.5	CADD	26
3.1.6	REVEL	27
3.1.7	BayesDel	27
3.1.8	ClinPred	27
3.2	Deep Learning Approaches	28
3.2.1	PrimateAI	28
3.2.2	EVE	28
3.2.3	ESM1b	28
3.2.4	AlphaMissense	29
3.2.5	Other CNN tools	29
3.2.6	DeepInsight in Bioinformatics	31
3.3	Research Gap	31
4	Methodology	33
4.1	Data Collection and Integration	33
4.1.1	ClinVar Extraction	33
4.1.2	Allele Frequency (AF) Enrichment	35
4.1.3	DescribePROT Data	36
4.1.4	Mapping via UniProt	37

4.1.5	Merging ClinVar with DescribePROT	37
4.1.6	Biochemical Computations	39
4.1.7	Dataset Splitting	39
4.1.8	Class Imbalance and Class Weighting	40
4.2	Input Representation	41
4.2.1	Protein Window Matrix Construction	41
4.2.2	Dense Feature Vector	43
4.3	Model Training Framework	45
4.3.1	Baseline VGG-Based CNN Model	45
4.3.2	Dual-Input CNN + MLP Model	48
4.3.3	DeepInsight with DenseNet121 and MLP Model	50
4.3.4	Tabular MLP Model	50
4.3.5	DenseNet CNN-Only Model	52
4.4	Binary Classification	53
4.5	Window Size Selection	54
4.6	External Validation Dataset	55
5	Results and Discussion	56
5.1	Dataset Construction Results	56
5.1.1	Data Quality and Missing Value Handling	56
5.2	Model Evaluation Results	57
5.3	Performance of the Baseline VGG CNN Model	57
5.3.1	Best Hyperparameters (Keras Tuner)	57
5.3.2	Overall Test-Set Metrics	59
5.3.3	Per-Class Metrics	59
5.4	Performance of the Hybrid VGG CNN + MLP Model	59
5.4.1	Best Hyperparameters (Keras Tuner)	61
5.4.2	Overall Test-Set Metrics	61
5.4.3	Per-Class Metrics	63
5.5	Performance of the DeepInsight DenseNet121 + MLP Model	63
5.5.1	Best Hyperparameters (Keras Tuner)	63
5.5.2	Overall Test-Set Metrics	63
5.5.3	Per-Class Metrics	63
5.6	Ablation Study: Isolating the Input Sources	66
5.6.1	Tabular MLP (Eight Features Only)	66
5.6.2	DenseNet CNN-Only (DeepInsight Image Only)	68
5.6.3	Feature Importance	69
5.7	Window Size Results	70
5.8	Binary Classification Results	72

5.9	External Validation Results	74
5.9.1	Binary External Validation	77
5.10	Discussion	78
5.10.1	External Dataset	81
5.10.2	Significance of DescribePROT Features	82
5.10.3	Comparison with Previous Studies	82
5.10.4	Limitations	83
5.11	Summary	84
6	Conclusion and Future Work	86
6.1	Conclusion	86
6.2	Future Work	87
	Appendices	95
A	List of Abbreviations	96

List of Figures

2.1	Sequential flow of genetic information from DNA replication and RNA transcription to final protein translation within a cell.	5
2.2	Major Types of Coding Genetic Variants and Their Effects on Protein Translation.	6
2.3	Screenshot for NCBI ClinVar data.	7
2.4	Sample of amino acid features and PTM sites across the NOC2L protein sequence.	9
2.5	Primary and secondary structure of a protein.	10
2.6	Intrinsic disorder in a protein.	11
2.7	PSSM from DescribePROT for positions 153–161 of the NOC2L protein. The outline marks the wild-type residue.	13
2.8	A perceptron and a fully connected multilayer perceptron.	16
2.9	ReLU and Swish activation functions, and a softmax example.	19
2.10	DenseNet dense block with five convolutional layers.	21
2.11	DeepInsight pipeline for tabular-to-image conversion.	22
4.1	Overall methodology workflow: data collection, feature engineering, model training, and evaluation.	34

4.2	Extraction of DNA nucleotide and amino acid changes from SPDI and HGVS annotations.	35
4.3	Decoding of PTMbinary data into residue-specific post-translational modification (PTM) features.	38
4.4	The 31×15 protein-window matrix for a single variant.	42
4.5	Composite allele frequency before (left) and after (right) the $\log_{10}$ transformation.	44
4.6	The eight-feature dense vector for one Pathogenic variant, showing the raw values stored in the dataset file before standardisation. . . .	44
4.7	Activity diagram of the shared model training framework: data preparation, Hyperband hyperparameter search, two-pass training, and overfitting correction.	46
4.8	Baseline VGG-CNN architecture.	47
4.9	Dual-input VGG + MLP architecture (Model 2).	49
4.10	DeepInsight + DenseNet121 + MLP architecture (Model 3).	51
4.11	Tabular MLP architecture (Model 4).	52
4.12	DenseNet CNN-only architecture.	53
5.1	Class distribution of the final dataset after filtering and preprocessing.	56
5.2	One-vs-rest ROC-AUC curves for the baseline VGG CNN model.	60
5.3	Confusion matrix for the baseline VGG CNN model. Values are row-normalised, so each row sums to 1 and the diagonal is the recall for that class.	60
5.4	One-vs-rest ROC-AUC curves for the Hybrid VGG CNN + MLP model.	62
5.5	Confusion matrix for the Hybrid VGG CNN + MLP model. Values are row-normalised, so each row sums to 1 and the diagonal is the recall for that class.	62
5.6	ROC-AUC one-vs-rest curves for the DeepInsight DenseNet121 + MLP model.	65
5.7	Confusion matrix for the DeepInsight DenseNet121 + MLP model. Values are row-normalised.	65
5.8	Confusion matrix for the tabular MLP (eight features only). Values are row-normalised.	67
5.9	One-vs-rest ROC-AUC curves for the tabular MLP (eight features only).	67
5.10	Confusion matrix for the DenseNet CNN-only model (DeepInsight image only). Values are row-normalised.	68

5.11	One-vs-rest ROC-AUC curves for the DenseNet CNN-only model (DeepInsight image only).	69
5.12	Permutation feature importance for the Hybrid VGG CNN + MLP model: drop in Macro F1 when each of the 8 dense features and 15 CNN channels is shuffled on its own.	71
5.13	Binary ROC curves for every model on the same 23,569 test variants. The figure also includes the DeepInsight image-only ablation, which is not listed in Table 5.16.	73
5.14	Class distribution training and test set compared with the external validation set.	75
5.15	Distribution of the eight dense features in the training set and the external validation set, after both were scaled with the StandardScaler fitted on the training data.	76
5.16	Confusion matrix of the Hybrid VGG CNN + MLP model on the external validation set.	77
5.17	One-vs-rest ROC curves of the Hybrid VGG CNN + MLP model on the external validation set.	78
5.18	Confusion matrix of the binary VGG CNN + MLP model on the external validation set, as counts and row-normalised percentages.	79
5.19	ROC curve of the binary VGG CNN + MLP model on the external validation set.	80
5.20	Indicative comparison of binary benign-versus-pathogenic AUC between the proposed models and published pathogenicity predictors.	83

List of Tables

2.1	The ten DescribePROT prediction tools used in this study and the amino acid features they predict.	9
2.2	The window matrix channels (31×15).	14
2.3	The dense vector features.	14
3.1	Summary of pathogenicity predictors and their reported performance.	30
4.1	Training and test set distribution after stratified splitting.	39
4.2	Training and test set distribution for the binary target.	40
4.3	Class weights used during model training.	40

4.4	Hyperparameter search space for VGG-based model.	48
4.5	Additional hyperparameters for dual-input model.	49
4.6	Hyperparameter search space for the tabular MLP model.	52
4.7	Filtering cascade for the external validation dataset.	55
5.1	Summary of the deep learning models evaluated in this study. . . .	58
5.2	Best hyperparameters selected by Keras Tuner for the baseline VGG CNN model.	58
5.3	Overall test-set performance of the baseline VGG CNN model. . . .	59
5.4	Per-class performance metrics for the baseline VGG CNN model. . .	59
5.5	Best hyperparameters selected by Keras Tuner for the Hybrid VGG CNN + MLP model.	61
5.6	Overall test-set performance of the Hybrid VGG CNN + MLP model.	61
5.7	Per-class performance metrics for the Hybrid VGG CNN + MLP model.	63
5.8	Fixed hyperparameters for the DeepInsight DenseNet121 + MLP model. The backbone was trained from scratch and was not tuned with Keras Tuner because of its cost on CPU.	64
5.9	Overall test-set performance of the DeepInsight DenseNet121 + MLP model.	64
5.10	Per-class performance metrics for the DeepInsight DenseNet121 + MLP model.	64
5.11	Per-class performance metrics for the tabular MLP (eight features only).	66
5.12	Per-class performance metrics for the DenseNet CNN-only model (DeepInsight image only).	68
5.13	Overall performance of the five models on the 23,569-variant test set. The dual-input VGG CNN + MLP is best on every metric. Best value per row in bold; for log loss, lower is better.	69
5.14	Effect of the protein window size on the dual-input VGG + MLP model (test set).	70
5.15	Effect of the protein window size on the two binary models (test set, 23,569 variants).	72
5.16	Binary classification performance on the held-out test set of 23,569 variants, of which 34.0% are pathogenic. Each model is shown at its best residue window. Ranked by MCC; best value per column in bold.	73
5.17	Overall performance of the Hybrid VGG CNN + MLP model on the internal held-out test set and on the external validation set. . .	76

5.18	Per-class performance metrics of the Hybrid VGG CNN + MLP model on the external validation set.	77
5.19	Performance of the binary VGG CNN + MLP model on the internal held-out test set and on the independent external validation set. The external set carries a markedly lower pathogenic rate (18.4% against 34.0%), which affects the threshold-dependent metrics far more than the ranking metrics.	79
5.20	Comparison between previous studies and the contributions of this thesis.	82

Chapter 1

Introduction

Human DNA carries the instructions that control how the body works. A change in these instructions is called a genetic variant, and it can alter protein function and cause disease. Deciding whether a variant is pathogenic or benign is one of the main challenges in clinical genetics today [1].

Advances in sequencing technology have made it possible to read a whole human genome quickly and cheaply. Millions of variants now need interpretation, to establish whether they explain a patient’s symptoms. Classification has not kept up with discovery [2].

The ClinVar database, maintained by the National Center for Biotechnology Information (NCBI), is the most widely used resource for variant interpretation [3]. It now holds more than three million records [4]. However, the growth in submissions has not resolved the uncertainty: in the data used in this study, 80% of variants were classified as Variant of Uncertain Significance (VUS), which leaves patients without clear diagnostic guidance.

Machine learning models have been developed to help with this problem. The model learns patterns from known clinical classifications. This offers supporting evidence for clinical interpretation [5]. Recently, many tools have been developed, such as SIFT [6], PolyPhen-2 [7], and more recent deep learning tools such as AlphaMissense [8] and ESM1b [9].

This thesis develops a deep learning model to classify germline missense variants as Benign, Likely Benign, Likely Pathogenic, or Pathogenic by combining clinical variant data from ClinVar with amino acid-level protein features from the DescribePROT database [10].

1.1 Problem Statement

Despite the large number of variant prediction tools available, several limitations remain. Many deep learning tools depend only on amino acid sequences, even

though the structural and functional features of proteins bear directly on protein function [10, 11]. Information about post-translational modification (PTM) sites, which can affect how a variant impacts protein function, is rarely included in these models.

A second limitation concerns how the features are arranged. Representing the structural and functional environment surrounding a mutation site as a spatial matrix, rather than a flat feature vector, allows convolutional networks to learn local positional patterns in the secondary structure, disorder, and conservation around the variant.

This study builds a classifier for germline missense variants on this richer feature representation, with PTM information included. The ACMG/AMP guidelines define five classes [12]. Once variants of uncertain significance are removed, ClinVar labels the remaining variants in four categories, and the models are trained on those. Because clinical practice acts on a binary decision, the same predictions are also reported as pathogenic against benign.

1.2 Research Objectives

This study aimed to develop a deep learning model to classify germline missense variants. Three objectives were defined:

- Construct a dataset that integrates amino acid structural and functional features from DescribePROT with ClinVar germline missense variants of known clinical classification, supported by allele frequency data from gnomAD, TOPMed, the 1000 Genomes Project, and other population databases [13–15].
- Develop a spatial representation of the amino acid environment surrounding each mutation site, as a 31×15 matrix of features, alongside a vector representation of the variant site itself, holding allele frequency, BLOcks SUBstitution Matrix (BLOSUM62) substitution scores, and physicochemical changes as an 8-dimensional feature vector [16].
- Build and compare Convolutional Neural Network (CNN) models, from the Visual Geometry Group (VGG) architecture to multimodal architectures that merge the matrix features with the vector features, including the DeepInsight method for converting tabular protein features into a 2D image suitable for CNN processing [17].

1.3 Thesis Structure

Chapter 1 introduces the research scope and objectives. Chapter 2 gives the biological and computational background needed to understand this study. It describes the protein features used as model inputs and covers machine learning concepts such as CNNs, DenseNet, and DeepInsight. Chapter 3 reviews related work, including existing computational pathogenicity predictors, modern deep learning models, and uses of DeepInsight in bioinformatics. Chapter 4 describes the experimental methodology: data collection, feature extraction and engineering, the two data representations, and the deep learning architectures tested. Chapter 5 starts with a dataset characterization, then reports the performance of each model, the ablation study, and the external validation. It closes with a discussion that interprets the findings, compares the proposed models with published predictors, and examines why the amino acid structural descriptors matter. Chapter 6 concludes the thesis and sets out directions for future work.

Chapter 2

Background

2.1 Human Genome and Genetic Variation

The human body contains approximately 30 to 37 trillion cells, each carrying a copy of the DNA genome. DNA instructions are written in four letters called nucleotides: adenine (A), thymine (T), cytosine (C), and guanine (G). The instructions in DNA are first copied into messenger RNA (mRNA). Every three nucleotides form a unit called a codon, and each codon specifies one of the 20 amino acids or a signal to stop translation. These amino acids combine to form functional proteins. The sections of DNA that carry the protein instructions are called genes, and the human genome contains approximately 20,000 to 25,000 genes [18]. Figure 2.1 illustrates the flow of genetic information from DNA to proteins.

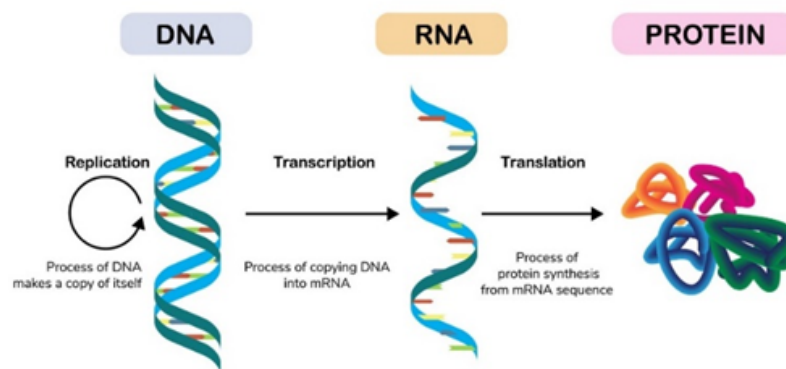


Figure 2.1: Sequential flow of genetic information from DNA replication and RNA transcription to final protein translation within a cell.

Proteins carry out essential functions in the body, acting as enzymes, transporters, signalling molecules, or structural components. The function of a protein depends on its amino acid sequence and its three-dimensional shape [19].

Variants are classified by their effect on the protein. Because different codons can encode the same amino acid, some nucleotide changes leave the protein sequence unchanged; these are synonymous, or silent, variants. Insertions or deletions that are not multiples of three shift the reading frame and usually destroy protein function, and are called frameshift variants. A nonsense variant introduces a premature stop codon, which shortens the protein. A variant that replaces one amino acid with another is a missense variant [20]. Figure 2.2 shows the major types of coding variants and their effects on protein translation.

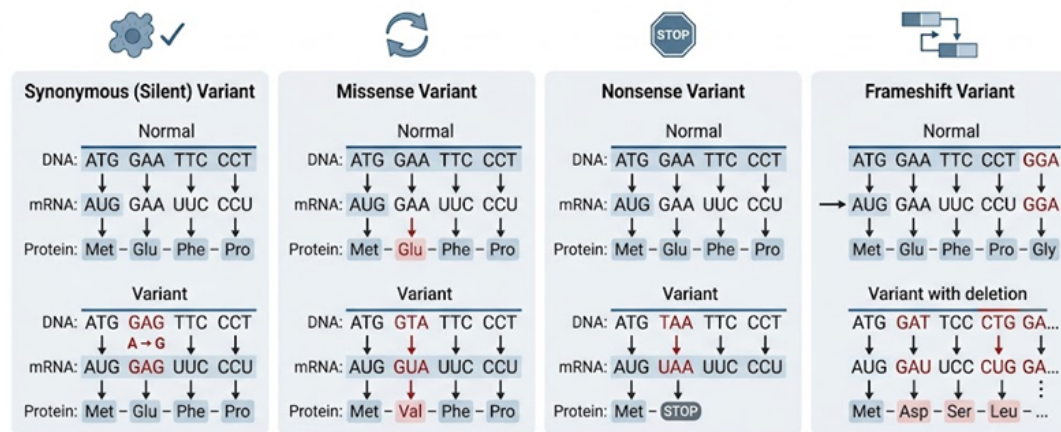


Figure 2.2: Major Types of Coding Genetic Variants and Their Effects on Protein Translation.

Germline variants arise in reproductive cells, pass to the next generation, and appear in every cell of the body. Somatic variants arise during a person's lifetime, often from DNA replication errors or exposure to environmental factors. This thesis focuses on germline missense variants because they underlie inherited genetic disease. Their effects vary widely: one substitution may cause partial loss of function, another may disrupt a modification site, and another may have no effect at all. This range makes the clinical impact hard to predict, which is what motivates a machine learning approach [21].

The American College of Medical Genetics and Genomics (ACMG) and the Association for Molecular Pathology (AMP) published guidelines in 2015 for classifying variants into five categories: Pathogenic (P), Likely Pathogenic (LP), VUS, Likely Benign (LB), and Benign (B) [12]. These categories are determined by the strength of clinical and scientific evidence, with P and LP variants having a high probability (>90%) of being disease-causing, while B and LB variants are considered harmless or likely harmless. The VUS category is a critical but ambiguous middle ground, where current evidence is not enough for a clinical decision. In this study, VUS variants were excluded, which leaves four labelled categories: Benign, Likely Benign, Likely Pathogenic and Pathogenic. The models

are trained on these four. For reporting, they are also grouped into two classes, with Benign and Likely Benign merged into benign and Likely Pathogenic and Pathogenic merged into pathogenic, because that is the split clinical practice acts on. Excluding VUS keeps only variants with clear clinical outcomes, which reduces noise from ambiguous labels during training.

2.2 Clinical Variant and Protein Databases

2.2.1 ClinVar

ClinVar is a free public database designed by the NCBI that collects and shares information on genetic variants. This information is provided by clinical laboratories [3]. ClinVar links DNA and amino acid variants to their position, gene name, and germline classification. The database classifies variants into five categories based on the ACMG/AMP guidelines [12].

ClinVar is a widely used resource for determining whether a variant is pathogenic or benign. ClinVar also provides information about diseases linked to variants. However, this study has some limitations. Certain genes are underrepresented, and rare variants may not have sufficient evidence. Submissions from different laboratories can introduce conflicting interpretations [3].

Variation	Gene (Protein Change)	Type (Consequence)	Condition	Classification, Review status
☐ NM_001005484.2(OR4F5):c.107 A>G (p.Glu36Gly)	OR4F5 (E36G)	Single nucleotide variant (missense variant)	not specified	G Likely benign ★
☐ NM_001005484.2(OR4F5):c.214 C>T (p.Pro72Ser)	OR4F5 (P72S)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.281 A>G (p.Lys94Arg)	OR4F5 (K94R)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.287 T>G (p.Ile96Ser)	OR4F5 (I96S)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.377 T>C (p.Met126Thr)	OR4F5 (M126T)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.389 T>C (p.Ile130Thr)	OR4F5 (I130T)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.396 G>A (p.Met132Ile)	OR4F5 (M132I)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★
☐ NM_001005484.2(OR4F5):c.479 C>G (p.Ala160Gly)	OR4F5 (A160G)	Single nucleotide variant (missense variant)	not specified	G Uncertain significance ★

Figure 2.3: Screenshot for NCBI ClinVar data.

2.2.2 UniProt

The Universal Protein Resource (UniProt) is a widely used database for protein sequences and functional information. It provides curated protein sequences and functional information [22]. Each UniProt entry is identified by a unique accession

number and a human-readable entry name that encodes the protein name and organism. UniProt was used as a reference for mapping ClinVar variants to their corresponding protein positions in DescribePROT.

2.2.3 DescribePROT

The DescribePROT database provides structural and functional predictions for proteins from 273 reference proteomes [10, 23]. Version 2.0 covers 19 descriptors generated by 11 predictive tools, including ASAquick, DFLpred, DisoRDPbind, DR-NApred, fIDPnn, MMseqs2, MoRFchiBi, MusiteDeep, PSIPRED, and SCRIBER, categorised as 4 structural, 13 functional, and 2 sequence-based. For Homo sapiens, each protein entry includes the full amino acid sequence, sequence length, and amino acid feature values covering secondary structure, intrinsic disorder, solvent accessibility, nucleic acid binding, protein-protein interaction, molecular recognition features (MoRFs), evolutionary conservation, and post-translational modification (PTM) sites [10].

The amino acid values are stored as strings in which the values are separated by commas, where each value corresponds to a specific amino acid position in the protein sequence. Figure 2.4 shows a sample record for the NOC2L protein. Most of these values are computational predictions generated by each tool from the protein sequence. The database also provides experimental annotations for three descriptors, solvent accessibility, secondary structure, and intrinsic disorder, taken from PDB and DisProt, but these cover only the proteins whose structures have been solved [10]. This amino acid-level information is important for generating the spatial feature matrix and allows the extraction of local structural features centered on each mutation site. However, DescribePROT has several limitations. It describes the amino acids in the original protein sequence and does not directly predict the effects of specific variants [10].

Table 2.1 lists the ten prediction tools used in this study and the feature that each one contributes.

2.2.4 Population Frequency Databases

Allele frequency databases record how often each variant appears in the general population. Variants that cause serious disease are very rare. Benign variants are common, because they do not harm health or reproduction [24].

The Genome Aggregation Database (gnomAD) is one of the largest public resources. It holds allele frequencies from 141,456 individuals [13]. The 1000 Genomes Project sequenced 2,504 individuals from 26 populations across five ancestry groups [15]. The Exome Sequencing Project (ESP) sequenced the exomes

Field / Key	Stored Value (first 8 positions shown)
ACC	Q9Y3T9
ACC_entry	NOC2L_HUMAN
ProteinName	Nucleolar_complex_protein_2_homolog
seqlength	749
seq	MAAGSRK...
ASAquick_normscore	0.659, 0.540, 0.444, 0.327, 0.314, 0.309, 0.341, 0.356, ...
fIDPnn_score	0.42, 0.41, 0.36, 0.28, 0.22, 0.26, 0.25, 0.26, ...
DisoPROscore	0.711, 0.733, 0.769, 0.760, 0.754, 0.760, 0.766, 0.803, ...
DisoDNAscore	0.350, 0.334, 0.243, 0.236, 0.217, 0.214, 0.201, 0.160, ...
DisoRNAscore	0.059, 0.061, 0.056, 0.058, 0.050, 0.052, 0.053, 0.052, ...
DRNAPredDNAscore	0.129, 0.092, 0.141, 0.102, 0.161, 0.235, 0.301, 0.279, ...
DRNAPredRNAscore	0.072, 0.055, 0.060, 0.056, 0.067, 0.071, 0.069, 0.065, ...
MoRFchiBiScore	0.810, 0.801, 0.801, 0.795, 0.792, 0.791, 0.798, 0.799, ...
SCRIBERscore	0.264, 0.212, 0.162, 0.146, 0.152, 0.164, 0.187, 0.129, ...
DfLpredScore	0.046, 0.053, 0.046, 0.045, 0.045, 0.041, 0.045, 0.045, ...
MMseq2_conservation_score	3.68, 2.58, 2.21, 2.52, 2.09, 2.29, 1.83, 2.45, ...
MMseq2_conservation_level	9, 5, 3, 4, 2, 3, 1, 4, ...
PSIPRED_helix	0, 14, 19, 19, 25, 37, 47, 51, ...
PSIPRED_strand	0, 1, 1, 1, 1, 1, 1, 1, ...
PTMbinary	00000000...
MMseqs2_pssm	Pos Cns A C D E F G H I K L M N P Q R S T V W Y, 0 M -3 -1 ...

Figure 2.4: Sample of amino acid features and PTM sites across the NOC2L protein sequence.

Table 2.1: The ten DescribePROT prediction tools used in this study and the amino acid features they predict.

Tool	What it predicts	Feature type
PSIPRED	Secondary structure: α -helix, β -strand, and coil propensity per residue	Structural
ASAquick	Relative solvent accessibility, how exposed each residue is to solvent	Structural
fIDPnn	Per-residue intrinsic disorder probability	Structural / disorder
DfLpred	Disordered flexible linker regions inside intrinsically disordered regions	Structural / disorder
DisoRDPbind	Protein-, DNA-, and RNA-binding residues located in disordered regions	Functional
DRNAPred	DNA- and RNA-binding residues	Functional
SCRIBER	Protein-protein interaction binding residues	Functional
MoRFchiBi	Molecular recognition features: short disordered segments that fold upon binding to a partner	Functional
MusiteDeep	Post-translational modification (PTM) sites	Functional
MMseqs2	Sequence homology search used to derive the PSSM and per-residue evolutionary conservation	Sequence-based

of about 6,503 individuals [25, 26]. The Exome Aggregation Consortium (ExAC) collected exome data from about 60,706 individuals [27]. Trans-Omics for Precision Medicine (TOPMED) provides sequence data from 53,831 individuals [14].

2.3 Protein Structure and Function Features

2.3.1 Primary and Secondary Structure

Proteins can be represented by primary, secondary, and tertiary structures. The primary structure is the amino acid sequence of the protein. Proteins fold into three main secondary structural elements: alpha helices, beta strands, and coils [28]. In this study, the PSIPRED secondary structure was extracted from the DescribePROT database [10]. Figure 2.5 shows the primary and secondary structure of a protein.

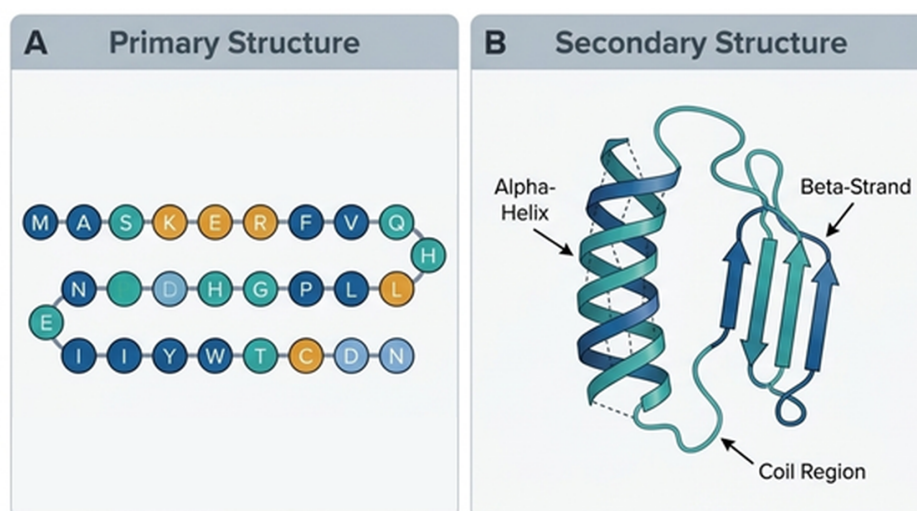


Figure 2.5: Primary and secondary structure of a protein.

2.3.2 Intrinsic Disorder

Some parts of a protein do not fold into a fixed shape. They stay flexible, and are known as Intrinsically Disordered Regions (IDRs). Many of these regions become structured only when they bind to a partner molecule [29]. Figure 2.6 shows a disordered region inside a protein.

Two disorder features were included in this study. The intrinsic Disorder Predictor (fIDPnn) score gives each residue a probability of being disordered [30]. The Disordered Flexible Linker (DFLpred) score marks flexible linker regions, which let parts of a protein move and change shape [31].

Disorder matters for pathogenicity prediction for two reasons. First, about 30–40% of the human proteome lies in disordered regions [29]. Second, a variant

in a disordered region acts differently from one in a structured region. It usually disturbs a binding site instead of the stability of the fold [32].

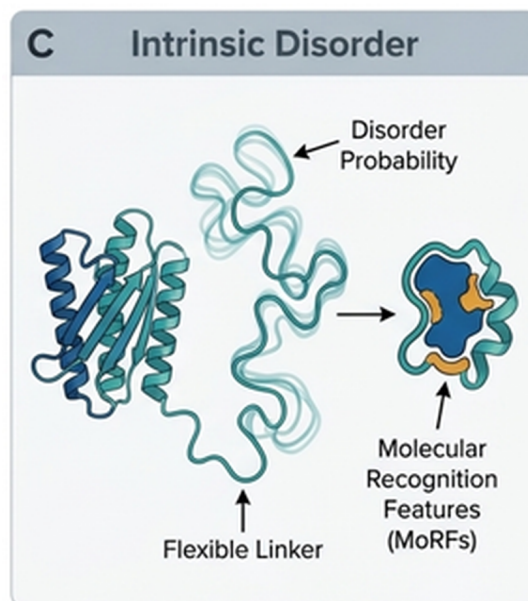


Figure 2.6: Intrinsic disorder in a protein.

2.3.3 Solvent Accessibility

Solvent accessibility measures how much each amino acid is exposed to the water around the protein. Amino acids buried deep in the hydrophobic core have almost no solvent accessibility, and they hold the protein structure together [33].

Variants at these buried positions are more likely to disturb the fold and cause disease. About 67% of residues that carry disease-related variants are buried, while most neutral variants sit at exposed positions [34]. Surface amino acids have high solvent accessibility, because they are not part of the hydrophobic core. They can also bind other molecules, such as DNA or other proteins, so a variant there may break a functional interaction instead of the stability of the fold [35]. In this study, solvent accessibility was taken from the Accessible Surface Area (ASAquick) predictor [36].

2.3.4 Molecular Binding and Interaction Sites

Proteins interact with other molecules, such as proteins, DNA, and RNA. Amino acids at binding sites are often highly conserved, because a variant there can break the interaction and disturb the biological process [37]. Seven features describe binding sites in this study, as shown in Table 2.2.

MoRFchibi predicts molecular recognition features (MoRFs), which are short disordered segments that fold when they bind to a protein partner [38]. Several of

these seven features describe the same property using different predictors. This overlap is deliberate. When one predictor misses a binding site, another may still detect it, so the model is less likely to overlook a functionally important position.

2.3.5 Post-Translational Modifications

Amino acids in a protein undergo chemical changes after the protein is made. These changes are called PTMs, and they control protein activity, stability, and binding to other molecules [39]. DescribePROT reports nine PTM types: phosphorylation, ubiquitination, glycosylation, acetylation, methylation, SUMOylation, palmitoylation, hydroxylation, and pyrrolidone carboxylic acid [10].

A variant at a PTM site stops the modification from being added. A variant near a PTM site can also matter, because it may change the short sequence that the modifying enzyme recognises, even when the modified residue itself is unchanged [39].

2.3.6 Evolutionary Conservation

Evolutionary conservation measures how well a position in a protein sequence has been kept across different species. Positions that matter for protein function stay conserved, because mutations there are harmful. Positions that accept substitutions vary between species, which suggests that changes there have little effect on function [16].

A position-specific scoring matrix (PSSM) gives a different score to each amino acid at every position in a protein sequence. It is built from a multiple sequence alignment (MSA) of related proteins. An amino acid that appears at the same position in many species receives a higher score. A positive score means the amino acid is preferred at that position, and a negative score means it is disfavoured [40].

Many-against-Many sequence searching (MMseqs2) produces both a full PSSM and a per-residue conservation score. The conservation score summarises the conservation at one position as a single value between 0 and 4.34 bits, and it is used as a matrix feature. The full PSSM is used separately, to read the score of the mutant amino acid at the variant position, which is stored as the PSSM score in the dense feature vector [41].

Unlike PSSM scores, BLOSUM62 gives the same substitution score wherever the position is in the protein. It counts how often each pair of amino acids replaces the other in related proteins, and compares that with how often the pair would meet by chance. Amino acids with similar chemical properties score higher, because they replace each other more often [16]. Figure 2.7 shows an example position-specific scoring matrix.

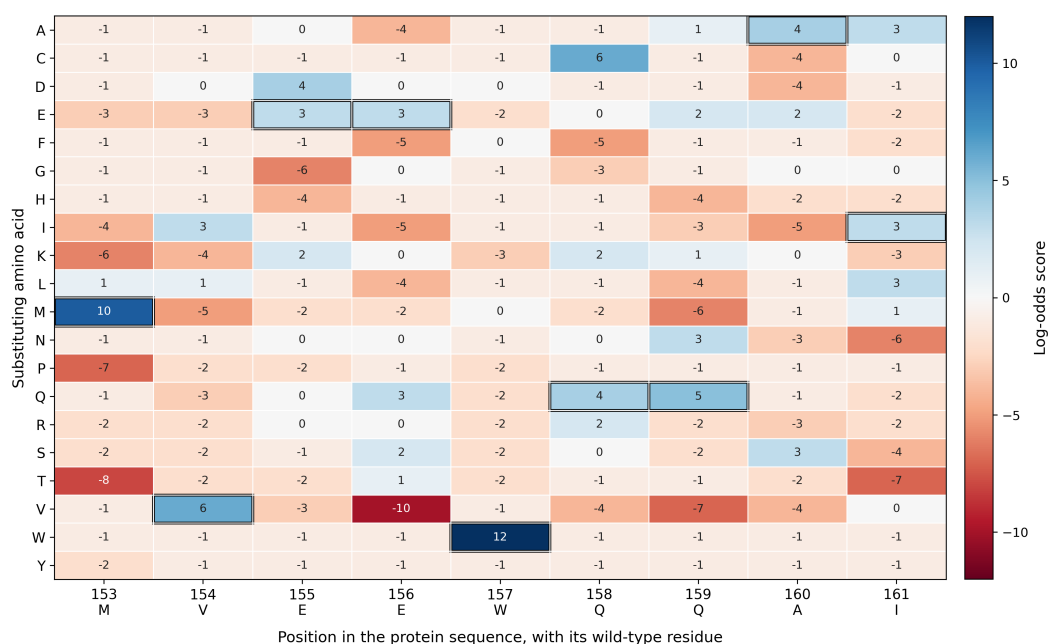


Figure 2.7: PSSM from DescribePROT for positions 153–161 of the NOC2L protein. The outline marks the wild-type residue.

2.3.7 Physicochemical Properties of Amino Acids

The 20 standard amino acids differ in molecular weight, hydrophobicity, charge, polarity, and size [42]. When one amino acid replaces another, the size of the physicochemical difference between them affects how the protein works [43].

Three delta features describe these differences. Delta mass is the difference in molecular weight, in Daltons. Delta hydrophathy is the change in how strongly a residue prefers the hydrophobic core over the surface, measured on the Kyte-Doolittle scale [44]. Delta charge is the difference in net charge at pH 7.0. A large absolute delta means a large physicochemical change, which is more likely to affect protein function [43].

2.4 Machine Learning

2.4.1 Supervised Classification

Machine learning is a part of artificial intelligence that lets a system learn patterns from data instead of following rules written by hand. In supervised learning, the model is trained on labelled data, where each sample has input features and a known correct label. The model learns how the features map to the label. After training, it can predict labels for new samples it has not seen [45].

Classification is a supervised learning task that puts each sample into one of a set of predefined categories [46]. In this thesis, the task is to decide whether a

Table 2.2: The window matrix channels (31×15).

Feature Name	Category	Description	Source
<i>Amino Acid Identity</i>			
AA_token	Amino Acid	Integer identity of the residue (1–20, A–Y; 0 = padding/unknown), a single channel	Computed
<i>Primary and Secondary Structure</i>			
PSIPRED_strand	Secondary Structure	Beta-strand confidence score (0–1, normalized from 0–100)	DescribePROT
PSIPRED_helix	Secondary Structure	Alpha-helix confidence score (0–1, normalized from 0–100)	DescribePROT
<i>Intrinsic Disorder</i>			
fIDPnn_score	Intrinsic Disorder	intrinsic disorder probability	DescribePROT
DFLpredScore	Intrinsic Disorder	Disordered flexible linker	DescribePROT
<i>Solvent Accessibility</i>			
ASAquick_normscore	Solvent Accessibility	Normalized relative solvent accessibility (0–1)	DescribePROT
<i>Molecular Binding and Interaction Sites</i>			
DisoDNAScore	Binding Sites	DNA-binding probability in disordered regions	DescribePROT
DisoPROScore	Binding Sites	Protein-binding probability in disordered regions	DescribePROT
DisoRNAScore	Binding Sites	RNA-binding probability in disordered regions	DescribePROT
DRNAPredDNAScore	Binding Sites	DNA-binding probability from DRNAPred	DescribePROT
DRNAPredRNAScore	Binding Sites	RNA-binding probability from DRNAPred	DescribePROT
SCRIBERscore	Binding Sites	Protein-protein interaction binding	DescribePROT
MoRFchibiScore	Binding Sites	Probability score for short disordered segments that fold upon binding to a partner protein	DescribePROT
<i>Post-Translational Modifications</i>			
PTMbinary	PTM	Binary flag: 1 if any PTM site is present at the residue position (9 PTM types from MusiteDeep), 0 otherwise	DescribePROT
<i>Evolutionary Conservation</i>			
MMseq2_conservation_score	Evolutionary Conservation	Per-residue evolutionary conservation score (normalized to 0–1 from 0–4.34 bits)	DescribePROT

Table 2.3: The dense vector features.

Feature Name	Category	Description	Source
log_Composite_AF	Allele Frequency	Log ₁₀ of the composite allele frequency across the population databases	Engineered
n_databases_seen	Allele Frequency	Number of population databases that report the variant	Engineered
Delta_mass	Physicochemical	Molecular weight difference between variant and reference amino acid	Biopython Protein-Analysis
Delta_hydrophathy	Physicochemical	Kyte-Doolittle hydrophobicity difference between variant and reference amino acid	Biopython Prot-ParamData
Delta_charge	Physicochemical	Net electric charge difference at physiological pH 7.0	Biopython Protein-Analysis
BLOSUM_score	Evolutionary Conservation	BLOSUM62 log-odds substitution score for the specific amino acid variant	Biopython BLOSUM62 matrix
pssm_score	Evolutionary Conservation	Position-Specific Scoring Matrix log-likelihood for the variant amino acid	DescribePROT
PTM_Binary_Sum	PTM	1 if any PTM type is annotated at the variant position, otherwise 0	Derived from DescribePROT

variant is pathogenic or benign, using its biological features. The input is the 31×15 matrix together with an eight-dimensional vector.

2.5 Neural Networks and Deep Learning

2.5.1 Multilayer Perceptron

An artificial neural network is a machine learning model inspired by biological neural systems [47]. A perceptron takes n inputs $x_1, x_2, \dots, x_n$, multiplies each input x_i by its own learnable weight w_i , adds a bias term b , and passes the result through an activation function f , which gives the output y :

$$y = f \left(\sum_{i=1}^n w_i x_i + b \right)$$

Multilayer Perceptron (MLP) is built by stacking perceptrons into fully connected layers, where every neuron in one layer connects to every neuron in the next layer. A layer with n inputs and m outputs holds $n \times m$ weights and m bias terms. In this study, MLP layers process the eight-dimensional dense feature vector [48]. Figure 2.8 shows the structure of a multilayer perceptron.

2.5.2 Loss Functions and Optimization

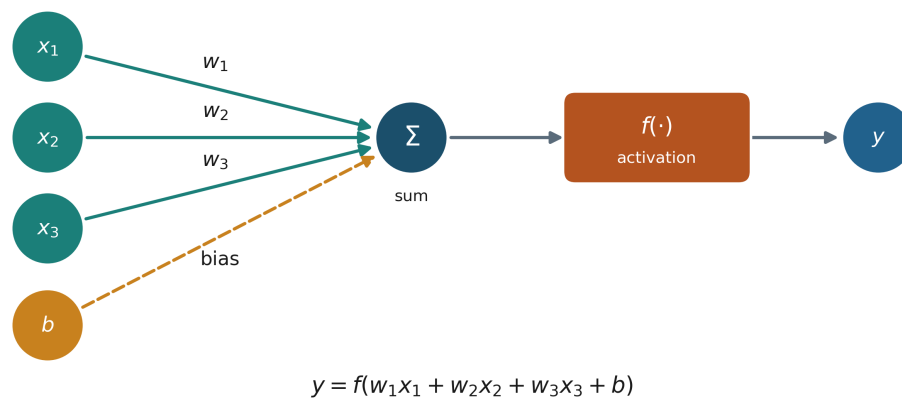
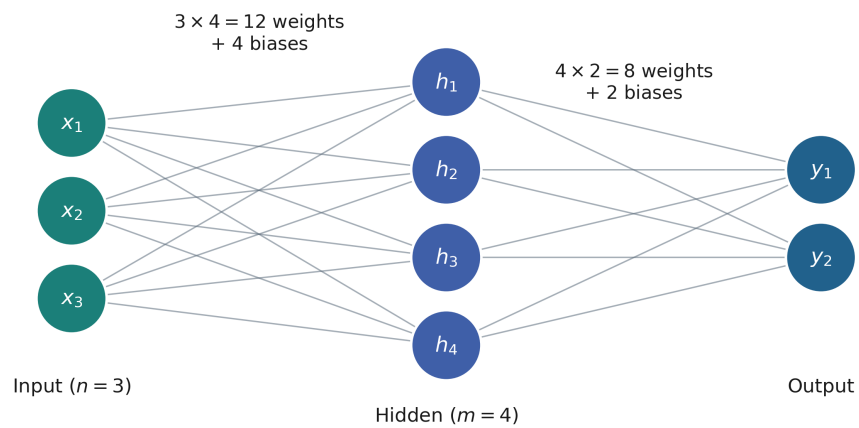
A neural network learns by minimizing a loss function, which measures the gap between its predicted probabilities and the true labels. For classification with a softmax output, the standard loss is categorical cross-entropy:

$$L = - \sum_{k=1}^K y_k \log(\hat{y}_k)$$

where L is the loss for one sample, K is the number of classes, y_k is 1 for the true class and 0 for the others, and $\hat{y}_k$ is the predicted probability of class k . When labels are stored as integers instead of one-hot vectors, the same quantity is computed as sparse categorical cross-entropy. This study uses the sparse form, because each variant carries a single integer class label [46].

An optimizer updates the network weights to reduce this loss. Gradient descent moves each weight in the direction that lowers the loss, by a step size set by the learning rate [47]. Adam and Nadam are adaptive optimizers that keep a separate step size for each weight, tuned automatically from running estimates of the gradient, so they usually converge faster than plain gradient descent on deep models [47]. This study tuned between Adam and Nadam.

When the classes are imbalanced, the frequent classes contribute most of the

Perceptron (single neuron)**Multilayer perceptron, fully connected****Figure 2.8:** A perceptron and a fully connected multilayer perceptron.

summed loss, so the optimizer favors them and neglects the rare ones. Class weighting counters this by multiplying each sample's loss by a weight inversely proportional to its class frequency, so a rare Pathogenic variant counts as much as several common Benign ones [49].

2.5.3 Regularization and Overfitting

A model overfits when it learns patterns specific to the training data that do not generalize, so training accuracy keeps rising while validation accuracy stalls or falls. The gap between training and validation performance measures the degree of overfitting [47]. This study monitors that gap directly and retrain with stronger regularization when it grows too large.

Several techniques limit overfitting. Dropout randomly deactivates a fraction of neurons during each training step, which stops the network from relying on any single feature and pushes it to use several features together [47]. L2 regularization adds a penalty proportional to the squared magnitude of the weights, discouraging large weights that fit noise [47]. Layer normalization rescales the activations within each layer to a stable range, which smooths training and reduces sensitivity to the scale of the inputs [50].

Training is also shaped by callbacks that check the validation metric after every epoch. Early stopping halts training once the validation score stops improving and restores the best weights. Learning-rate reduction lowers the step size when the metric plateaus, and checkpointing saves the best model seen so far [50].

2.6 Convolutional Neural Networks (CNN)

2.6.1 Convolution, Filters, and Feature Maps

A CNN is a deep learning model designed to process images. Instead of connecting every input value to every neuron, as in an MLP, a CNN uses kernels that slide across the input matrix to pick up local patterns [51]. This operation is called convolution:

$$(I * K)(i, j) = \sum_u \sum_v I_{i+u, j+v} K_{u,v}$$

where I is the input matrix, K is the filter, also called the kernel, i and j give the position in the output, and u and v run over the rows and columns of the kernel. The result is a feature map that highlights the pattern the kernel responds to [47].

2.6.2 Stride and Padding

The stride is the number of positions the filter moves after each step. A stride of 1 moves the filter one position at a time, so every location in the input is examined. A stride of 2 moves it two positions at a time, which makes the feature map smaller [47].

The output size of a convolutional layer depends on the input size n , the filter size f , the stride s , and the padding p :

$$\text{output size} = \left\lfloor \frac{(n + 2p - f)}{s} \right\rfloor + 1$$

Padding adds extra values around the border of the input, so the kernel can also cover the edge positions, and it controls the output size after the convolution [50]. Without padding the output is smaller, because the kernel cannot fully overlap the positions near the border [47].

The models in this study were built with Keras, a Python library for deep learning, in which a convolutional layer is called `Conv2D`. Every convolutional layer in this study used the Keras setting `padding='same'`, which adds zeros around the border so that each convolution keeps the same height and width as its input. The size is reduced by the pooling layers instead, while the number of feature maps grows as more filters are applied in each block.

2.6.3 Activation Functions

An activation function introduces nonlinearity into a neural network. This lets the network learn complex patterns instead of simple linear relationships [47].

The Rectified Linear Unit (ReLU) is widely used in deep learning models. It returns zero for negative inputs and passes positive inputs through unchanged [47]:

$$f(x) = \max(0, x)$$

Swish is a smoother alternative to ReLU [52]. Unlike ReLU, it lets small negative values pass through, which can help the gradient flow in deep networks:

$$f(x) = x \cdot \sigma(x)$$

where σ is the sigmoid function. In this study, the hyperparameter search chose between ReLU and Swish for the hidden layers.

Softmax is used in the final output layer. It turns the raw output scores into probabilities, by exponentiating each score and dividing by the sum of all exponentiated scores [47]:

$$P(\text{class } k) = \frac{e^{z_k}}{\sum_j e^{z_j}}$$

where z_k is the raw score for class k , and the sum in the denominator runs over all classes j . The output is a vector of non-negative values that sum to 1, where each value is the predicted probability of one class. In this study, the softmax output layer has two neurons, producing probabilities for Benign and Pathogenic [46]. Figure 2.9 shows the ReLU and Swish activation functions, and a softmax example.

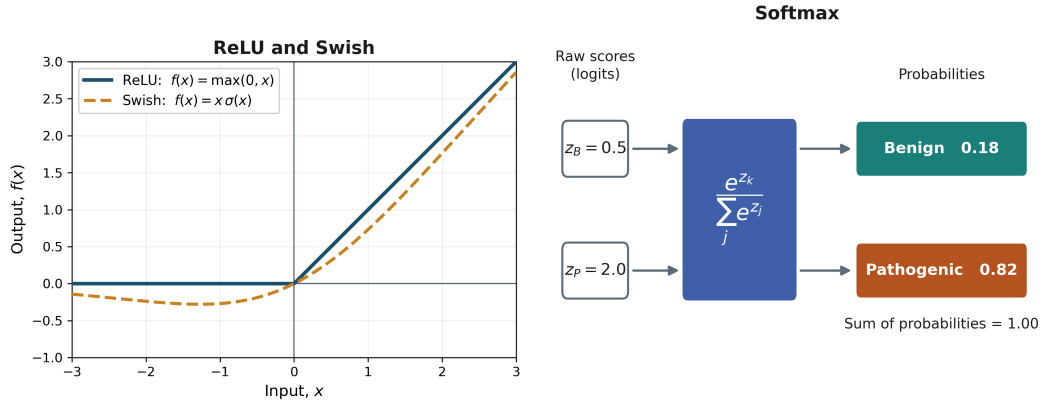


Figure 2.9: ReLU and Swish activation functions, and a softmax example.

2.6.4 Pooling Operations

After convolution and activation, a pooling layer reduces the size of the feature maps, which lowers the computational cost of the later layers [47]. Average pooling replaces each pooling region with its average value. Max pooling keeps only the largest value [53]. Global pooling goes further and reduces each feature map to a single number: global average pooling takes its mean, and global max pooling takes its largest activation [54]. This is applied to each feature map on its own, so the layer produces one value per feature map rather than one value in total. In this study the DenseNet head concatenates both, so every feature map contributes its average and its peak response.

2.6.5 From Images to Protein Sequences

A CNN is not limited to photographs. Any two-dimensional matrix can be treated as a single-channel image, so a convolutional filter slides across it and finds local patterns in the same way it finds edges in a picture. This makes CNNs useful for structured biological data, such as a matrix of per-residue protein features. In the matrix used here, neighbouring rows are neighbouring positions in the protein

sequence, so a filter that covers several rows reads a short stretch of the sequence at once.

2.6.6 Multimodal Fusion

A single network branch processes one kind of input. To combine two different inputs, such as the protein window matrix and the eight variant-level features, each input gets its own branch: a CNN branch for the matrix and an MLP branch for the vector. Each branch produces a feature vector, and the two are joined into one longer vector by concatenation. A dense fusion layer then learns how the two sources interact, before the softmax output [47]. This lets the model weigh structural context and variant-level evidence together, and it is the basis of the hybrid architectures evaluated in this study.

2.7 Advanced CNN Architectures

2.7.1 VGG

Simonyan and Zisserman proposed VGG and showed that depth matters for CNN performance. Instead of large filters, VGG stacks several small 3×3 filters one after another. Two stacked 3×3 layers cover the same receptive field as one 5×5 layer, but they use fewer parameters and add one more nonlinearity between them [55].

A VGG network is built from convolutional blocks. Each block applies two or more 3×3 convolutions and ends with a max-pooling layer that halves the spatial size. The number of filters doubles from one block to the next, so the feature maps get smaller while the number of features grows. This study uses that repeating block pattern, not a fixed configuration such as VGG16 or VGG19. The baseline and hybrid models use a stack of 3×3 convolutional blocks, sized to the small 31×15 protein matrix.

2.7.2 DenseNet

Huang et al. proposed DenseNet, short for Densely Connected Convolutional Networks, to improve how data flows through deep networks. In a traditional CNN, each layer receives input only from the layer before it, which makes it hard to carry information through a deep network. DenseNet connects the layers densely instead. Each layer in a dense block receives the feature maps of all earlier layers, and passes its own feature maps to all later layers in the same block. This lets the network reuse features and helps the gradient flow during training [56].

The dense block is the main component of the architecture. Suppose the input to the block has C_0 feature maps, and each layer produces k new feature maps, where k is called the growth rate. A dense block with L layers then outputs $C_0 + Lk$ feature maps [56]. Figure 2.10 shows a dense block with five convolutions.

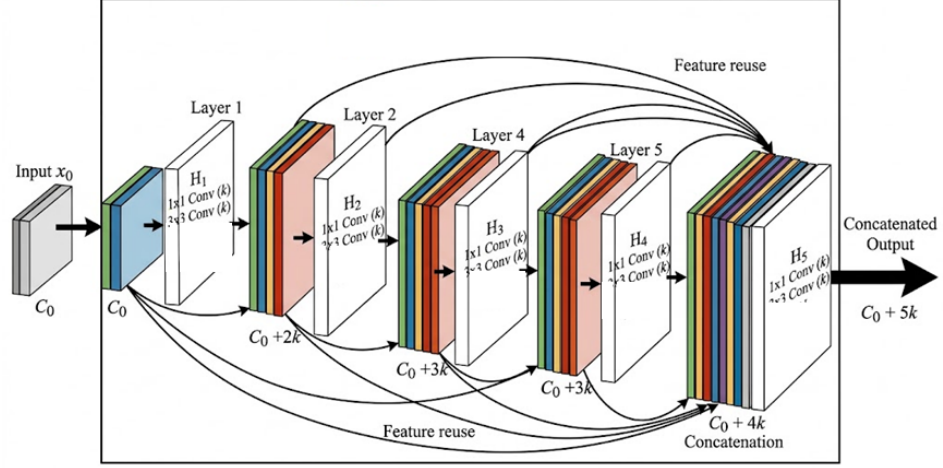


Figure 2.10: DenseNet dense block with five convolutional layers.

Inside each dense block, every layer is a bottleneck layer. It applies BatchNorm, then ReLU, then a 1×1 convolution with $4k$ filters, then BatchNorm, ReLU, and a 3×3 convolution with k filters [56]. The 1×1 convolution reduces the number of channels before the more expensive 3×3 convolution. The output of each bottleneck layer is concatenated with all previous feature maps and passed to the next layer in the same block [56].

Between dense blocks, DenseNet uses a transition layer. It applies a 1×1 convolution that reduces the number of channels by a compression factor θ , followed by average pooling that reduces the height and width of the feature maps [56]. If the input has C channels, the compressed output has

$$C_{\text{compressed}} = \theta C, \quad 0 < \theta \leq 1$$

channels, so a smaller θ removes more channels and $\theta = 1$ keeps them all.

After the final dense block, global pooling turns each feature map into a single value, which gives a one-dimensional feature vector instead of flattening the whole map [54]. The model in this study concatenates global average pooling and global max pooling, so each feature map contributes both its mean and its largest activation.

2.8 DeepInsight

The DeepInsight method converts tabular data into an image so that a CNN can be applied to it. In a natural image, nearby pixels are related and belong to the same object, and convolutional filters rely on this when they detect edges and textures. Tabular features have no such arrangement. In the 31×15 protein matrix the order of the feature columns is arbitrary, so biologically related features can sit far apart. DeepInsight rearranges the features into a 2D layout where similar features become neighbouring pixels, so the convolution has local structure to read [17].

DeepInsight builds this layout with t-distributed stochastic neighbor embedding (t-SNE), a nonlinear dimensionality reduction method that projects high-dimensional data into a low-dimensional space while keeping neighbourhood relationships [57]. The 31×15 matrix is flattened into 465 features, and each feature is treated as a point described by its values across all N variants. t-SNE gives each feature a position on a 2D grid, so features with similar value patterns land close together. It does this by minimising the Kullback-Leibler (KL) divergence between the pairwise similarities in the original high-dimensional space, p_{ij} , and those in the 2D space, q_{ij} :

$$KL = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

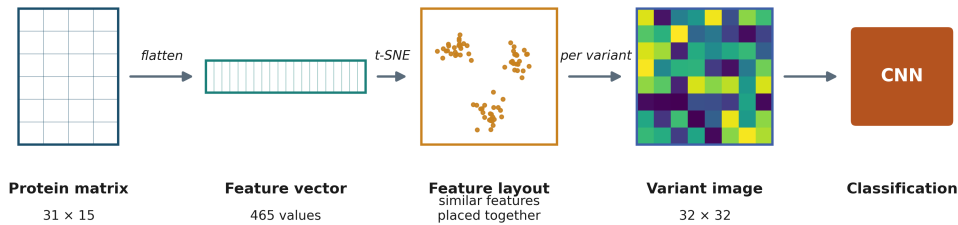


Figure 2.11: DeepInsight pipeline for tabular-to-image conversion.

Figure 2.11 summarises the full pipeline. The t-SNE layout is learned once and reused for every sample: each variant becomes a different image because it has different feature values, not because the feature positions change [17].

2.9 Hyperparameter Optimization

A neural network has two kinds of settings. Parameters are the weights and biases the network learns from the data during training. Hyperparameters are chosen before training and control how the network is built and how it learns, such as the number of layers, the number of filters per layer, the learning rate, and the

dropout rate [47]. Gradient descent tunes the parameters automatically, but not the hyperparameters. Those must be chosen separately, and the choice strongly affects performance. Searching for the best combination is called hyperparameter optimization.

Trying combinations by hand is slow and rarely finds the best setting. Keras Tuner automates this search [58]. The developer defines a search space, which is the range of allowed values for each hyperparameter, and a metric to optimize. Keras Tuner then builds and trains many versions of the model, each with a different combination from that space, and keeps the one that scores best.

An epoch is one full pass over the training data, during which the model updates its weights after each mini-batch using backpropagation [47]. Training to convergence takes many epochs, so training every candidate to the end would be far too slow. Hyperband, the search algorithm used in this study, avoids that cost [59]. It starts many candidates and trains each for only a few epochs. It then ranks them, drops the weakest, and gives the survivors more epochs. Repeating this round after round is called successive halving, and it spends the training budget on the candidates that already look promising. The one that survives the final round is selected.

This makes Hyperband more efficient than the classical alternatives. Grid search trains a model for every combination in the space, which becomes impossibly slow as the number of hyperparameters grows. Random search samples combinations at random but still trains each one for the full number of epochs, spending as much effort on poor configurations as on good ones. Hyperband instead concentrates its budget on the strongest candidates [59].

2.10 Evaluation Metrics

This study evaluates the models using values from the confusion matrix [60]. True positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) are counted per class with the one-vs-rest method. For each class i , that class is the positive class, and all the other classes together form the negative class.

- TP_i : Variants of class i that the model correctly predicted as class i .
- FP_i : Variants from other classes that the model incorrectly predicted as class i .
- FN_i : Variants of class i that the model incorrectly predicted as any other class.

- TN_i : Variants from other classes that the model correctly did not predict as class i .

Accuracy is the proportion of correctly classified samples.

$$\text{Accuracy} = \frac{\sum_{i=1}^K TP_i}{N}$$

where K is the number of classes and N is the total number of samples.

Recall, also called sensitivity, is the share of positive samples that the model identifies correctly.

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i}$$

Specificity, also called the true negative rate, is the share of negative samples that the model identifies correctly.

$$\text{Specificity}_i = \frac{TN_i}{TN_i + FP_i}$$

Precision is the share of correct predictions among all samples the model predicted as positive.

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i}$$

The F1-score is the harmonic mean of precision and recall for one class.

$$F1_i = 2 \times \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$$

Macro F1 takes the F1-score of each class and averages them, giving every class equal weight.

$$\text{Macro F1} = \frac{1}{K} \sum_{i=1}^K F1_i$$

where K is the number of classes and $F1_i$ is the F1-score of class i .

The area under the receiver operating characteristic curve (ROC-AUC) measures how well the model separates the positive class from the negative class across all decision thresholds. The ROC curve plots the true positive rate against the false positive rate as the threshold changes. The macro-averaged ROC-AUC uses the same one-vs-rest strategy: it computes one AUC for each class and averages them, giving every class equal weight.

Chapter 3

Literature Review

This chapter reviews the computational tools that predict whether a missense variant is pathogenic. The tools are grouped by approach. The first group covers traditional and machine learning methods, from early conservation-based tools such as SIFT to ensemble predictors such as REVEL. The second group covers deep learning methods, including convolutional neural networks, generative models, and protein language models. The chapter then describes DeepInsight, the method this thesis builds on, and ends by identifying the gap that this work addresses.

3.1 Traditional and Machine Learning Approaches

3.1.1 SIFT

Sorting Intolerant from Tolerant (SIFT) was one of the earliest tools for predicting whether an amino acid variant affects protein function. SIFT measures evolutionary conservation from multiple sequence alignments and estimates how well each substitution is tolerated at a given position [6]. A SIFT score below 0.05 indicates that a substitution is likely to be damaging, whereas a score above 0.05 suggests that the substitution is tolerated [6].

A later version, SIFT 4G, made the method much faster and extended precomputed predictions to whole genomes across more than 200 organisms [61].

The SIFT tool has limitations. Because SIFT relies primarily on evolutionary conservation, it does not explicitly incorporate protein structural or functional information [6]. It also tends to produce false-positive predictions, classifying some benign variants as harmful [62].

This tool is widely used in clinical genetics, and it is one of the computational predictors listed in the guidelines for interpreting genetic variants [12]. The conservation information used by SIFT is conceptually related to the PSSM feature employed in this thesis [10]. The PSSM scores are obtained from DescribePROT and are included as part of the input feature set [10].

3.1.2 PolyPhen-2

Polymorphism Phenotyping version 2 (PolyPhen-2) builds on SIFT by incorporating both sequence and structure features [7]. PolyPhen-2 uses a Naïve Bayes classifier trained on eight sequence features, including evolutionary conservation, and three structure features [7]. It provides two trained models: HumDiv, trained on damaging mutations and differences between closely related species, and HumVar, trained on human disease mutations and common human variants [7].

The tool outputs a score between 0 (benign) and 1 (pathogenic) [7]. Comparisons between such tools on benchmark datasets should be read with care, because overlap between the training data and the evaluation data can inflate the measured performance of some tools [62].

3.1.3 MutationTaster

MutationTaster uses a Naïve Bayes classifier trained on features taken from NCBI resources [63].

3.1.4 FATHMM

Functional Analysis through Hidden Markov Models (FATHMM) predicts the functional impact of a variant using hidden Markov models (HMMs). An HMM describes a protein family through match, insert, and delete states, so each position is weighted by how conserved it is instead of all positions counting equally [64]. FATHMM also uses protein domain information from the SUPERFAMILY and Pfam databases, and it can give extra weight to known human disease mutations [64].

3.1.5 CADD

Combined Annotation-Dependent Depletion (CADD) trains a support vector machine with a linear kernel to separate two sets of variants: 14.7 million observed human variants and 14.7 million simulated variants [65].

Learning to tell these two sets apart lets CADD rank variants by deleteriousness using many genomic annotations, instead of relying on one biological feature or on sequence alignment alone. Because those annotations cover the whole genome, CADD scores both coding and non-coding variants, unlike earlier tools built for coding variants only [65]. Its scores are precomputed for all 8.6 billion possible human single-nucleotide variants [65].

Each of the tools described so far produces a single score from its own features. Combining several predictors works better than any single one, which led to the

ensemble methods described next [66, 67].

3.1.6 REVEL

Rare Exome Variant Ensemble Learner (REVEL) uses a Random Forest classifier. It combines thirteen prediction tools as features: MutPred, FATHMM, VEST, PolyPhen-2, SIFT, PROVEAN, MutationAssessor, MutationTaster, LRT, GERP++, SiPhy, phyloP, and phastCons [66]. REVEL is aimed at rare variants with allele frequencies below 0.5%. It was trained on disease variants from the Human Gene Mutation Database (HGMD) and on neutral variants from the Exome Sequencing Project, the ARIC study, and the 1000 Genomes Project [66].

REVEL outputs a score between 0 and 1, where higher values indicate a higher likelihood of pathogenicity. REVEL achieved an overall AUC (ROC) of 0.908, estimated from out-of-bag predictions on its training set [66]. On an independent ClinVar test set, REVEL had the best performance among the ensemble methods tested [66].

3.1.7 BayesDel

BayesDel uses a Bayesian likelihood ratio framework to score how likely a variant is to be pathogenic [67]. Instead of training a standard classifier, it combines several deleteriousness predictors: PolyPhen-2, SIFT, FATHMM, LRT, MutationTaster, MutationAssessor, PhyloP, GERP++, and SiPhy, together with the population allele frequency. It merges them as a weighted product of likelihood ratios, so that correlated predictors are down-weighted [67].

The method produces a score on a log scale, where a threshold at 0 separates deleterious from neutral variants [67]. In the original evaluation it reached an AUC of 0.94, against 0.90 for MetaSVM, 0.88 for MetaLR, and 0.88 for CADD [67].

3.1.8 ClinPred

ClinPred combines two machine learning models, a random forest and a gradient boosted decision tree, trained on ClinVar variant classifications [68]. It uses the scores of 16 existing prediction tools as features, taken from database for Nonsynonymous SNPs' Functional Predictions (dbNSFP). Its main addition is the variant's allele frequency across populations, taken from the gnomAD database [68].

A general concern for supervised predictors is circularity [62]. Many tools are trained on ClinVar or HGMD variants and then evaluated on data drawn from the same databases. When this happens, the model may already have seen patterns from the test variants, so strong performance on these databases does

not guarantee strong performance on new variants. To reduce this bias, recent benchmarks evaluate predictors against experimental deep mutational scanning data instead of clinical labels [69].

3.2 Deep Learning Approaches

3.2.1 PrimateAI

PrimateAI applies a CNN to a one-hot encoded sequence window centred on the mutation site, so the network reads the local amino acid context [70]. Its main idea is to use common variants from six non-human primate species, such as chimpanzees and gorillas, as examples of variants that are likely benign in humans [70]. Pathogenicity is then learned from patterns of evolutionary conservation and sequence variation, so variants in conserved regions are more likely to be predicted as deleterious [70].

3.2.2 EVE

EVE (Evolutionary model of Variant Effect) is trained on multiple sequence alignments (MSAs) rather than on clinical labels [71]. It uses a Variational Autoencoder (VAE) to learn the pattern of sequence variation within a protein family, then scores a variant by how well it fits that learned pattern. Because it never sees clinical annotations, it avoids the biases carried by existing variant databases [71]. Its limitation is coverage: EVE can only score residues with enough alignment depth, about 84% of residues across roughly 3,000 disease-related genes [9].

3.2.3 ESM1b

Models based on evolutionary conservation need many related sequences from other species. When a human protein is unique or has few known relatives, these models become weak or cannot run at all. ESM1b works differently. It treats a protein sequence as a sentence and each amino acid as a word, and learns from a large set of protein sequences without building any alignment [72]. To score a variant, ESM1b needs only the wild-type human protein sequence. The model gives a probability for each of the 20 amino acids at every position. It then scores the variant by the Log-Likelihood Ratio (LLR) of the mutant amino acid against the original one at that position, so a negative score means the model finds the mutant less likely and the change more likely to be damaging [9]. On ClinVar, ESM1b reached a ROC-AUC of 0.905 separating 19,925 pathogenic from 16,612 benign variants, and on HGMD against gnomAD it reached 0.897 [9].

ESM1b also has a limit on input length. The cost of self-attention grows with the square of the sequence length, so the model accepts at most 1,022 amino acids. About 12% of human proteins are longer than this. They are handled by cutting the sequence into overlapping windows of 1,022 amino acids rather than being dropped [9].

3.2.4 AlphaMissense

AlphaMissense was released by Google DeepMind in 2023 to predict the effect of missense variants [8]. It starts from AlphaFold2, which predicts protein structure, but it is trained for a different job: judging whether an amino acid change is harmful. The model joins the structural context it takes from AlphaFold2 with conservation information from multiple sequence alignments, and returns a score between 0 and 1 for every possible single amino acid substitution in the human proteome [8]. Using a threshold set to give 90% precision on ClinVar, it called 32% of all possible human missense variants likely pathogenic and 57% likely benign, and left the remaining 11% uncertain [8].

An independent test on 61 clinically validated variants in the CHRN genes gave AlphaMissense an accuracy of about 64%. This beat the two other tools in that study, but the result came mostly from the pathogenic side. The model labelled fewer than half of the non-pathogenic variants correctly, so it tends to call variants pathogenic too often. The authors concluded that such tools need more refinement and clinical validation before routine diagnostic use [73].

3.2.5 Other CNN tools

One study classified variants in the BRCA1 gene with a CNN by turning amino acid sequences into two-dimensional images. It reported 91% accuracy and 93% sensitivity for the benign class, and an AUC of 92%, on BRCA1 variants from ClinVar. Although limited to a single gene, the study concluded that 2D CNNs can capture spatial patterns in amino acid representations [74].

DeepRank-Mut is a model that applies a 3D CNN architecture to analyze the three-dimensional environment around mutation sites and learn structural patterns. The dataset contained 193,714 missense variants from ClinVar, gnomAD, and Dutch diagnostic laboratories. It achieved an accuracy of 0.77 and an MCC of 0.52 [75]. Table 3.1 summarizes the algorithmic approaches of the pathogenicity predictors discussed in this section.

Table 3.1: Summary of pathogenicity predictors and their reported performance.

Predictor	Year	Model type	Input	Variant types	Biological domain	ROC-AUC
<i>Traditional and Machine Learning Approaches</i>						
SIFT [6]	2003	Statistical (conservation)	Protein sequence + MSA	Missense	Sequence conservation	0.69-0.75
PolyPhen-2 [7]	2010	Naïve Bayes	Sequence + MSA + 3D structure	Missense	Conservation + structure	0.75-0.85
MutationTaster [63]	2010	Naïve Bayes	Sequence + NCBI/Ensembl	SNVs (incl. splice)	Conservation + splice/reg	0.74-0.82
CADD [65]	2014	Support Vector Machine	Genomic + protein annotations	SNVs, indels (coding + non-coding)	Genome-wide annotations	0.85-0.90
REVEL [66]	2016	Ensemble (Random Forest)	13 tool scores	Rare missense	Conservation + structure/function	0.90-0.93
BayesDel [67]	2017	Weighted naïve Bayes	Tool scores + allele frequency	Coding/non-coding SNVs, indels	Conservation + AA properties	0.90-0.93
ClinPred [68]	2018	Random Forest + Boosting	dbNSFP scores + gnomAD AF	Missense (nsSNVs)	Clinical pathogenicity	0.90-0.95
<i>Deep Learning Approaches</i>						
PrimateAI [70]	2018	Convolutional Neural Network	Sequence window + primate variants	Missense	Local sequence context	0.80
EVE [71]	2021	Variational Autoencoder	MSA	Missense	Evolutionary constraints	0.911
ESM1b [9]	2023	Protein Language Model	Single amino acid sequence	Missense, in-frame indels, stop-gains	Sequence context, no alignment	0.897-0.905
AlphaMissense [8]	2023	Transformer	Sequence + AlphaFold2 structure	Missense	Structure + evolution	0.940
Şilbr & Kurt [74]	2025	2D Convolutional Net	2D amino acid images	Missense (BRCA1)	2D AA representation	0.92
DeepRank-Mut [75]	2023	3D Convolutional Net	3D structure (voxel grid) + PSSM	Missense	3D structural environment	0.77†

Citations in the first column point to each tool's original publication and support the method description, not the AUC. ROC-AUC values are approximate, taken from original publications or from later benchmarks on different datasets, and are not comparable across rows. The AlphaMissense and EVE values are the exception: both come from the same ClinVar benchmark of 18,924 variants [8]. †DeepRank-Mut reports accuracy (0.77) and MCC (0.52); it did not report an AUC.

3.2.6 DeepInsight in Bioinformatics

DeepInsight is a method for applying CNNs to tabular data by turning each feature vector into a 2D image. It uses t-SNE to place the features on a 2D plane so that features behaving in a similar way end up close together [17]. The idea rests on a property of CNNs: nearby pixels in an image usually carry related information. The original work showed the method on gene expression data and on several standard tabular benchmarks, and later studies extended it to other biological problems [17].

Applying DeepInsight to protein variant classification, as done in this thesis, has not been reported before as far as we are aware. The method has been used on gene expression and similar tabular data, but not on amino acid structural and functional descriptors for predicting variant pathogenicity. This thesis extends it by combining the t-SNE image with an MLP branch on the feature vector.

3.3 Research Gap

The tools reviewed in this chapter share three limitations that this thesis addresses.

First, most of them read the same narrow input: the amino acid sequence and the conservation pattern taken from an alignment of related proteins. Databases such as DescribePROT already provide much more for every residue in a proteome, including intrinsic disorder, secondary structure, solvent accessibility, flexible linkers, binding sites for DNA, RNA and other proteins, and molecular recognition features [10]. These descriptors say what a residue does inside the protein, not only how strongly it is conserved, and few predictors feed them to a model directly.

Second, post-translational modification sites are almost never used. A variant that destroys or creates a modification site can change how the protein is switched on and off, yet this information rarely reaches the model.

Third, methods that turn tabular data into images so that CNNs can be used, such as DeepInsight, have not been applied to variant pathogenicity as far as we are aware [17].

This thesis addresses all three by bringing five kinds of information about a variant into a single model: per-residue structural and functional descriptors from DescribePROT, PTM sites, evolutionary scores, population allele frequencies, and the physicochemical change caused by the substitution. The descriptors are arranged as a two-dimensional protein window and as a DeepInsight image, and are joined with a variant-level feature vector through an MLP branch. Once variants of uncertain significance are removed, ClinVar labels the remaining variants in four categories, so the models are trained on those four; the same predictions are also

reported as the benign against pathogenic split that clinical practice acts on [12].

Chapter 4

Methodology

This chapter describes how germline missense variants of clinical significance are classified. It combines clinical variant data from the NCBI ClinVar repository with per-residue protein features from the DescribePROT database [10]. The work starts with data collection and the merging of the two datasets. From the merged data we build two representations: a two-dimensional matrix that captures the local protein environment around the mutation site, and a vector that summarises population frequency and biochemical effects of the substitution. The chapter then introduces the deep learning models. Three main architectures read these two representations: a VGG-based CNN on the protein window, a dual-branch model that fuses a VGG branch with an MLP branch, and a DeepInsight–DenseNet121 model that turns the protein window into an image [17]. Two further models act as ablations that isolate each input: a tabular MLP on the feature vector alone, and a DenseNet CNN on the image alone. The models are trained on the four ClinVar categories that remain after variants of uncertain significance are removed, and their predictions are also reported as a binary benign against pathogenic split.

Figure 4.1 gives the big picture before the chapter goes stage by stage.

4.1 Data Collection and Integration

4.1.1 ClinVar Extraction

ClinVar is a public NCBI repository where laboratories report human genetic variants [3]. It provides two files. The tabular file (Tab-Separated Values (TSV)) holds protein-level variants, gene names, protein changes, classifications, and Sequence Position Deletion Insertion (SPDI) notation. The Variant Call Format (VCF) file holds genomic coordinates, reference and alternate alleles, allele frequencies (AF_ESP, AF_TGP, AF_EXAC), and rsIDs.

The ClinVar tabular export lists coordinates for both the GRCh37 and GRCh38 assemblies. The pipeline used only the GRCh38 columns (`GRCh38Chromosome`

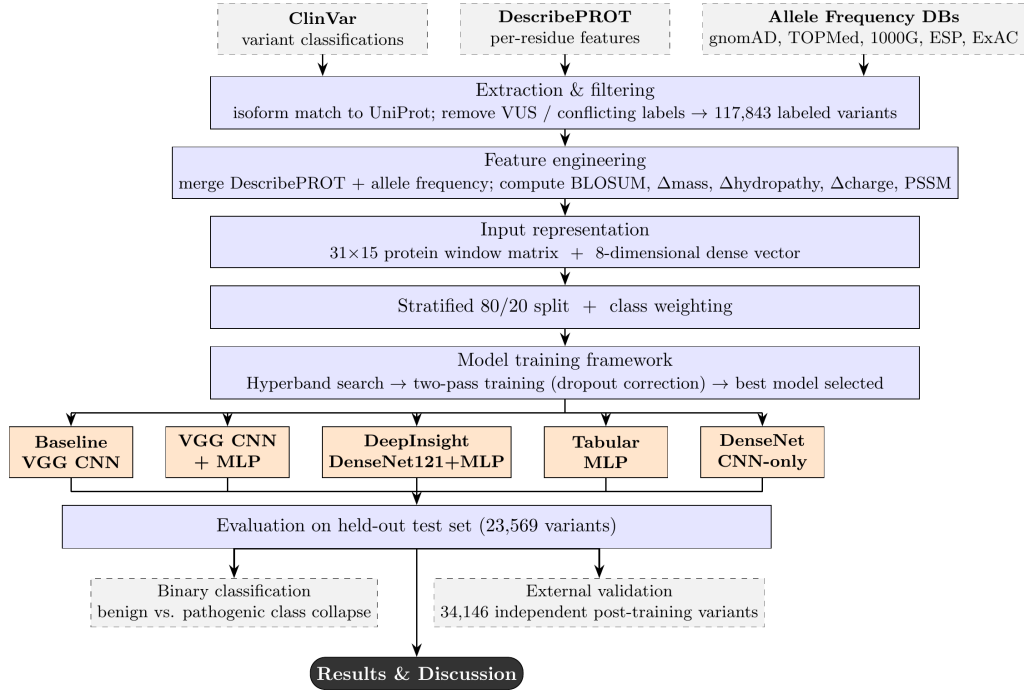


Figure 4.1: Overall methodology workflow: data collection, feature engineering, model training, and evaluation.

and `GRCh38Location`), and the allele frequencies came from ClinVar’s GRCh38 VCF. Because both sources use GRCh38, the merge key stays consistent and no coordinate conversion was needed.

The dataset was downloaded on 14 August 2024. Filters set on the ClinVar website before download selected the germline classification and the missense molecular consequence, so somatic variants and other consequence types were excluded at the source.

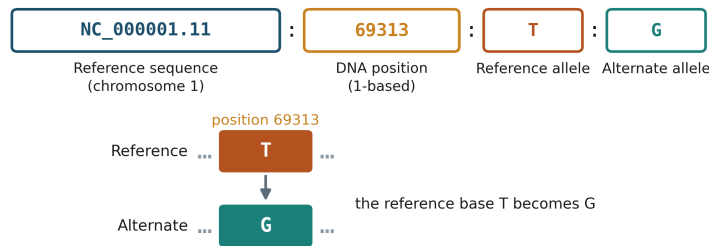
The tabular file provides the SPDI field `"NC_000001.11:69313:T:G"`, which gives the DNA reference and alternate alleles. The Human Genome Variation Society (HGVS) field gives the amino acid change, for example:

`NM_000277.3(PAH):c.1222C>T(p.Arg408Trp)`

The pipeline skipped rows whose Name field carried no protein-level change, and rows whose Protein change field held no valid amino acid substitution.

Finally the VCF records were merged with the TSV rows. The VCF file was loaded into an in-memory dictionary, and each record was indexed by a key holding chromosome, genomic position, reference nucleotide and alternate nucleotide. This produced a JavaScript Object Notation (JSON) file with 1,446,303 variants. Figure 4.2 shows how the DNA nucleotide and amino acid changes are extracted from the SPDI and HGVS annotations.

1. SPDI (DNA level)



2. HGVS (protein level)

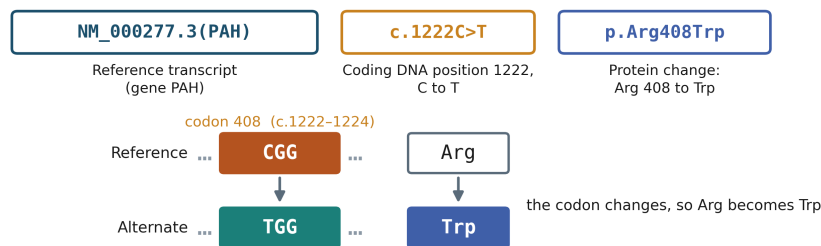


Figure 4.2: Extraction of DNA nucleotide and amino acid changes from SPDI and HGVS annotations.

4.1.2 Allele Frequency (AF) Enrichment

The NCBI Variation Services Representational State Transfer (REST) Application Programming Interface (API) was used to collect population allele frequencies from several cohorts, including gnomAD, TOPMed, the 1000 Genomes Project, the National Heart, Lung, and Blood Institute (NHLBI) Exome Sequencing Project (GoESP), and ExAC.

Each variant in the ClinVar JSON file that carried a numeric rsID was sent to the NCBI Variation API endpoint <https://api.ncbi.nlm.nih.gov/variation/v0/refsnps/> followed by the rsID [76]. Ten parallel threads were used to shorten the running time, and progress was written to disk every 50 rsIDs so that an interrupted run could resume. Variants without an rsID, and variants whose request returned no frequency data, kept only the allele frequencies taken from the VCF.

The API returns a raw allele count and a total count, and the allele frequency is computed as:

$$AF = \frac{\text{allele count}}{\text{total count}}$$

when the total count is greater than zero, and is set to zero otherwise. These frequencies were then merged with the values already taken from the VCF. Eleven of the merged columns carry data for this dataset.

These eleven columns are summarised into two features. The first, `Composite_AF`, is the maximum value across them, with missing values replaced by 0.0.

Different cohorts represent different global populations, and a variant may appear rare in some cohorts while common in others. Taking the maximum captures the strongest available evidence that a variant is common somewhere in the world. The minimum and the mean both hide this signal. Many cohorts report zero or no data for a population-specific variant, so the minimum collapses to zero, and the mean is pulled down by the cohorts where the variant was never observed. A high frequency in any single population is strong evidence that a variant is tolerated, so the maximum is the value that matters for pathogenicity [12].

`Composite_AF` measures magnitude. The second feature, `n_databases_seen`, measures consistency: it counts how many of these columns report a frequency above zero. A variant reported by many independent cohorts is more likely to be a genuine common polymorphism than one seen in a single dataset. The log transformation applied to `Composite_AF` before training is described in Section 4.2.2.

4.1.3 DescribePROT Data

DescribePROT [10] is a database of amino acid-level structural and functional descriptors. Its 2023 update covers 273 reference proteomes and provides 19 descriptors produced by 11 predictive tools: 4 structural (solvent accessibility, secondary structure, intrinsic disorder, disordered linkers), 13 functional (protein-binding, MoRFs, RNA-binding, DNA-binding, and nine PTM types), and 2 sequence-based (conservation and signal peptides). This study uses only the human proteome subset (`9606_databases.json`).

From DescribePROT, 13 continuous per-residue scores are extracted: intrinsic disorder propensity (`fIDPnn`), secondary structure confidence for beta-strand and alpha-helix (`PSIPRED`, two scores), flexible linker probability (`DFLpred`), relative solvent accessibility (`ASAquick`), disorder-based DNA-, protein-, and RNA-binding scores (`DisoRDPbind`, three scores), sequence-based DNA- and RNA-binding scores (`DRNAPred`, two scores), protein-binding score (`SCRIBER`), molecular recognition feature score (`MoRFchibi`), and evolutionary conservation score (`MMseqs2`). A further per-residue channel marks whether any of nine PTM types (phosphorylation, glycosylation, ubiquitination, SUMOylation, acetylation, methylation, pyrrolidone carboxylic acid, palmitoylation, and hydroxylation) is present at that residue. These 14 channels, together with an amino acid identity token, form the 15 channels of the 31×15 input matrix described in Section 4.2.1.

The data were stored in a Python dictionary indexed by UniProt entry name, so that lookups during merging were fast.

4.1.4 Mapping via UniProt

ClinVar uses gene symbols, while DescribePROT uses UniProt entry names, so the two cannot be joined directly. Each gene symbol was sent to the UniProt REST service at <https://rest.uniprot.org/uniprotkb/stream> and resolved to its reviewed Swiss-Prot entry, taking the first entry name ending in `_HUMAN` [22]. Results were cached, so each gene symbol was requested only once. This entry name is the key that joins the two datasets.

4.1.5 Merging ClinVar with DescribePROT

Amino acid-level features were extracted from DescribePROT and attached to each ClinVar variant. For every variant, the values at the mutated position were taken, including the PSSM score and the PTM annotations, and joined with the allele frequencies and the variant metadata into a single record.

DescribePROT stores PTMs as nine binary columns, one for each modification type. These were combined into a single feature:

$$\text{PTM_Binary_Sum}(i) = \max\{\text{ptm}_t(i) : t \in \{1, 2, \dots, 9\}\}$$

so the feature is 1 when at least one of the nine types is annotated at position i , and 0 otherwise. Each individual PTM column is zero for more than 99% of residues, so combining them keeps the signal without adding nine inputs that are almost all zeros. Figure 4.3 shows how the PTMbinary field is decoded into residue-specific PTM features.

The PSSM score comes from the `MMseqs2_pssm` field. DescribePROT stores the whole matrix of a protein as one long string: commas separate the rows, and spaces separate the values inside a row. The first row is a header, `Pos Cns A C D E F G H I K L M N P Q R S T V W Y`. Every row after it holds three things: the position, the wild-type residue at that position, and 20 scores, one for each amino acid in alphabetical order. Extracting the score for a variant takes three steps. First, the string is split into rows to rebuild the table. Second, the row of the variant position is selected; DescribePROT counts positions from zero while ClinVar counts from one, so one is subtracted from the ClinVar position. Third, the score is read from that row, at the column of the mutant amino acid.

For example, the matrix of NOC2L (`NOC2L_HUMAN`, 749 residues) holds the row `152 M -1 -1 -1 -3 ...` for residue 153. The wild-type methionine scores 10

4.1.6 Biochemical Computations

Three delta features describe how the substitution changes a physical property of the residue:

$$\Delta\text{property}(i) = \text{property}_{\text{mut}}(i) - \text{property}_{\text{ref}}(i)$$

where $\text{property}_{\text{ref}}(i)$ is the value for the reference amino acid at position i and $\text{property}_{\text{mut}}(i)$ the value for the mutant amino acid. The three properties are molecular weight, hydropathy on the Kyte-Doolittle scale [44], and charge at pH 7.0. A large delta means the substitution replaces a residue with one of very different chemistry.

The BLOSUM62 score for the reference-to-mutant pair was read from the substitution matrix. The matrix is symmetric, so the score does not depend on the order of the pair [16].

4.1.7 Dataset Splitting

The dataset is split into a training set (80%) and a test set (20%) by stratified random sampling with a fixed seed of 42, so that every model is evaluated on exactly the same variants [77].

A plain random split can leave a minority class underrepresented in the test set by chance, which makes the scores for that class unstable [49]. Stratification avoids this. Each class is divided separately using the same 80/20 ratio, so the class proportions in the training and test sets match those of the full dataset. The split produces 94,274 training variants and 23,569 test variants, distributed as shown in Table 4.1.

Table 4.1: Training and test set distribution after stratified splitting.

Class	Training	Test
Benign	24,286	6,072
Likely Benign	37,941	9,486
Likely Pathogenic	19,305	4,826
Pathogenic	12,742	3,185

The binary target reuses this same split. The two benign classes are grouped together and the two pathogenic classes are grouped together, so no variant moves between the training and test sets and the binary and four-class results are measured on exactly the same variants. Table 4.2 gives the resulting distribution. Pathogenic variants make up 34.0% of the training set and 34.0% of the test set, so the stratification carries over to the grouped labels.

Table 4.2: Training and test set distribution for the binary target.

Class	Training	Test	Total
Benign (Benign + Likely Benign)	62,227	15,558	77,785
Pathogenic (Likely Pathogenic + Pathogenic)	32,047	8,011	40,058
Total	94,274	23,569	117,843

4.1.8 Class Imbalance and Class Weighting

The four classes are not the same size. ClinVar is built from variants that laboratories choose to submit and interpret, so the class sizes reflect what gets submitted and curated, not how often each kind of variant occurs. Common population variants are easy to call benign, so the Benign and Likely Benign classes are large. A firm Pathogenic call needs strong evidence, which fewer variants have, so the Pathogenic class is the smallest. There are three times as many Likely Benign variants as Pathogenic ones.

This imbalance pushes training toward the large classes. Every sample adds its own cross-entropy loss to the total, so the common classes control most of the loss, and the optimizer learns to predict them well while paying little attention to the rare ones. The model can then reach a high overall accuracy while its recall and F1 on the small classes stay low [49].

To correct this, each class was given a weight inversely proportional to its frequency in the training set, using Scikit-learn’s balanced option:

$$w_i = \frac{N}{K \times n_i},$$

where N is the number of training samples (94,274), K is the number of classes (4), and n_i is the number of training samples in class i . Minority classes receive larger weights, so their errors count more during training. Table 4.3 lists the resulting weights: Likely Benign receives the smallest (0.62) and Pathogenic the largest (1.85), so one Pathogenic error costs about three times as much as one Likely Benign error.

Table 4.3: Class weights used during model training.

Class	Training Count	Weight
Benign	24,286	0.97
Likely Benign	37,941	0.62
Likely Pathogenic	19,305	1.22
Pathogenic	12,742	1.85

The same procedure applied to the binary target gives a weight of 0.76 for benign and 1.47 for pathogenic, a milder correction because grouping the classes

reduces the imbalance from roughly three to one down to about two to one.

The weights were computed from the training set only, after the split, so no information from the test set leaks into training. Unlike oversampling or undersampling, class weighting leaves the data unchanged and rebalances the loss instead.

4.2 Input Representation

4.2.1 Protein Window Matrix Construction

Each ClinVar variant is mapped to its protein sequence in DescribePROT. For each variant a 31×15 matrix is built by taking the 15 amino acids before the mutation site and the 15 after it. The window therefore holds 31 amino acids, with the mutated one in the middle, at position 16.

The matrix starts as an array of zeros created with NumPy. This also handles the ends of the protein: when the window runs past the start or the end of the sequence, those rows are simply left at zero. Algorithm 1 shows the construction for a single variant.

Each of the 31 positions holds 15 values. The first is the amino acid itself, stored as an integer between 1 and 20. The other 14 are the DescribePROT per-residue scores and the binary PTM flag listed in Section 4.1.3.

Most of these values already lie between 0 and 1. Two do not, so they are rescaled. The PSIPRED strand and helix scores are percentages, so they are divided by 100. The MMseqs2 conservation score runs from 0 to 4.34 bits, so it is divided by 4.34. Every channel then sits on the same $[0, 1]$ scale.

The result is a 2D array of 31 positional rows and 15 feature columns. Keras Conv2D layers do not accept 2D arrays: they need three dimensions, height, width and depth. A channel dimension is therefore added, giving a final input shape of $(31, 15, 1)$ for each variant, so the matrix is treated as a grayscale image.

The dataset contains 117,843 variants. All of their matrices are stacked into a tensor of shape $(117,843, 31, 15, 1)$ and saved together with the dense features and the class labels in `protein_cnn_final_dataset_v2.npz`, which is the file used for training and evaluation. Figure 4.4 shows the matrix for one example variant.

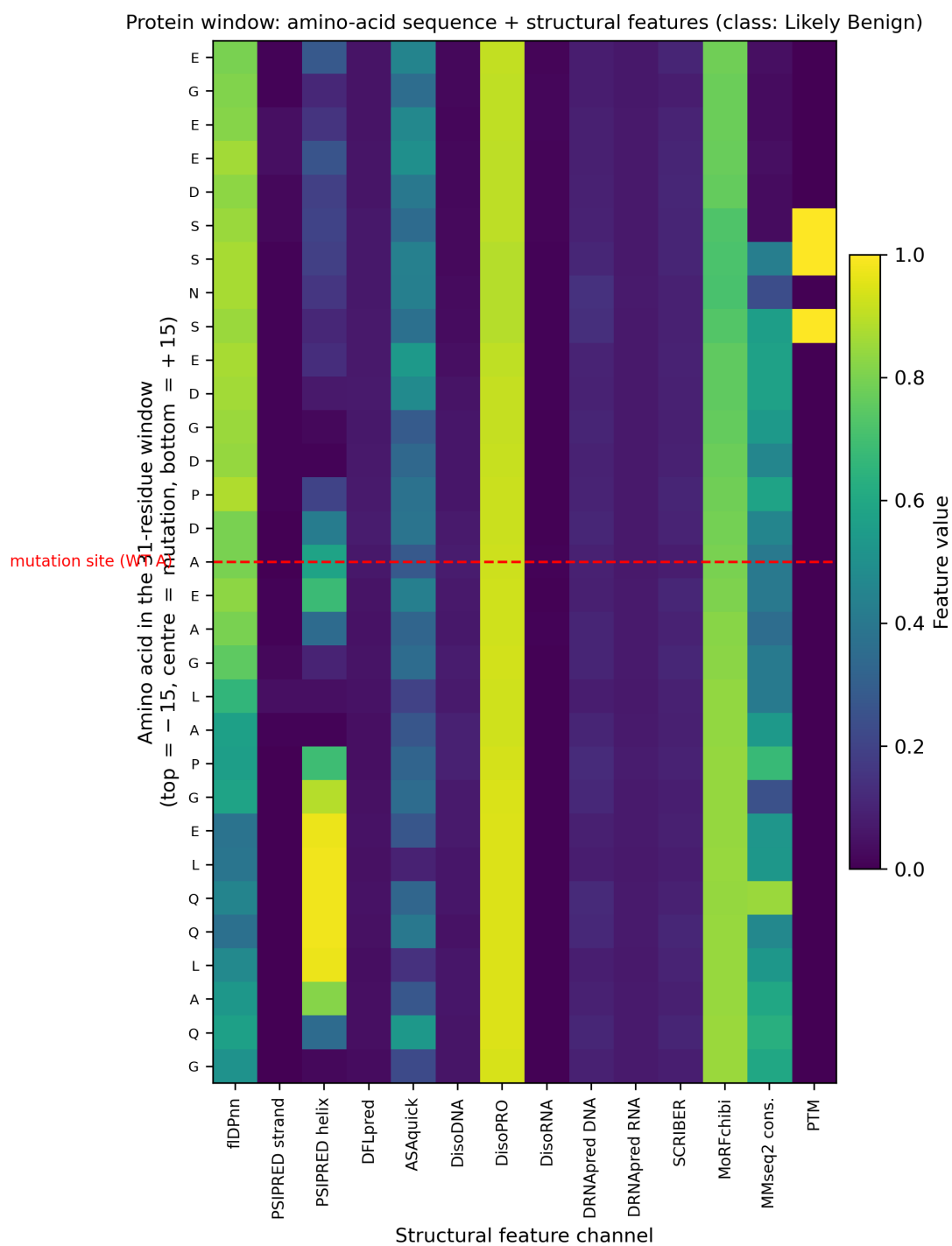


Figure 4.4: The 31×15 protein-window matrix for a single variant.

Algorithm 1 Protein window matrix construction for one variant

Input: `seq`: the protein sequence; `features`: the 14 normalized per-residue arrays; `p`: the affected amino acid position (1-based)

Output: `matrix`: a 31×15 array for one variant

```

1: matrix  $\leftarrow \mathbf{0}^{31 \times 15}$                                  $\triangleright$  the zeros also serve as the padding
2: pos  $\leftarrow p - 1$                                            $\triangleright$  DescribePROT indexes from 0

3: for w  $\leftarrow 0$  to 30 do
4:   g  $\leftarrow \text{pos} - 15 + w$                                  $\triangleright$  position in the full sequence
5:   if  $0 \leq g < \text{len}(\text{seq})$  then
6:     matrix[w, 0]  $\leftarrow \text{AA\_VOcab}[\text{seq}[g]]$                $\triangleright$  channel 0: amino acid, 1–20
7:     matrix[w, 1:15]  $\leftarrow \text{features}[g]$                    $\triangleright$  channels 1–14
8:   end if
9: end for

10: return matrix                                               $\triangleright$  rows that fall outside the protein stay zero

```

4.2.2 Dense Feature Vector

Each variant is represented by a dense feature vector of eight numerical features. These describe the variant itself, and most of them the local protein window cannot capture. The first two are population measures: the log-transformed composite allele frequency and the number of population databases in which the variant has been seen. Three features describe the substitution chemistry: `Delta_mass`, `Delta_hydrophathy` and `Delta_charge`. Two are evolutionary: the BLOSUM62 score and the PSSM score. The last is a binary indicator of a PTM site at the variant position.

The PSSM score is read from the `MMseqs2_pssm` field, taking the score of the mutant amino acid at the variant position, so its value depends on the specific substitution. This makes it different from the `MMseqs2_conservation_score` used in the protein window, which describes the position alone and is the same whatever the substitution is. The PTM indicator also overlaps with the PTM channel of the window, and the repetition is deliberate: the window channel spreads PTM information across all 31 positions and passes it through convolution and pooling, while the dense feature carries the PTM status of the mutated residue straight to the classifier, where it is not diluted.

Before scaling, the composite allele frequency is log-transformed:

$$\log_{10}(\text{Composite_AF} + 10^{-9})$$

The small constant avoids the logarithm of zero for variants absent from every

population database. The raw frequency is strongly right-skewed: almost all variants sit close to zero, which makes the untransformed feature hard for gradient-based training to use. After the transform the distribution becomes bimodal (Figure 4.5). Half of the pathogenic variants fall on the floor at -9 , meaning no population database records them at all, against 3% of the benign variants. The benign variants form a second peak near -3.5 , and their median is -3.49 against -6.15 for the pathogenic ones. This gap between rare and common variants is the signal the classifier needs.

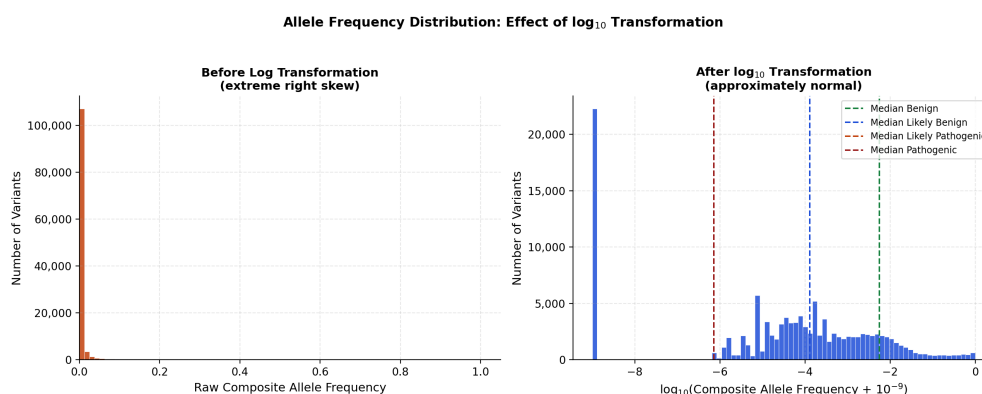


Figure 4.5: Composite allele frequency before (left) and after (right) the $\log_{10}$ transformation.

The dense features are scaled with `StandardScaler`, which shifts each feature to a mean of zero and a standard deviation of one. The scaler is fitted on the training data only, so nothing from the test set influences the scaling.

The dense vectors for all 117,843 variants form a matrix of shape $(117,843 \times 8)$, stored in the same file as the protein windows (Section 4.2.1). The file holds the raw values; the scaling is applied later, inside each training script, after the split. Figure 4.6 shows the dense vector for one example variant.

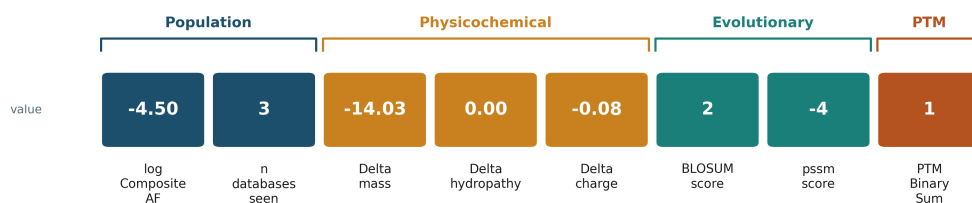


Figure 4.6: The eight-feature dense vector for one Pathogenic variant, showing the raw values stored in the dataset file before standardisation.

4.3 Model Training Framework

All five models share the same environment, data preparation, training protocol and evaluation metrics. They are built with TensorFlow 2.x using the Keras Functional API, which is needed because two of the models take two inputs at once. Every model is tuned with Keras Tuner Hyperband. Hyperband runs many short trials, drops the weak ones early, and gives the remaining budget to the configurations that look best. Each trial runs for at most 20 epochs, and the search selects on the validation Macro F1 score.

Each variant enters the network as two arrays: a CNN input of shape $(N, 31, 15, 1)$ and a dense input of shape $(N, 8)$. Training uses the stratified split of Section 4.1.7 and the class weights of Section 4.1.8. Labels are stored as integers, so the loss is Sparse Categorical Cross-Entropy. The search chooses between two adaptive optimizers, Adam and Nadam.

The 23,569 test variants also served as the validation data during training, so early stopping, the learning-rate schedule and the checkpoint all watched the score on this set. For this reason the independent dataset described in Section 4.6 carries more weight: it was not used at any point during training, and it gives the estimate of how the model performs on variants it has never seen.

Training runs in two passes. After the first pass, the gap between training accuracy and validation accuracy is measured. If the gap is larger than 5%, the model is trained again with a higher dropout rate: 0.10 higher when the gap is between 5% and 10%, and 0.20 higher above that. If the gap stays below 5%, the result of the first pass is kept.

Three callbacks run during training, and all of them watch the Macro F1 score:

- Early stopping, when the score does not improve for seven epochs.
- Halving the learning rate when the score stops improving for three epochs, down to a floor of 10^{-6} .
- Checkpointing, which keeps the weights from the epoch with the best score.

Figure 4.7 summarises the complete training pipeline as an activity diagram.

4.3.1 Baseline VGG-Based CNN Model

The baseline model is a VGG-style CNN that reads the protein feature matrix as a grayscale image. It borrows the core ideas of VGG rather than using a full VGG-16 or VGG-19 network [55]: small 3×3 convolution filters, two convolutions stacked before each pooling step, max-pooling to halve the spatial size, and a doubling of

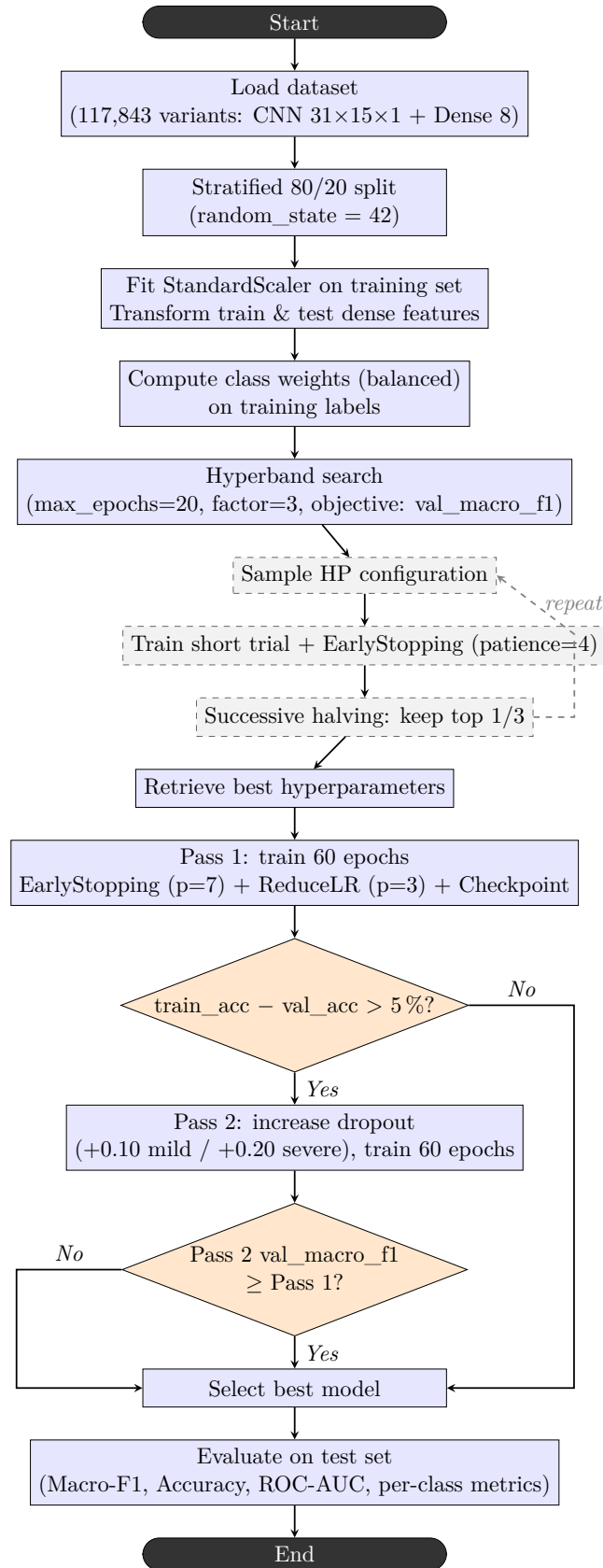


Figure 4.7: Activity diagram of the shared model training framework: data preparation, Hyperband hyperparameter search, two-pass training, and overfitting correction.

the filter count after each block. Together these build up the receptive field with few parameters per layer.

The full VGG-16 and VGG-19 networks were not used, for three reasons. First, they were built for 224×224 natural images, while the protein window is only 31×15 ; the five pooling stages of a full VGG would reduce this input to nothing after the first few blocks. Second, VGG-16 has about 138 million parameters and was trained on more than a million ImageNet images, so it would overfit a dataset of 117,843 variants. Third, the protein matrix is not a natural image, so ImageNet pretraining does not transfer. The depth is therefore kept shallow and tunable, between one and three blocks, to match the small input and the dataset size.

The convolutional layers are organized into blocks. Each block consists of:

- A 3×3 convolutional layer with same padding,
- A nonlinear activation function,
- A second 3×3 convolutional layer,
- A second activation function,
- A 2×2 max-pooling layer.

The number of filters doubles after each block, so a search that chooses 32 initial filters and three blocks gives 32, 64 and 128 filters.

After the last block the feature maps are flattened into a one-dimensional vector. This passes through a dropout layer, a dense hidden layer, a second dropout at half the first rate, and finally a softmax layer with four neurons, one for each pathogenicity class (Figure 4.8).

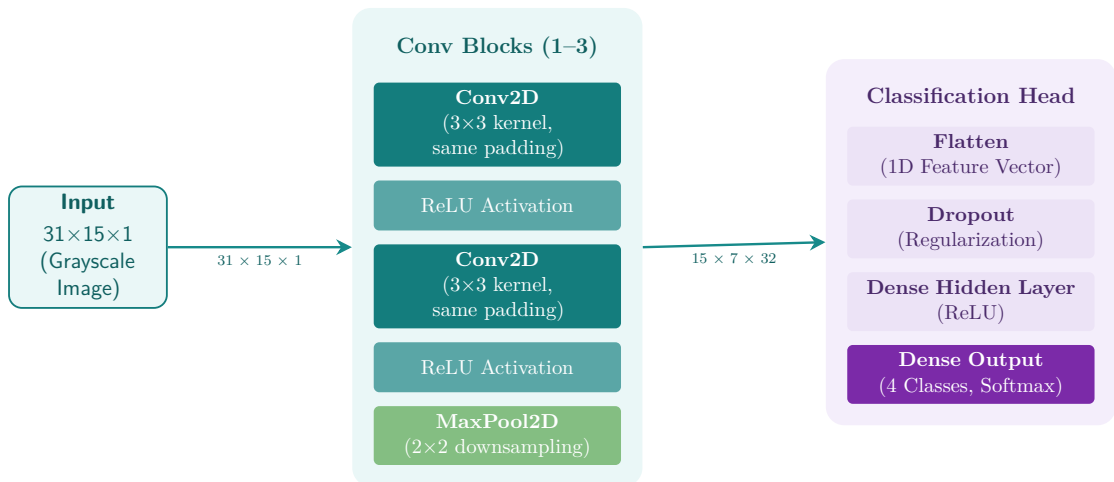


Figure 4.8: Baseline VGG-CNN architecture.

Hyperparameter Search Space

The hyperparameter search space explored for the baseline model is listed in Table 4.4.

Table 4.4: Hyperparameter search space for VGG-based model.

Hyperparameter	Options
<i>n_blocks</i>	1, 2, or 3
initial_filters	16, 32, or 64
activation	ReLU or Swish
optimizer	Adam or Nadam
learning_rate	0.001, 0.0005, or 0.0001
dropout	0.3 to 0.6
head_units	64, 128, or 256

4.3.2 Dual-Input CNN + MLP Model

The second model adds a branch that reads the eight tabular features alongside the protein window. It takes two inputs:

- a CNN input of shape $(N, 31, 15, 1)$, the same as Model 1,
- a dense input of shape $(N, 8)$.

The image branch is the same VGG stack as Model 1 and ends at a dense layer. The MLP branch passes the eight features through one to three dense layers, each followed by an activation and a dropout. Hyperband chooses the number of these layers and the units in each one. Both branches end at a dense layer that outputs a feature vector, not a class, so neither branch makes a prediction on its own.

The two feature vectors are then joined by a concatenate operation, which places them side by side in one longer vector. Concatenation only stacks the two sources; it does not mix them. The mixing happens in the fusion dense layer that follows, because every unit in that layer sees values from both branches at once. After the fusion layer come an activation and a dropout, and then a softmax layer with four neurons that produces the class probabilities. The prediction is therefore made once, on the fused representation, rather than separately in each branch (Figure 4.9).

Extended Hyperparameters

In addition to Model 1 hyperparameters, the dual-input model introduces the hyperparameters listed in Table 4.5:

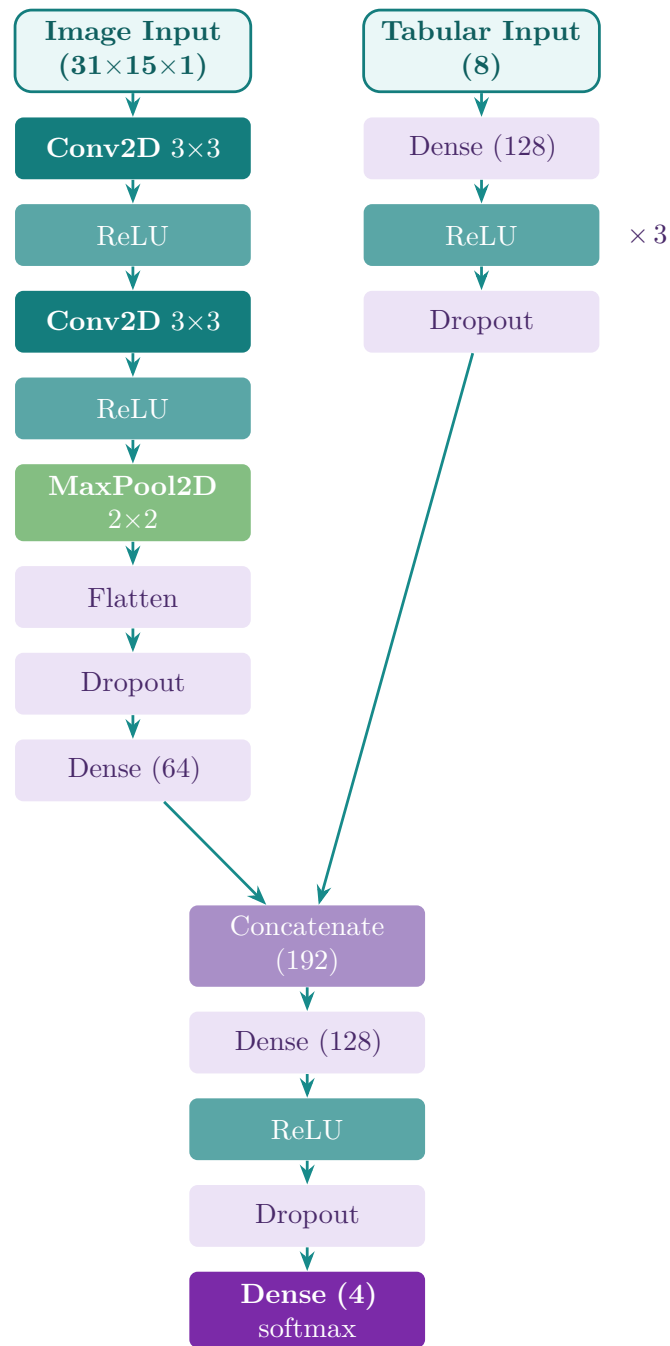


Figure 4.9: Dual-input VGG + MLP architecture (Model 2).

Table 4.5: Additional hyperparameters for dual-input model.

Hyperparameter	Options
mlp_layers	1, 2, or 3
mlp_units	32, 64, or 128
fusion_units	64, 128, or 256

4.3.3 DeepInsight with DenseNet121 and MLP Model

The third model combines the DeepInsight transformation, a DenseNet121 CNN, and an MLP branch.

DenseNet121 requires a minimum input size of 32×32 . Since the original protein feature matrix has size (31×15) , it is transformed using DeepInsight.

First, each variant is flattened into a 465-dimensional vector. Then t-SNE is applied to project features into a 2D space while preserving local relationships. These coordinates are mapped onto a fixed 32×32 grid, where each feature is assigned a unique pixel location. This produces a grayscale image of size $32 \times 32 \times 1$.

The DenseNet121 backbone is implemented using TensorFlow Keras with `include_top=False`, so it has no built-in classification head. Since the inputs are biological matrices rather than natural images, the backbone is initialized with random weights instead of ImageNet pretraining and is trained from scratch on the DeepInsight images.

DenseNet is built from dense blocks. Inside a dense block, each convolutional layer receives the feature maps of all preceding layers, concatenated along the channel dimension, and forwards its own output to every later layer. This dense connectivity lets the network reuse features and keeps the gradient flowing through many layers. DenseNet121 has about seven million parameters.

The output of the final dense block is summarized by two global pooling operations. Global average pooling takes the mean of each feature map, and global max pooling takes its strongest activation. The two summaries are concatenated and passed through a dropout layer with rate 0.4. Average pooling captures the overall level of each feature across the image, whereas max pooling captures its peak, so using both gives the classifier a richer summary than either one alone.

The pooled image features are then concatenated with the output of the MLP branch, and the combined vector is passed through a dense fusion layer and a four-class softmax output. The dense layers use L2 regularization and layer normalization to limit overfitting. Because training all 121 DenseNet layers is computationally heavy, this model uses fixed hyperparameters (Adam, learning rate 0.001, dropout 0.4, L2 rate 0.001; two MLP layers of 128 units and a fusion layer of 256 units) rather than a Hyperband search, while keeping the same two-pass overfitting check used by the other models (Figure 4.10).

4.3.4 Tabular MLP Model

The fourth model is a standalone MLP that uses only the eight tabular features, with no image branch. It is a baseline that measures how much of the dual-input performance comes from the tabular features alone.

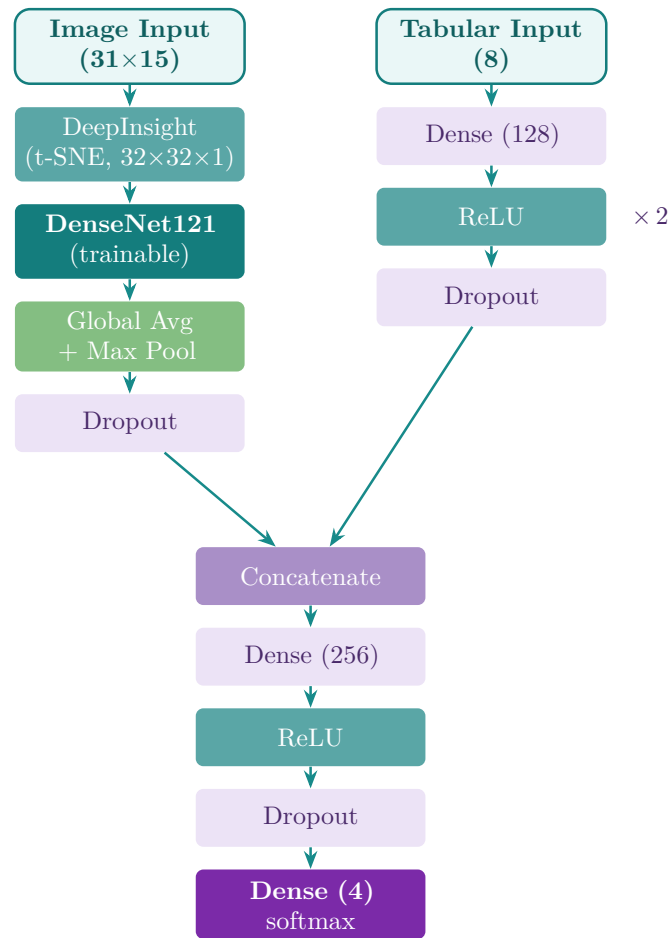


Figure 4.10: DeepInsight + DenseNet121 + MLP architecture (Model 3).

The input is the 8-dimensional feature vector. It passes through one to three hidden layers, each a dense layer followed by an activation and a dropout, and ends in a softmax output layer with four neurons.

The model reuses the shared training framework: the same seed-42 stratified split and test set, the same StandardScaler fitted on the training data, the same balanced class weights, and the same Hyperband search with the two-pass overfitting check. The hyperparameter search space is listed in Table 4.6.

Table 4.6: Hyperparameter search space for the tabular MLP model.

Hyperparameter	Options
n_layers	1, 2, or 3
units	32, 64, 128, or 256
activation	ReLU or Swish
dropout	0.2, 0.3, 0.4, or 0.5
optimizer	Adam or Nadam
learning_rate	0.001, 0.0005, or 0.0001

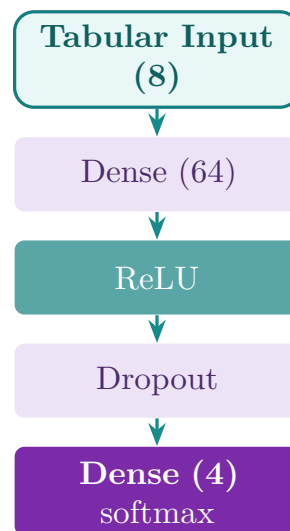


Figure 4.11: Tabular MLP architecture (Model 4).

4.3.5 DenseNet CNN-Only Model

The fifth model is an ablation of the DeepInsight-DenseNet121 model with the MLP branch removed. It keeps only the DeepInsight image and the DenseNet121 CNN, so it measures how much the image branch contributes on its own.

The DeepInsight transformation and the trainable DenseNet121 backbone are identical to the third model. The DenseNet output is summarized by global average pooling and global max pooling, concatenated, and passed through a dropout layer, a dense layer, and a four-class softmax. There is no MLP branch and no

tabular feature vector, so the model sees only the protein window converted to a DeepInsight image.

Together with the tabular MLP baseline, this model isolates the two input sources: one shows what the eight variant-level features contribute, the other what the DeepInsight image contributes.

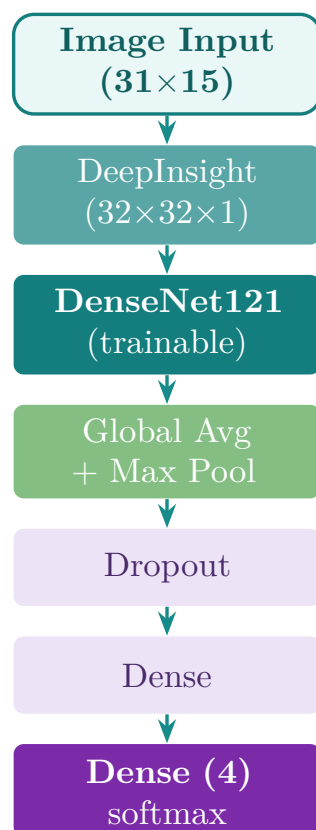


Figure 4.12: DenseNet CNN-only architecture.

4.4 Binary Classification

Most existing variant effect predictors report a single benign-versus-pathogenic decision rather than the four ACMG classes. To compare with these tools, and to measure performance on the easier two-class problem, four architectures were retrained as binary classifiers: the image-only VGG, the VGG + MLP fusion, the DeepInsight DenseNet121 + MLP, and the tabular MLP.

The four ordered classes were collapsed into two. A variant was labelled pathogenic if its class index was two or higher, so Likely Pathogenic and Pathogenic formed the pathogenic class, while Benign and Likely Benign formed the benign class. Pathogenic variants make up 34.0% of the dataset.

Each model keeps its four-class architecture. Only the output layer changes: the four-neuron softmax is replaced by a two-neuron softmax. The networks are

trained from scratch on the binary labels and do not reuse the weights of the four-class models.

The test set is identical to the four-class experiments: the same stratified 80/20 split with seed 42, stratified on the original four-class labels, produces the same 23,569-variant test set, so the binary and four-class results are measured on exactly the same variants. A further 15% of the training data is held out for validation, giving 80,132 training and 14,142 validation variants. The StandardScaler is fitted on the training part only, and balanced class weights are recomputed for the two classes.

The residue window was swept for the image-only VGG and the fusion model. Section 4.5 describes that experiment.

At test time, a variant is predicted pathogenic when the probability of the pathogenic class is at least 0.5. Models are ranked by the Matthews correlation coefficient (MCC) rather than accuracy: with 34% of variants pathogenic, a model that calls everything benign still scores 66% accuracy, while its MCC is zero. The results are reported in Section 5.8.

4.5 Window Size Selection

The protein window was set to 31 residues. Two experiments tested whether this width matters, one on the four-class task and one on the binary task.

On the four-class task, five window sizes were tested: 11, 21, 31, 41, and 51 residues. For each size, the protein-window matrix was rebuilt with that number of residues centred on the mutation, while the 15 feature channels stayed the same, so only the number of positional rows changed. The dual-input VGG and MLP model, with a fixed architecture, was then trained once on each of the five matrices. The dense features, the stratified 80/20 split, the class weights, and the training protocol were held fixed across all runs, so the window width was the only variable. Each of the five models was evaluated on the same held-out test set, and the sizes were compared using macro-F1, accuracy, and macro-AUC.

The binary experiment swept the window more densely. The image-only VGG and the VGG + MLP fusion were each trained at eight sizes: 5, 7, 11, 13, 17, 21, 25, and 31 residues. Two things differ from the four-class sweep. Each window ran its own Hyperband search, so the architecture was retuned at every size instead of being held fixed. And the windows were cut as centred slices of the stored 31-residue matrix rather than rebuilt from the protein sequence, so 31 is the widest size available. The DeepInsight model was trained at 31 residues only, because each size needs its own t-SNE mapping and a full DenseNet121 training run, and the tabular MLP does not read the window at all, so it was trained once.

The results of both experiments are reported in Section 5.7.

4.6 External Validation Dataset

To determine how the best model, VGG CNN + MLP, would perform on unseen variants, an independent external validation dataset was constructed. A fresh ClinVar variant dataset was downloaded on 2026-06-20 using the same search filter as the original extraction. The new file contained 2,338,971 rows, against 1,457,038 in the original 2024-08-14 extract. ClinVar gained approximately 880,000 records over the 22 months.

The fresh data passed through three filters before entering the feature pipeline. First, a date filter retained only variants whose Germline date last evaluated field exceeded 2024-08-14, leaving 1,181,955 candidates. Second, a class pre-filter dropped variants labelled “Uncertain significance” and “Conflicting interpretations”. Third, a strict deduplication step removed any variant whose VariationID appeared in the original raw extract or whose (chromosome, position, REF, ALT) tuple appeared in the trained pipeline CSV. After dedup the set held 43,256 variants. Table 4.7 shows the survival count at each stage of the construction pipeline.

Table 4.7: Filtering cascade for the external validation dataset.

Stage	Survivors
Initial rows in fresh ClinVar TSV	2,338,971
After date filter (> 2024-08-14)	1,181,955
After class pre-filter (B / LB / LP / P only)	106,045
After strict dedup against original 1.4M extract	43,256
After protein-position parsing and UniProt mapping	42,490
After isoform-mismatch and NULL-PSSM filters	34,146

The external dataset was used only once, after all training and tuning had finished. No hyperparameter, early-stopping decision or checkpoint was chosen using it, so it measures performance on variants that never influenced the model.

To avoid data leakage, the StandardScaler fitted on the original training set was reused to standardize the external dataset. No parameters were recalculated on the external data, so the external variants were scaled exactly like the internal test variants.

Chapter 5

Results and Discussion

5.1 Dataset Construction Results

The dataset process began with the collection of all germline missense variants from NCBI ClinVar. After the isoform and PSSM filters described in Chapter 4, 988,329 variants remained. Two label filters were then applied: 828,022 variants labelled VUS were removed, and a further 42,464 with conflicting interpretations were removed as well. This left 117,843 labelled variants, about 11.9% of the 988,329. Figure 5.1 shows the class distribution of the final dataset after filtering and preprocessing.

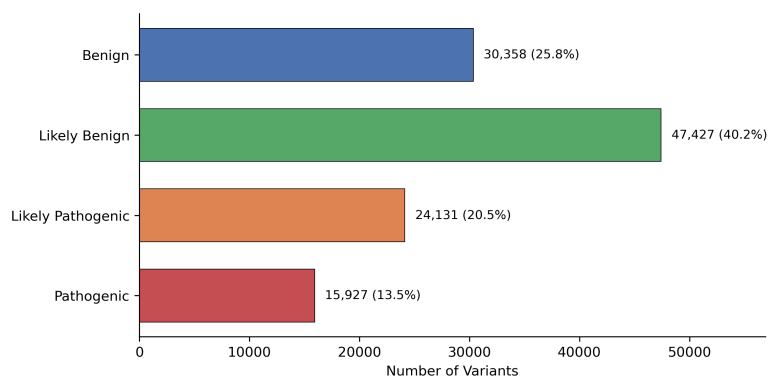


Figure 5.1: Class distribution of the final dataset after filtering and preprocessing.

5.1.1 Data Quality and Missing Value Handling

Allele frequency information was collected from multiple population databases through the NCBI Variation Services API and MyVariant.info. Many pathogenic variants are extremely rare or absent from population reference datasets, so a missing allele frequency usually means the variant was never observed, not that the record is incomplete. The frequencies were therefore reduced to two features: `log_Composite_AF`, the log-transformed maximum observed allele frequency across

all queried databases, and `n_databases_seen`, the number of databases in which a variant was observed.

An analysis of missing values revealed only a small number of missing entries within the DescribePROT-derived features. The `fDPnn` disorder score contained 258 missing values (0.026%), while the `DRNApred` DNA-binding and RNA-binding scores each contained 1,917 missing values (0.194%). During the construction of the CNN input windows, a total of 766 NaN values were identified, affecting 25 variants.

The proportion of missing values was very small, and 0.0 corresponds to a neutral or non-predicted state for these features, so all NaN values were replaced with 0.0 before training. Fewer than 0.2% of values in any channel were affected, so the substitution leaves the feature distributions essentially unchanged.

5.2 Model Evaluation Results

This section presents the results from the five deep learning models used for variant classification (Table 5.1). All models were evaluated on the same 23,569-variant test set across four classes: Benign, Likely Benign, Likely Pathogenic, and Pathogenic. Three are the main architectures (the baseline VGG CNN, the dual-input VGG + MLP, and the DeepInsight + DenseNet121 + MLP), and two are ablations that isolate a single input source: a tabular MLP that uses only the eight features, and a DenseNet CNN that uses only the DeepInsight image.

All five models were trained on the same computer. It has an NVIDIA GeForce 920M GPU, but TensorFlow 2.16 no longer supports GPU training on Windows, and this GPU is too old for it anyway. So every model in this chapter trained on CPU only. This made a big difference in training time. The VGG-based models (Model 1 and Model 2) and the tabular MLP (Model 4) each finished in under an hour. The DenseNet121-based models (Model 3 and Model 5) took about eight to ten hours each, because DenseNet121 has far more parameters to train.

5.3 Performance of the Baseline VGG CNN Model

The first model is a VGG-style CNN trained only on the protein window, the 31×15 feature matrix. It sees no variant-level annotations at all.

5.3.1 Best Hyperparameters (Keras Tuner)

The best hyperparameters selected by Keras Tuner for the baseline model are listed in Table 5.2.

Table 5.1: Summary of the deep learning models evaluated in this study.

Model Name	Input Type	Key Architecture	Training Time
Original VGG CNN	Raw protein window (31×15 matrix)	2-block VGG, CNN only	~40 min
VGG CNN + MLP	Raw protein window + 8 tabular features	2-block VGG + MLP branch	~29 min
DeepInsight DenseNet121 + MLP	t-SNE image (32×32) + 8 tabular features	Trainable DenseNet121 + MLP branch	~7h 55min
Tabular MLP	8 tabular features only	MLP only (no image)	~1–2 min
DenseNet CNN-only	t-SNE image (32×32) only	Trainable DenseNet121, CNN only	~9h 51min

Table 5.2: Best hyperparameters selected by Keras Tuner for the baseline VGG CNN model.

Hyperparameter	Best Value
n_blocks	2
initial_filters	64
activation	ReLU
optimizer	Nadam
learning_rate	0.001
dropout	0.40
head_units	64
tuning trials	30 (Hyperband)

5.3.2 Overall Test-Set Metrics

The overall test-set performance of the baseline model is summarized in Table 5.3.

Table 5.3: Overall test-set performance of the baseline VGG CNN model.

Metric	Value
Test samples	23,569
Accuracy	41.24%
Balanced Accuracy	42.35%
Macro F1-Score	0.4065
Weighted F1-Score	0.4143
MCC	0.2098
Log Loss	1.1125
Macro ROC-AUC	0.7529
Macro Avg Precision	0.4465

5.3.3 Per-Class Metrics

Table 5.4: Per-class performance metrics for the baseline VGG CNN model.

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	PR-AUC
Benign	3032	5318	12179	3040	0.4993	0.6961	0.3631	0.4205	0.6828	0.3897
Likely Benign	3280	2715	11368	6206	0.3458	0.8072	0.5471	0.4237	0.7181	0.5806
Likely Pathogenic	2070	2788	15955	2756	0.4289	0.8513	0.4261	0.4275	0.8088	0.4718
Pathogenic	1337	3029	17355	1848	0.4198	0.8514	0.3062	0.3541	0.8019	0.3439

All four classes score between 0.35 and 0.43 on F1, so the model separates them poorly (Table 5.4). Pathogenic is the weakest, with an F1 of 0.3541 and a precision of 0.3062, which means only 30.6% of the variants the model calls Pathogenic really are. ROC-AUC runs from 0.68 to 0.81, so the model still ranks variants better than chance even where its hard classifications fail (Figure 5.2).

The confusion matrix shows how the variants of each true class, in the rows, are spread over the four predicted classes, in the columns (Figure 5.3). The errors fall mostly between neighbouring classes. On the benign side, 0.46 of true Likely Benign variants are called Benign. On the pathogenic side, 0.38 of true Pathogenic variants are called Likely Pathogenic. Most of the model’s mistakes are therefore one step away from the true class rather than across the benign and pathogenic boundary.

5.4 Performance of the Hybrid VGG CNN + MLP Model

The second architecture combines the protein window, processed by a VGG CNN branch, with the eight variant-level features, processed by a separate MLP branch. The two branch outputs are fused before the final classification.

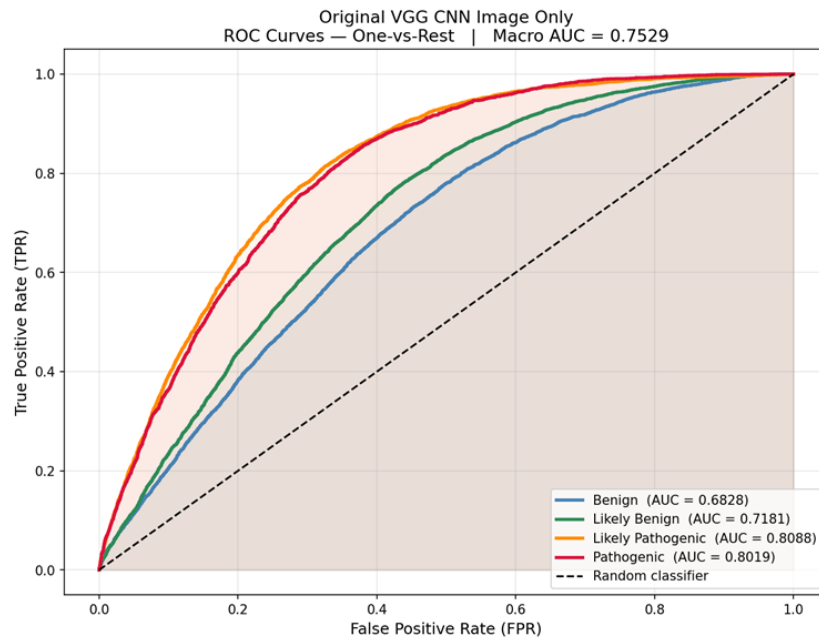


Figure 5.2: One-vs-rest ROC-AUC curves for the baseline VGG CNN model.

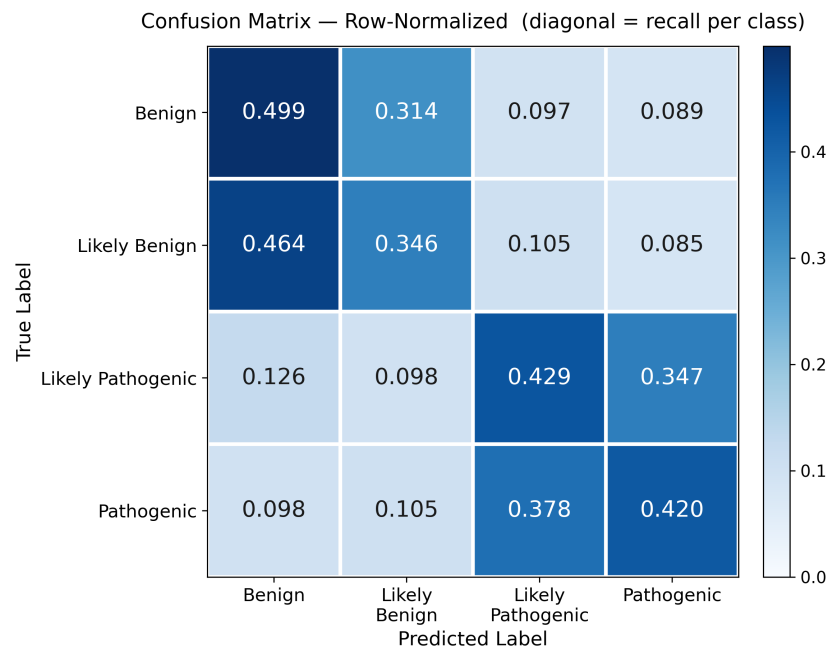


Figure 5.3: Confusion matrix for the baseline VGG CNN model. Values are row-normalised, so each row sums to 1 and the diagonal is the recall for that class.

5.4.1 Best Hyperparameters (Keras Tuner)

The best hyperparameters selected by Keras Tuner for the hybrid model are listed in Table 5.5.

Table 5.5: Best hyperparameters selected by Keras Tuner for the Hybrid VGG CNN + MLP model.

Hyperparameter	Best Value
n_blocks	2
initial_filters	64
activation	ReLU
optimizer	Adam
learning_rate	0.001
dropout	0.30
head_units	64
mlp_layers	3
mlp_units	128
fusion_units	128
tuning trials	30 (Hyperband)

5.4.2 Overall Test-Set Metrics

Table 5.6: Overall test-set performance of the Hybrid VGG CNN + MLP model.

Metric	Value
Test samples	23,569
Accuracy	66.77%
Balanced Accuracy	61.68%
Macro F1-Score	0.6179
Weighted F1-Score	0.6685
MCC	0.5329
Log Loss	0.7307
Macro ROC-AUC	0.8910
Macro Avg Precision	0.6614

This model reaches an accuracy of 66.77% and a Macro F1 of 0.6179 (Table 5.6). Its Macro ROC-AUC of 0.8910 is the highest of the five models evaluated (Figure 5.4).

Benign and Likely Benign recall reach 0.72 and 0.77, so the model handles the two benign classes well (Figure 5.5). Most of the remaining errors sit in the Likely Pathogenic and Pathogenic classes, where 0.50 of true Pathogenic variants are predicted as Likely Pathogenic.

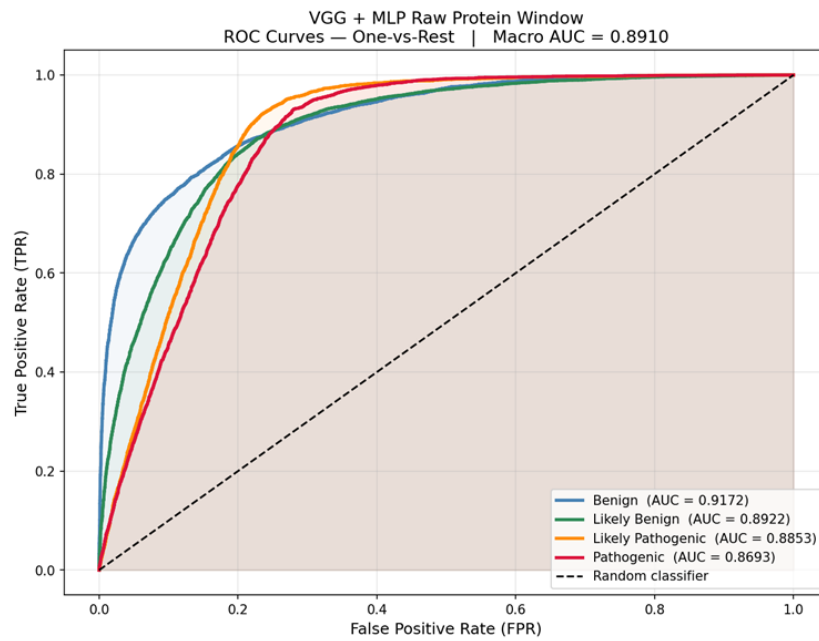


Figure 5.4: One-vs-rest ROC-AUC curves for the Hybrid VGG CNN + MLP model.

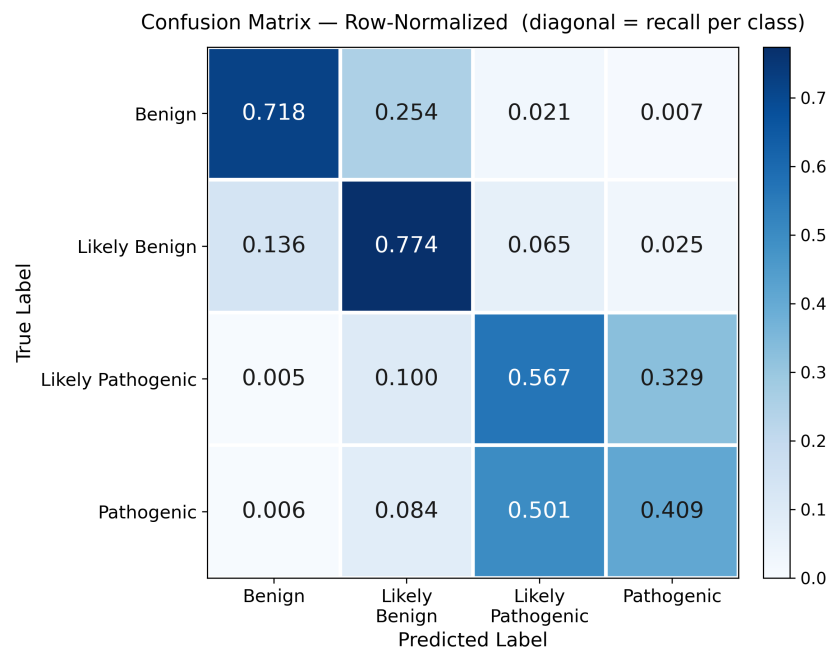


Figure 5.5: Confusion matrix for the Hybrid VGG CNN + MLP model. Values are row-normalised, so each row sums to 1 and the diagonal is the recall for that class.

5.4.3 Per-Class Metrics

Table 5.7: Per-class performance metrics for the Hybrid VGG CNN + MLP model.

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	PR-AUC
Benign	4359	1335	16162	1713	0.7179	0.9237	0.7655	0.7409	0.9172	0.8296
Likely Benign	7342	2290	11793	2144	0.7740	0.8374	0.7623	0.7681	0.8922	0.8329
Likely Pathogenic	2735	2343	16400	2091	0.5667	0.8750	0.5386	0.5523	0.8853	0.5649
Pathogenic	1302	1863	18521	1883	0.4088	0.9086	0.4114	0.4101	0.8693	0.4183

Adding the MLP branch raises the Macro F1 from 0.4065 to 0.6179 and the accuracy from 41.24% to 66.77% (Table 5.7). The gain is uneven across the classes. Benign and Likely Benign reach F1 scores of 0.74 and 0.77, while Likely Pathogenic reaches 0.55 and Pathogenic only 0.41. The model finds just 41% of true Pathogenic variants, so most of the improvement comes from the benign side.

5.5 Performance of the DeepInsight DenseNet121 + MLP Model

The third architecture uses the DeepInsight transformation to turn the protein window into a two-dimensional image. A DenseNet121 backbone reads that image, while the eight variant-level features go through an MLP branch, as in the second model.

5.5.1 Best Hyperparameters (Keras Tuner)

Unlike the other models, the DenseNet121 backbone was too costly to tune on CPU, so this model used a fixed hyperparameter configuration (Table 5.8). The DenseNet121 was trained from scratch: random initialization, with all 121 layers trainable on the DeepInsight images.

5.5.2 Overall Test-Set Metrics

The model reaches an accuracy of 63.58% and a Macro F1 of 0.5744 (Table 5.9). It beats the baseline but stays below the hybrid VGG CNN + MLP model on both measures (Figure 5.6).

5.5.3 Per-Class Metrics

Training the DenseNet121 from scratch recovers the Likely Pathogenic class relative to the baseline, lifting its recall by 0.08 (Table 5.10). The cost falls on the Pathogenic class, whose recall drops below the baseline. The two pathogenic classes remain the hardest to separate.

Table 5.8: Fixed hyperparameters for the DeepInsight DenseNet121 + MLP model. The backbone was trained from scratch and was not tuned with Keras Tuner because of its cost on CPU.

Hyperparameter	Value
DenseNet121 weights	random init, fully trainable
image pooling	GlobalAvg + GlobalMax (concatenated)
MLP dense layers	2
MLP dense units	128
fusion head units	256
activation	ReLU
optimizer	Adam
learning_rate	0.001
dropout	0.40
L2 regularization	0.001
batch size	128

Table 5.9: Overall test-set performance of the DeepInsight DenseNet121 + MLP model.

Metric	Value
Test samples	23,569
Accuracy	63.58%
Balanced Accuracy	56.99%
Macro F1-Score	0.5744
Weighted F1-Score	0.6308
MCC	0.4823
Log Loss	0.8505
Macro ROC-AUC	0.8653
Macro Avg Precision	0.6253

Table 5.10: Per-class performance metrics for the DeepInsight DenseNet121 + MLP model.

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	Avg Prec (PR-AUC)
Benign	4295	1301	16196	1777	0.7073	0.9256	0.7675	0.7362	0.9033	0.8155
Likely Benign	7298	3236	10847	2188	0.7693	0.7702	0.6928	0.7291	0.8569	0.7925
Likely Pathogenic	2457	2410	16333	2369	0.5091	0.8714	0.5048	0.5070	0.8659	0.5407
Pathogenic	936	1636	18748	2249	0.2939	0.9197	0.3639	0.3252	0.8353	0.3526

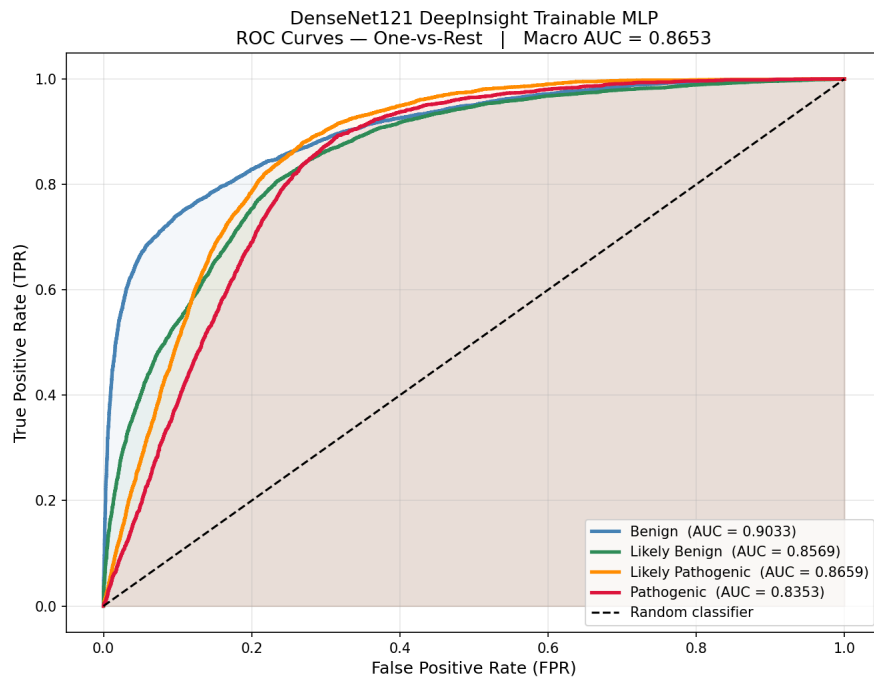


Figure 5.6: ROC-AUC one-vs-rest curves for the DeepInsight DenseNet121 + MLP model.

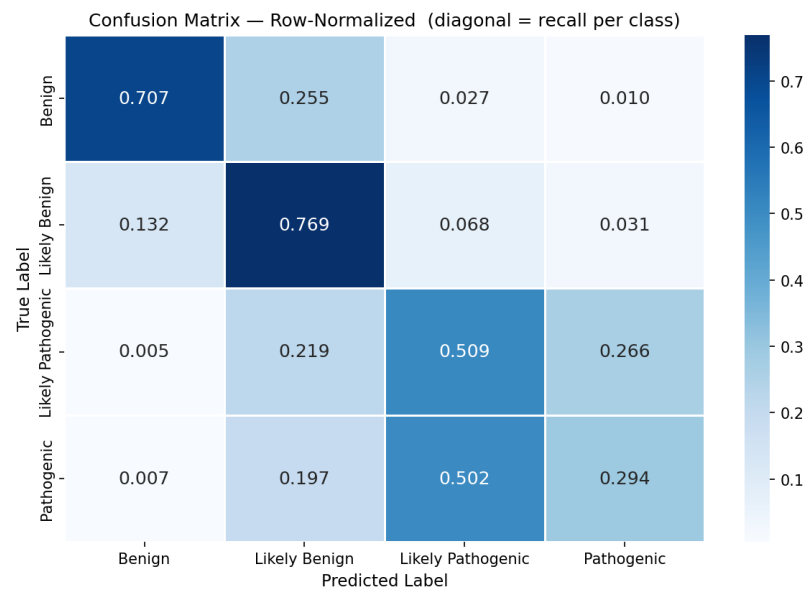


Figure 5.7: Confusion matrix for the DeepInsight DenseNet121 + MLP model. Values are row-normalised.

The confusion matrix shows the main error running from Pathogenic into the neighbouring Likely Pathogenic class (Figure 5.7). Most true Pathogenic variants that miss the diagonal are called Likely Pathogenic. This explains why Pathogenic recall stays low while its specificity stays high: the model rarely labels a non-pathogenic variant as Pathogenic, but it also rarely commits to the Pathogenic label at all.

5.6 Ablation Study: Isolating the Input Sources

The three main models mix two input sources: the structural or DeepInsight image and the eight variant-level features. To measure how much each source contributes, two ablation models each keep a single input. The first is a tabular MLP that sees only the eight features. The second is a DenseNet121 that sees only the DeepInsight image. Both reuse the same seed-42 test set, the same balanced class weights, and the same evaluation protocol as the main models. Two metrics carry the interpretation: recall, the fraction of a class’s true members the model actually finds, and specificity, the fraction of variants outside that class the model correctly rejects.

5.6.1 Tabular MLP (Eight Features Only)

The tabular MLP applies a single hidden layer of 64 ReLU units with dropout 0.3 (Nadam, learning rate 5×10^{-4}) to the eight standardized features, with no image branch. On the test set it reaches 61.49% accuracy, a Macro F1-score of 0.5693, and a Macro ROC-AUC of 0.8663 (Table 5.13). These numbers sit within 0.05 Macro-F1 of the two full fusion models, even though this model discards the entire image.

Table 5.11: Per-class performance metrics for the tabular MLP (eight features only).

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	PR-AUC
Benign	4272	1300	16197	1800	0.7036	0.9257	0.7667	0.7338	0.8934	0.8071
Likely Benign	6790	2497	11586	2696	0.7158	0.8227	0.7311	0.7234	0.8622	0.7957
Likely Pathogenic	1890	2427	16316	2936	0.3916	0.8705	0.4378	0.4134	0.8679	0.5387
Pathogenic	1540	2853	17531	1645	0.4835	0.8600	0.3506	0.4064	0.8418	0.3559

The per-class pattern mirrors the fusion models (Table 5.11): the two benign classes reach F1 scores above 0.72, while the pathogenic classes trail. Pathogenic recall is actually higher than the trainable DenseNet’s (0.2939), which shows that the eight features alone place true Pathogenic variants more often than the image branch does. Specificity stays above 0.82 for every class, so the model rarely over-assigns any single label.

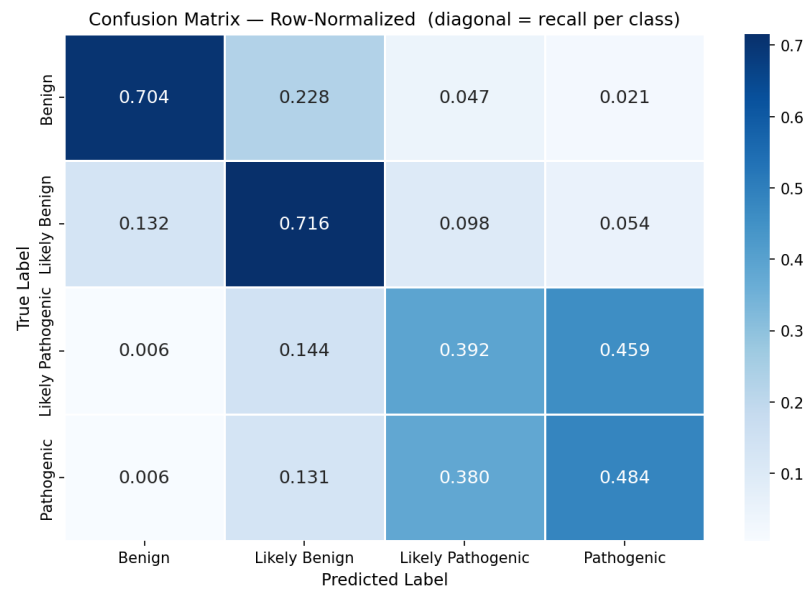


Figure 5.8: Confusion matrix for the tabular MLP (eight features only). Values are row-normalised.

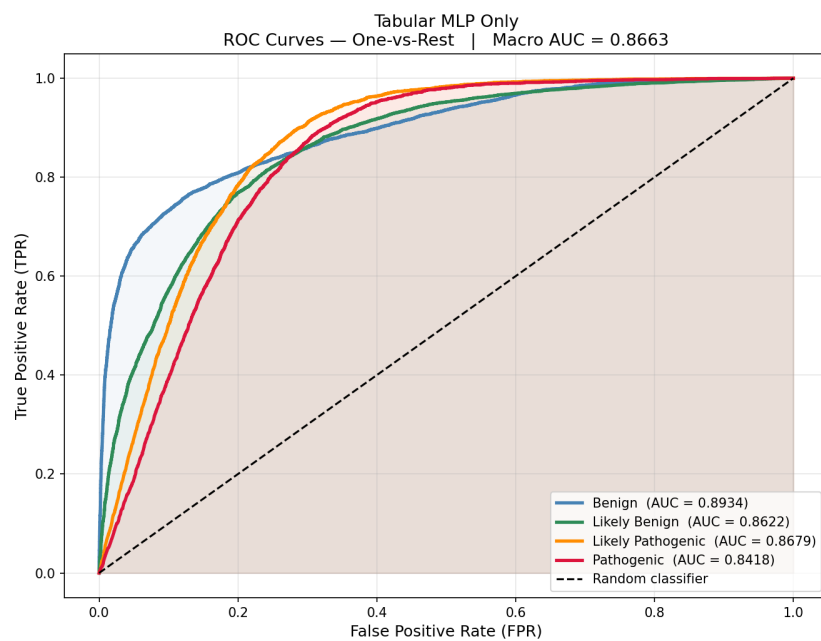


Figure 5.9: One-vs-rest ROC-AUC curves for the tabular MLP (eight features only).

5.6.2 DenseNet CNN-Only (DeepInsight Image Only)

The second ablation trains the same DenseNet121 backbone on the DeepInsight image alone, with the MLP branch removed. Its performance collapses to 30.09% accuracy, a Macro F1-score of 0.2390, an MCC of 0.0541, and a Macro ROC-AUC of 0.5999 (Table 5.13). For a four-class problem, a Macro ROC-AUC near 0.60 and an MCC near zero are close to random guessing.

Table 5.12: Per-class performance metrics for the DenseNet CNN-only model (DeepInsight image only).

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	PR-AUC
Benign	3893	11357	6140	2179	0.6411	0.3509	0.2553	0.3652	0.5727	0.3080
Likely Benign	1851	1681	12402	7635	0.1951	0.8806	0.5241	0.2844	0.5828	0.4787
Likely Pathogenic	1295	3259	15484	3531	0.2683	0.8261	0.2844	0.2761	0.6267	0.2667
Pathogenic	52	181	20203	3133	0.0163	0.9911	0.2232	0.0304	0.6174	0.1809

The per-class numbers expose the failure (Table 5.12). The model defaults to Benign, labelling 15,250 of the 23,569 test variants Benign, so Benign recall looks reasonable while its specificity collapses. At the other extreme it almost never predicts Pathogenic, finding 52 of the 3,185 true cases. The high Pathogenic specificity that follows is meaningless, since a model that never predicts a class cannot produce false positives for it. The DeepInsight image on its own carries almost no class signal.

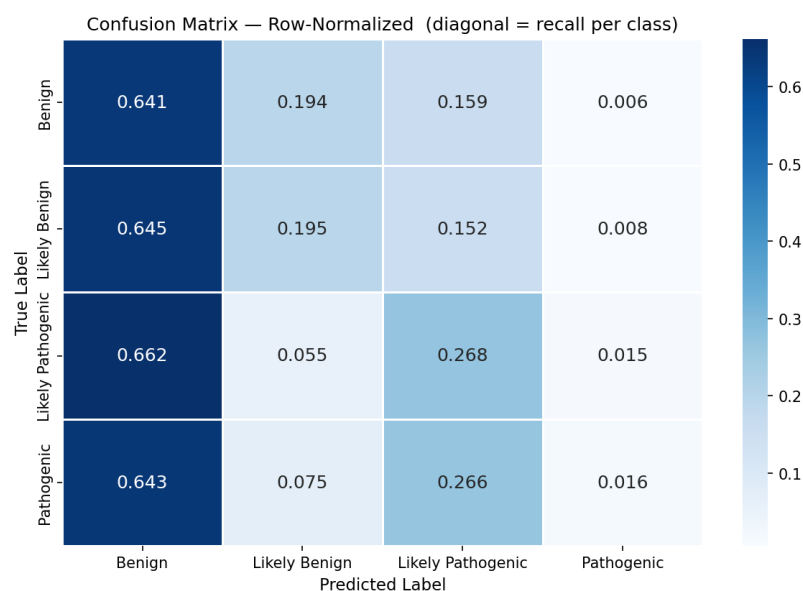


Figure 5.10: Confusion matrix for the DenseNet CNN-only model (DeepInsight image only). Values are row-normalised.

Read together, the two ablations locate the predictive signal. The tabular MLP reaches a Macro F1 of 0.5693 from eight features alone, against 0.6179 for the best fusion model, so those eight features recover 92% of it. The image-only

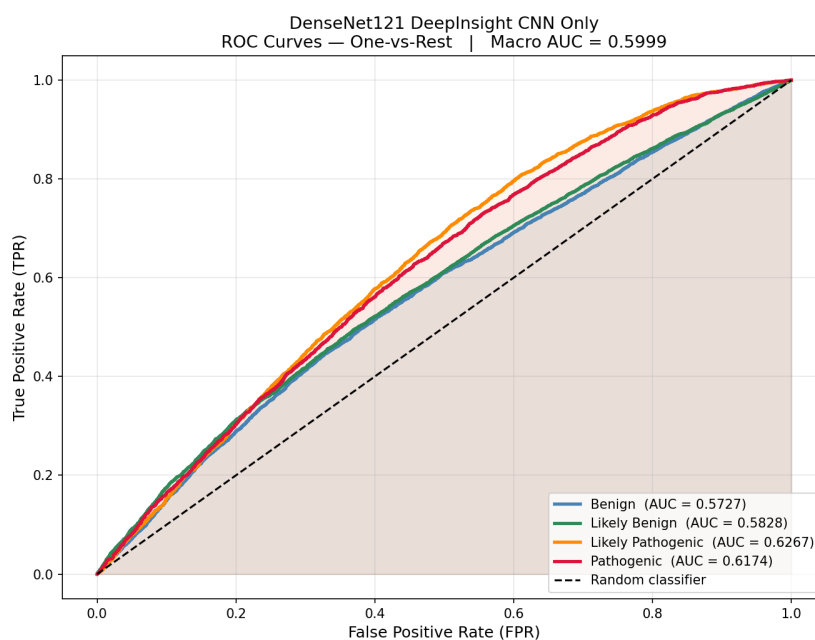


Figure 5.11: One-vs-rest ROC-AUC curves for the DenseNet CNN-only model (DeepInsight image only).

DenseNet falls to a Macro ROC-AUC of 0.5999 and an MCC of 0.0541, close to random. Almost all of the predictive signal therefore comes from the variant-level features rather than from the protein window image. This also explains why Model 3 does not overtake the simpler VGG CNN + MLP: a deeper backbone cannot compensate for an input that carries little class information.

Table 5.13: Overall performance of the five models on the 23,569-variant test set. The dual-input VGG CNN + MLP is best on every metric. Best value per row in bold; for log loss, lower is better.

Metric	VGG	VGG+MLP	DenseNet+MLP	Tabular MLP	DenseNet CNN
Accuracy	0.4124	0.6677	0.6358	0.6149	0.3009
Balanced accuracy	0.4235	0.6168	0.5699	0.5736	0.2802
Macro-F1	0.4065	0.6179	0.5744	0.5693	0.2390
Weighted-F1	0.4143	0.6685	0.6308	0.6198	0.2692
MCC	0.2098	0.5329	0.4823	0.4647	0.0541
Log loss	1.1125	0.7307	0.8505	0.8466	1.3652
Macro-AUC	0.7529	0.8910	0.8653	0.8663	0.5999
Macro-AP	0.4465	0.6614	0.6253	0.6244	0.3086

5.6.3 Feature Importance

To see which inputs the best model actually uses, permutation importance was run on the Hybrid VGG CNN + MLP model, on the same 23,569-variant test set. Each of the 8 dense features and each of the 15 CNN window channels was shuffled on its own, ten times, and the drop in Macro F1 was measured. A bigger drop means the model depends on that input more. Figure 5.12 shows the result.

One feature is far more important than the rest: `log_Composite_AF`. Shuffling it alone drops Macro F1 from 0.6179 to 0.4213, a loss of 0.197. This is about four times bigger than the next feature. Two more dense features come after it: `pssm_score` (0.042) and `n_databases_seen` (0.023). These are the same three features the tabular MLP ablation already pointed to. Now they are confirmed on the model that is actually used.

No single CNN channel is as important as `log_Composite_AF`, but the 15 channels are not useless either. `MMseq2_conservation_score`, the evolutionary conservation channel, is the strongest one (0.019), followed by `ASAquick_normscore` (0.018) and `PSIPRED_helix` (0.016). This makes sense, because how conserved and how exposed a residue is are both known signals for pathogenicity. The `AA_token` channel, the simple integer code for the residue, also costs the model some score when shuffled (0.010). This is worth mentioning next to Section 6.2, where this encoding is said to carry little information: permutation importance shows the trained model still relies on it, even if a better encoding could remove the need for it.

At the bottom, four features barely change the score: `Delta_mass`, `Delta_charge`, `Delta_hydropathy`, and `PTM_Binary_Sum` each drop Macro F1 by less than 0.001. This does not mean the model cannot use them. It only means this trained model does not lean on them much.

5.7 Window Size Results

Table 5.14 reports the four-class window experiment described in Section 4.5.

Table 5.14: Effect of the protein window size on the dual-input VGG + MLP model (test set).

Window (residues)	Macro-F1	Accuracy	Macro-AUC
11	0.607	0.654	0.888
21	0.612	0.658	0.887
31	0.603	0.650	0.890
41	0.609	0.656	0.884
51	0.612	0.659	0.887

The five sizes finish within 0.009 of each other on Macro-F1 and within 0.010 on accuracy, even though the widest window holds nearly five times as many residues as the narrowest. No size is clearly best. The 51-residue window gives the highest accuracy and ties with 21 for the highest Macro-F1, while 31 gives the highest Macro-AUC. On every metric the gap between the best and the worst size stays under one percentage point. Widening the window does not add usable signal, and

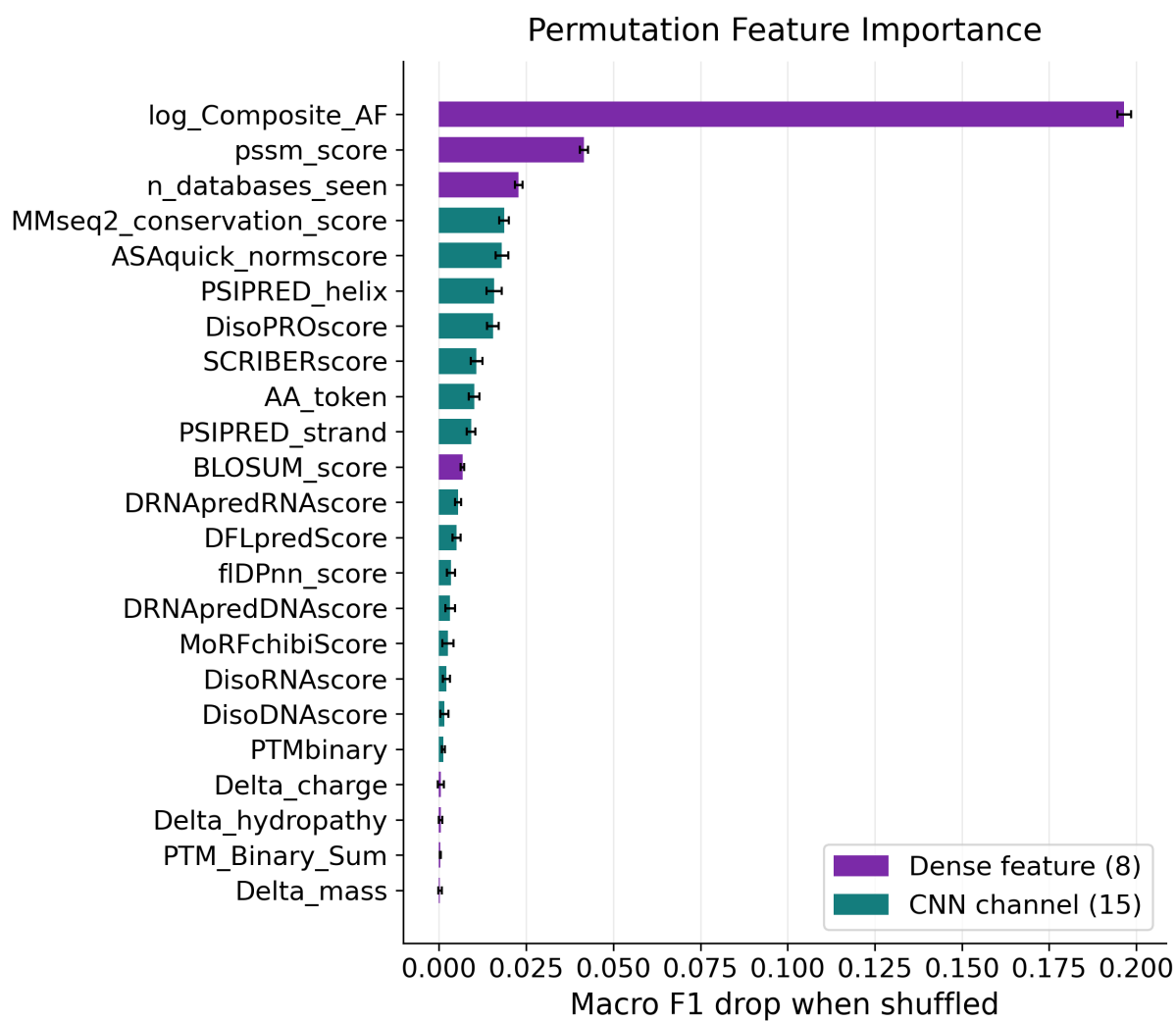


Figure 5.12: Permutation feature importance for the Hybrid VGG CNN + MLP model: drop in Macro F1 when each of the 8 dense features and 15 CNN channels is shuffled on its own.

narrowing it does not remove any. The 31-residue window was kept because it scores highest on Macro-AUC and because every other experiment in this study uses that size. This agrees with the ablation study above, where the variant-level features rather than the protein window carry most of the predictive signal.

The binary sweep covers eight sizes and two models, and it separates the two cases more clearly (Table 5.15). For the fusion model the window barely matters: MCC moves from 0.8217 at five residues to 0.8343 at 31, a spread of 0.013 across a window six times wider, and ROC-AUC stays between 0.969 and 0.973 at every size. For the image-only model the window matters more, and not in a simple way. MCC ranges from 0.5505 at 13 residues to 0.6076 at 21, a spread of 0.057, more than four times the fusion spread, and the best and worst sizes sit next to each other in the sweep rather than at the two ends of it.

That pattern fits the ablation. When the eight variant-level features are present they carry the decision, so changing the window changes little. When they are removed the model has only the window to work with, and its score then follows the window, but the ordering of the sizes looks like noise rather than a trend: 21 residues beats both 17 and 25, and the widest window is not the best. Neither model improves steadily as the window grows, so no size in this range captures a wider biological context that the others miss.

Table 5.15: Effect of the protein window size on the two binary models (test set, 23,569 variants).

Window (residues)	VGG CNN + MLP		VGG CNN (image only)	
	MCC	AUC	MCC	AUC
5	0.8217	0.9697	0.5727	0.8761
7	0.8272	0.9730	0.5780	0.8782
11	0.8278	0.9729	0.5762	0.8772
13	0.8304	0.9721	0.5505	0.8632
17	0.8319	0.9730	0.5758	0.8760
21	0.8320	0.9729	0.6076	0.8855
25	0.8318	0.9693	0.5768	0.8758
31	0.8343	0.9723	0.5797	0.8800

5.8 Binary Classification Results

The four binary models were compared at the window that scored best for each one (Table 5.16).

Plotting the curves together separates the models into three groups (Figure 5.13). The two fusion models run almost on top of each other near the top-left corner. The tabular MLP sits below them but still well clear of chance. The two image-only

Table 5.16: Binary classification performance on the held-out test set of 23,569 variants, of which 34.0% are pathogenic. Each model is shown at its best residue window. Ranked by MCC; best value per column in bold.

Model	Window	MCC	AUC	AP	Sens.	Spec.
VGG CNN + MLP	31	0.8343	0.9723	0.9476	0.9130	0.9305
DeepInsight DenseNet121 + MLP	31	0.8084	0.9694	0.9429	0.9009	0.9184
Tabular MLP (8 features)	–	0.7285	0.9477	0.9012	0.8745	0.8725
VGG CNN (image only)	21	0.6076	0.8855	0.8132	0.7932	0.8300

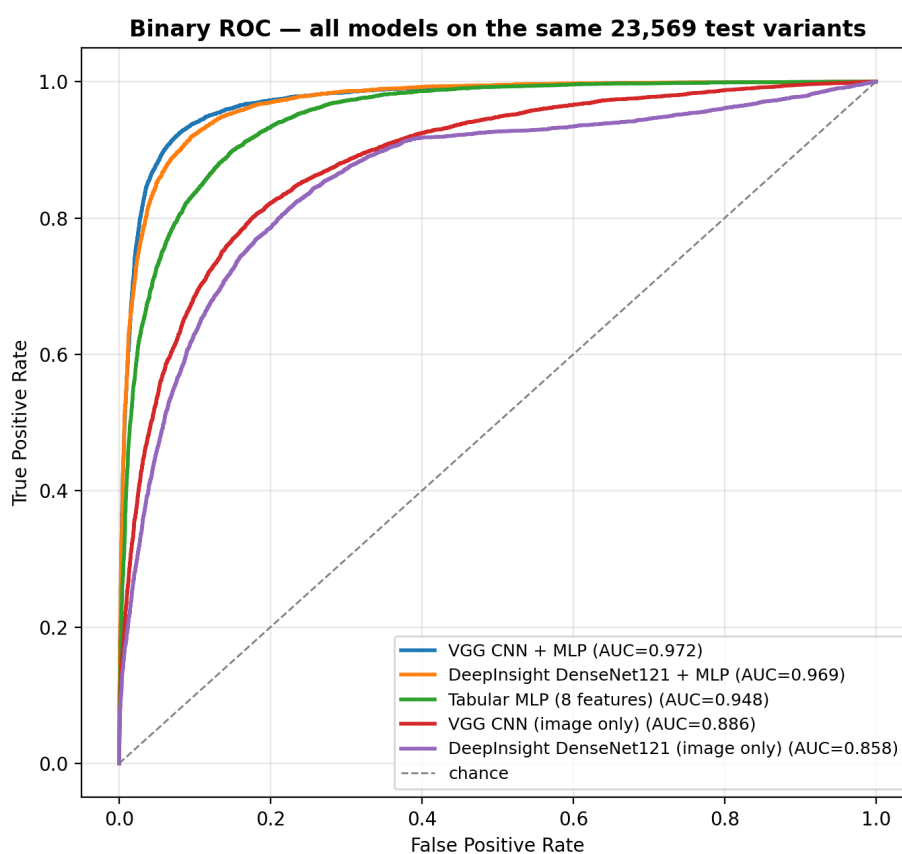


Figure 5.13: Binary ROC curves for every model on the same 23,569 test variants. The figure also includes the DeepInsight image-only ablation, which is not listed in Table 5.16.

models fall away from both, and they stay close to each other even though one is a VGG reading the protein window and the other a DenseNet121 reading a DeepInsight image. What separates the groups is whether a model can see the eight variant-level features, not which architecture reads the window.

The VGG CNN + MLP fusion won again, with an MCC of 0.8343 and a ROC-AUC of 0.9723 at the 31-residue window. It missed 697 of the 8,011 pathogenic variants in the test set and wrongly flagged 1,082 of the 15,558 benign ones. Sensitivity and specificity finish within two points of each other, so the model does not buy one class at the expense of the other. All figures are reported at the default 0.5 threshold; moving the cutoff to 0.59 would raise MCC to 0.8370, a gain of 0.003, so the result does not depend on where the threshold sits.

The two ablations place the signal, as they did on four classes. The fusion model beats the tabular MLP by 0.106 MCC, which is what the protein window adds to the eight variant-level features. It beats the image-only VGG by 0.227 MCC, which is what the eight features add to the window. Both inputs help, and the features help about twice as much.

On four classes the tabular MLP recovered 92% of the fusion model's Macro-F1; here it recovers 87% of the fusion MCC. The protein window therefore counts for more in the binary decision than in the four-class one. Merging Likely Pathogenic with Pathogenic removes the boundary that caused most four-class errors, and on the coarser benign-against-pathogenic split the window features carry more of the remaining work.

The fusion model was also evaluated on the same 34,146 external variants used for the four-class model, without retraining. In this set 18.4% of variants are pathogenic, against 34.0% in training. ROC-AUC rose slightly, from 0.9723 to 0.9792, so the model still separates pathogenic from benign variants at least as well as on the internal test set. MCC fell from 0.8343 to 0.7157, and precision fell further, from 0.8711 to 0.6216: of the 9,788 variants the model called pathogenic, 3,704 were not. This follows the same pattern as the four-class external result described below: the model still separates the two classes well, but it predicts Pathogenic more often than the external class balance supports.

5.9 External Validation Results

The best model, the VGG CNN + MLP fusion, was evaluated on an independent external validation dataset, constructed as described in Section 4.6. The external set is described first, then the results.

The external set skews toward Likely Benign, which rises from 40.2% of the internal data to 78.2% (Figure 5.14). Pathogenic representation drops from 13.5%

to 4.6%. This shift reflects ClinVar submission patterns: routine clinical sequencing of healthy individuals produces large numbers of likely-benign reports, while pathogenic discoveries arrive in smaller batches but represent disproportionately important diagnoses.

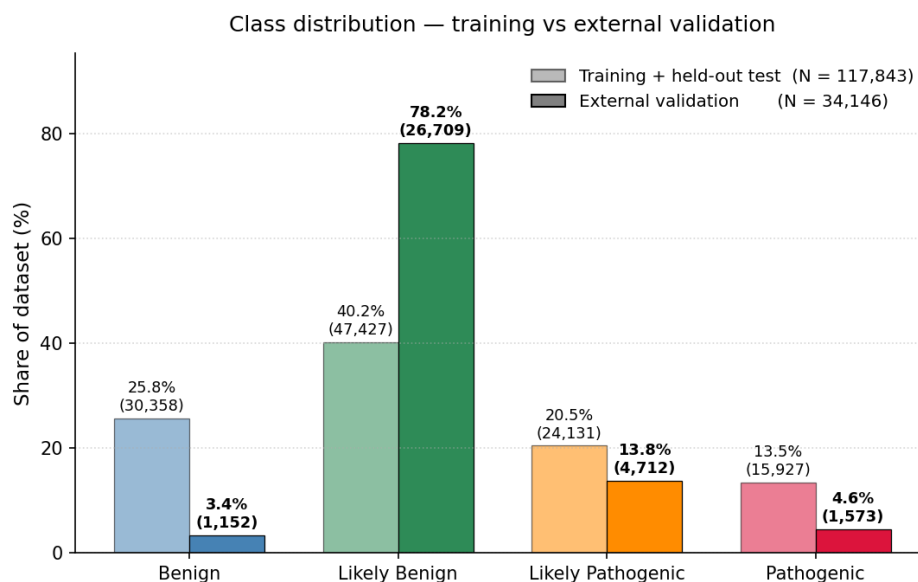


Figure 5.14: Class distribution training and test set compared with the external validation set.

74.7% of variants in the external dataset had a non-zero Composite_AF value, compared with 84.3% in the training dataset. Figure 5.15 shows the distribution of the eight standardized dense features in the training and external datasets. The four substitution features (Δ mass, Δ hydropathy, Δ charge, and BLOSUM62) barely move, since a substitution’s mass, hydropathy, and charge change are fixed physical properties and do not depend on how much a variant has been studied. The three features that do drift, log_Composite_AF, n_databases_seen, and the PSSM score, are exactly the three the tabular MLP relies on most: together they account for 94% of that model’s feature importance. External variants are newer ClinVar submissions, seen in fewer population databases and sitting at less-studied protein positions, so the features the model leans on most are also the ones that shift most between training and deployment.

The model was applied to the 34,146 external variants without retraining, using the scaler fitted on the original training set (Section 4.6).

Table 5.17 sets the external results beside the internal test results for the same model. The Macro ROC-AUC falls by only 0.01, so the model separates unseen variants almost as well as it separates variants from its own dataset.

Two other numbers move in opposite directions. Accuracy rises from 66.77% to 77.39%, while the Macro F1 drops from 0.6179 to 0.4698. Both follow from the

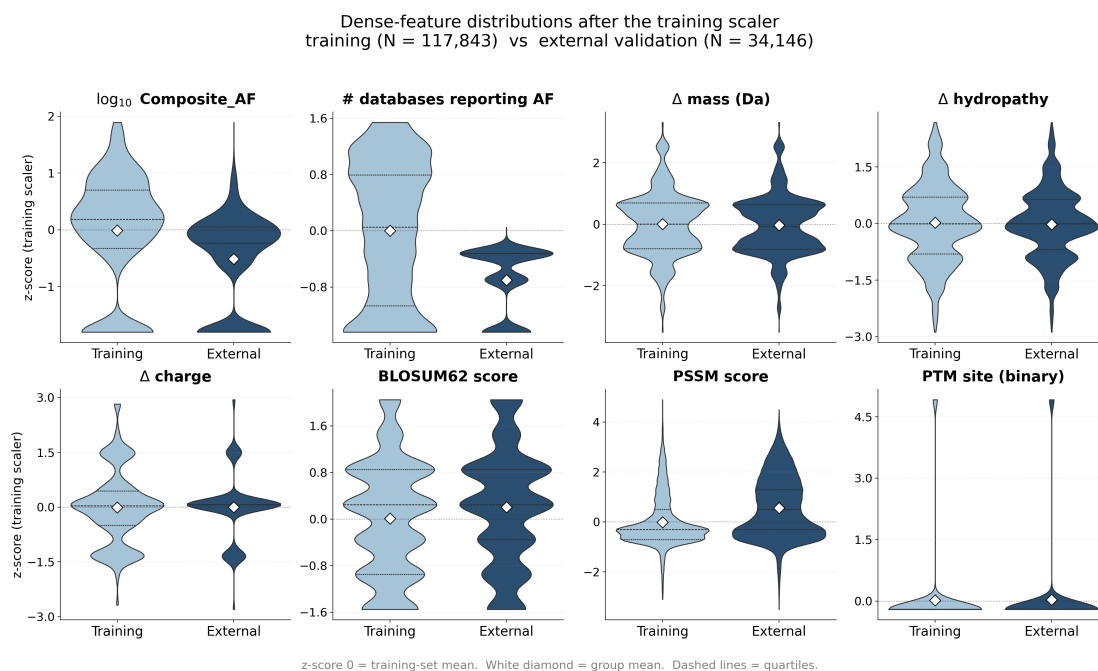


Figure 5.15: Distribution of the eight dense features in the training set and the external validation set, after both were scaled with the StandardScaler fitted on the training data.

class distribution. Likely Benign makes up 78.2% of the external set and the model handles that class well, with an F1 of 0.9056, so it pulls the accuracy up. Macro F1 gives every class the same weight, so the small Benign class, only 3.4% of the set with an F1 of 0.2005, pulls the average down. The higher accuracy therefore reflects the composition of the external set rather than a better model.

Table 5.17: Overall performance of the Hybrid VGG CNN + MLP model on the internal held-out test set and on the external validation set.

Metric	Internal test	External set
Test samples	23,569	34,146
Accuracy	66.77%	77.39%
Balanced Accuracy	61.68%	51.14%
Macro F1-Score	0.6179	0.4698
Weighted F1-Score	0.6685	0.7978
MCC	0.5329	0.4782
Log Loss	0.7307	0.6181
Macro ROC-AUC	0.8910	0.8811
Macro Avg Precision	0.6614	0.4785

The per-class numbers explain this (Table 5.18). ROC-AUC stays high for every class, above 0.76 everywhere and above 0.89 for three of the four, so the model still separates the classes. What changes is how often it predicts each one. The models were trained with class weights that raise the cost of missing a Pathogenic

variant, because Pathogenic was the rarest class in training at 13.5%. The model therefore predicts Pathogenic freely: it gives that label to 11.6% of the external variants, close to the rate it learned. But only 4.6% of the external variants are Pathogenic, so most of those predictions are wrong and precision falls to 0.1827. Recall does not drop; the model still finds pathogenic variants. The class weighting that helped on the internal test set works against it here, because the external set has a different class balance.

Table 5.18: Per-class performance metrics of the Hybrid VGG CNN + MLP model on the external validation set.

Class	TP	FP	TN	FN	Recall	Specificity	Precision	F1	ROC-AUC	PR-AUC
Benign	208	715	32,279	944	0.1806	0.9783	0.2254	0.2005	0.7625	0.1569
Likely Benign	22,910	978	6,459	3,799	0.8578	0.8685	0.9591	0.9056	0.9424	0.9802
Likely Pathogenic	2,584	2,799	26,635	2,128	0.5484	0.9049	0.4800	0.5119	0.9237	0.5644
Pathogenic	722	3,230	29,343	851	0.4590	0.9008	0.1827	0.2614	0.8960	0.2125

Figure 5.16 shows that most classification errors occurred between neighbouring classes. In particular, many Benign variants were predicted as Likely Benign, while some Pathogenic variants were classified as Likely Pathogenic. This suggests that the model generally preserves the distinction between benign and pathogenic variants but has greater difficulty separating adjacent clinical significance categories.

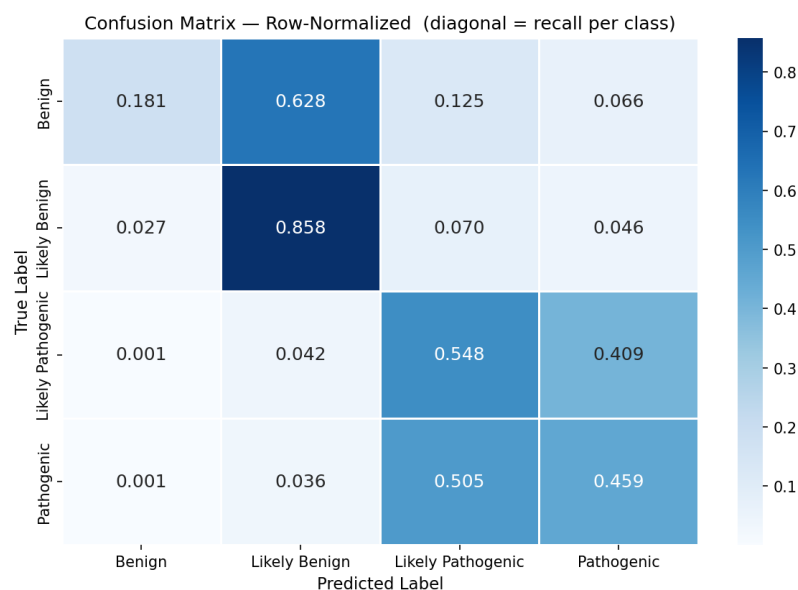


Figure 5.16: Confusion matrix of the Hybrid VGG CNN + MLP model on the external validation set.

Figure 5.17 shows the one-vs-rest ROC curves.

5.9.1 Binary External Validation

The binary model was tested on the same external set. It was not retrained, and its decision threshold was not changed. Table 5.19 puts the result next to the

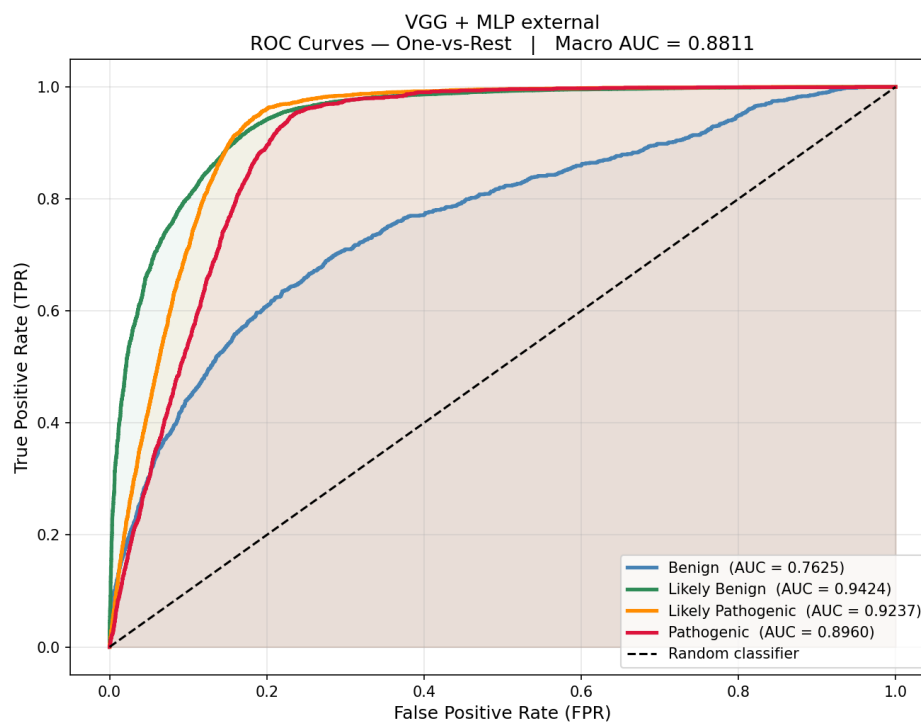


Figure 5.17: One-vs-rest ROC curves of the Hybrid VGG CNN + MLP model on the external validation set.

internal test set.

The model still separates the two classes well. Its ROC-AUC is 0.9792 on the external set, a little above the 0.9723 it reached on the internal test set, and balanced accuracy stays almost the same, 91.75% against 92.17%. Sensitivity even rises, to 96.80%: the model finds 6,084 of the 6,285 true pathogenic variants and misses only 201 (Figure 5.18).

Precision drops from 87.11% to 62.16%. This comes from the mix of classes, not from a weaker model. Only 18.4% of the external variants are pathogenic, against 34.0% in the internal test set. When true pathogenic variants are rarer, there are more benign variants that the model can mistake for pathogenic, so precision falls even though the ranking is just as good. ROC-AUC does not depend on the threshold or on how many variants of each class are present, and it is slightly higher on the external set (Figure 5.19).

5.10 Discussion

This chapter interprets the results reported in Chapter 5. The pattern is consistent across all three main architectures: adding the eight variant-level features to the VGG image branch raised Macro-F1 from 0.4065 to 0.6179, and the tabular MLP using only those eight features nearly matched both fusion models on its own.

Table 5.19: Performance of the binary VGG CNN + MLP model on the internal held-out test set and on the independent external validation set. The external set carries a markedly lower pathogenic rate (18.4% against 34.0%), which affects the threshold-dependent metrics far more than the ranking metrics.

Metric	Internal test	External
Test variants	23,569	34,146
Pathogenic rate	34.0%	18.4%
MCC	0.8343	0.7157
ROC-AUC	0.9723	0.9792
Average precision	0.9476	0.9216
Accuracy	92.45%	88.56%
Balanced accuracy	92.17%	91.75%
Sensitivity	91.30%	96.80%
Specificity	93.05%	86.71%
Precision	87.11%	62.16%
F1 (pathogenic)	89.16%	75.70%
Brier score	0.0578	0.0872

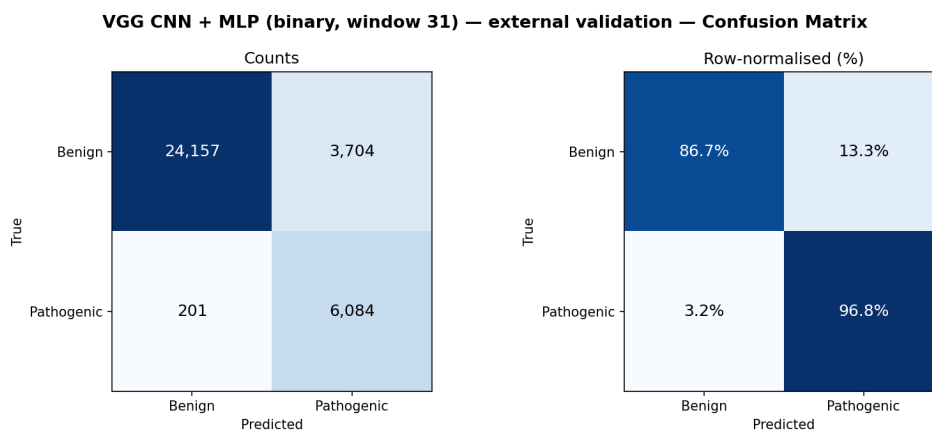


Figure 5.18: Confusion matrix of the binary VGG CNN + MLP model on the external validation set, as counts and row-normalised percentages.

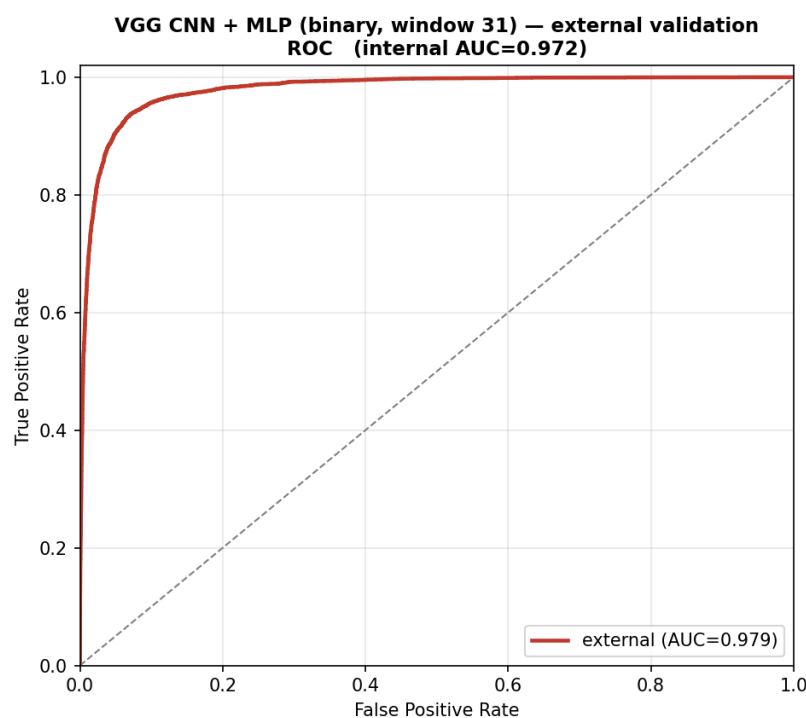


Figure 5.19: ROC curve of the binary VGG CNN + MLP model on the external validation set.

The pattern holds twice, so the improvement is not a property of one architecture. Feature integration, not network depth, is what moved the score. Once variants of uncertain significance are removed, ClinVar labels the remaining variants in four categories, and the models were trained on those four. The results show why the intermediate categories are hard to predict. The strongest source of error was the boundary between Likely Pathogenic and Pathogenic. Both Model 2 and Model 3 frequently placed Pathogenic variants into Likely Pathogenic, leaving Pathogenic with the lowest recall of the four classes in every model. Grouping the four labels into two, as most published predictors do, removes exactly this boundary, which is why the binary task is easier. All models classify benign categories more accurately than pathogenic ones. The best model achieved F1-scores of 0.74 for Benign and 0.77 for Likely Benign, whereas performance was 0.55 for Likely Pathogenic and 0.41 for Pathogenic variants. The factor that explains this is the data imbalance.

Class weighting is a structural requirement here, not a cosmetic choice. Retraining the VGG CNN + MLP model without the balanced class weights, under an otherwise identical setup, raised raw accuracy slightly (from 66.77% to 67.17%) but lowered every balance-aware metric: Macro-F1 fell from 0.6179 to 0.6001 and balanced accuracy from 0.6168 to 0.6034. The cost fell on the rarest class, where it matters most: Pathogenic recall collapsed from 0.4088 to 0.2261, so the unweighted model missed 77% of true Pathogenic variants while still scoring 0.67

accuracy. This confirms that accuracy is misleading under this imbalance, and that balanced weighting is what lets the model learn the minority Pathogenic class at all. The DeepInsight with DenseNet121 framework was introduced to investigate whether advanced image representations could improve classification performance. Despite its architectural complexity, Model 3 did not outperform the simpler Model 2. DeepInsight was designed for datasets with hundreds or thousands of independent features, where t-SNE can arrange correlated features into coherent regions and a CNN can read them as texture. The protein input does not meet these conditions. Flattening the 31×15 matrix yields 465 values, but these are 31 sequence positions repeated across 15 channels, so they are highly redundant rather than independent. t-SNE maps them onto a sparse 32×32 grid in which most pixels stay empty, and adjacency in that grid reflects statistical similarity, not the sequence order that actually carries biological meaning. A DenseNet121 expects the dense edges and textures of a natural photograph; given a sparse, arbitrarily arranged 32×32 image, its convolutional priors have little to lock onto, and with no natural-image pretraining it must learn everything from a weak signal. The ablation makes the consequence explicit (Table 5.13): the image-only DenseNet reaches a Macro ROC-AUC of 0.5999 and an MCC of 0.0541, close to random guessing. Because the image branch carries almost no class signal, fusing it with the MLP branch adds little: Model 3 performs at about the level of the tabular features alone, and below Model 2, whose VGG branch reads the raw protein matrix directly rather than a DeepInsight transform of it. These findings suggest that increasing architectural complexity alone does not guarantee improved pathogenicity prediction performance [47].

5.10.1 External Dataset

Model 2 was evaluated on 34,146 ClinVar variants submitted after the training data was collected. Macro-F1 fell from 0.6179 to 0.4698, a drop of 0.148. Macro ROC-AUC held at 0.8811, close to the internal 0.8910. The model still separates the classes; it predicts them at the wrong rate.

The drop in Macro F1 comes mostly from the class distribution. The external dataset was heavily dominated by the Likely Benign class (78.2%), while Pathogenic and Benign variants were less represented. Accuracy rose to 77.39% as a result, but that rise reflects the class imbalance, not a genuine gain in predictive performance.

ROC-AUC stayed above 0.76 for every class and above 0.89 for three of them (Section 5.9). Figure 5.15 shows that the allele-frequency features and the PSSM score are the ones that shift most between the training and external sets, while the physicochemical substitution features barely move.

Overall, the external validation shows that the model separates benign from pathogenic variants on data 22 months removed from training, but its raw predictions reflect the class balance it was trained on rather than the class balance of the external set.

5.10.2 Significance of DescribePROT Features

No prior study appears to have evaluated the amino acid DescribePROT structural and functional features. Previous studies have shown that structural information improves variant classification [8, 78], but none have integrated the per-residue DescribePROT feature set into a deep learning framework for ACMG/AMP four-class classification.

5.10.3 Comparison with Previous Studies

This thesis addresses several important gaps in the existing literature (Table 5.20).

Table 5.20: Comparison between previous studies and the contributions of this thesis.

Gap in Previous Studies	Contribution of This Thesis
Heavy focus on binary classification	Four-class clinical classification
Limited use of residue-level structural descriptors	First integration of DescribePROT residue-level features
Small gene-specific datasets	Large multi-gene ClinVar dataset
Lack of fair model comparison	Standardized evaluation of five architectures
Focus on single architectures	Comparative analysis of CNN, CNN-MLP fusion, and DeepInsight-DenseNet, with single-input ablations

Figure 5.20 presents an indicative comparison between the proposed models and published pathogenicity predictors on a binary benign-versus-pathogenic classification task. Models 2 and 3 achieved AUC values of 0.972 and 0.969, respectively. The improvement from four-class Macro F1 (0.6179) to binary AUC (0.972) reflects the elimination of the Likely Pathogenic versus Pathogenic boundary, which accounted for the majority of classification errors in the four-class setting. AlphaMissense [8] achieves a binary ClinVar AUC of 0.940 without direct ClinVar training, using AlphaFold structure and evolutionary multiple sequence alignments at proteome scale. The evaluation conditions differ, so the two numbers cannot be compared directly. What the figure does show is that a model built only from per-residue DescribePROT features and eight variant-level annotations reaches this range on its own test set.

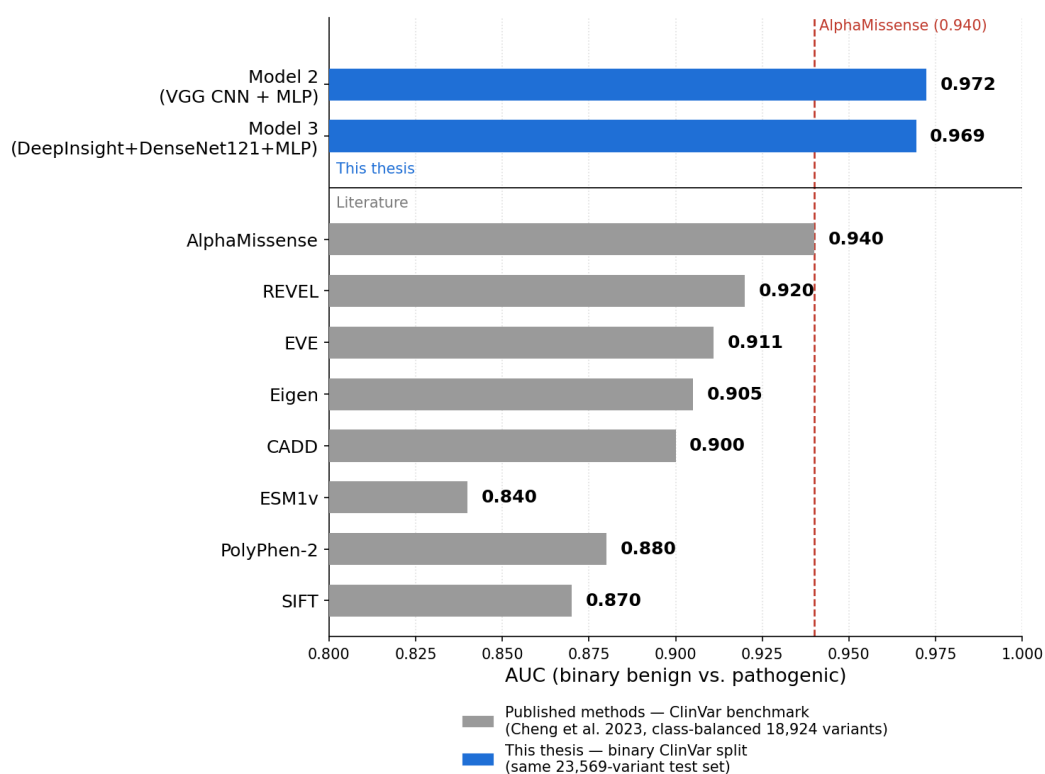


Figure 5.20: Indicative comparison of binary benign-versus-pathogenic AUC between the proposed models and published pathogenicity predictors.

5.10.4 Limitations

Four design choices constrain the four-class results and shape how they should be read.

The first is the compound-label mapping. ClinVar submits some variants under two-part labels such as “Benign/Likely benign” or “Pathogenic/Likely pathogenic.” These affect 13,933 variants, roughly 12% of the dataset. The mapping sends the 8,269 “Benign/Likely benign” variants to Benign and the 5,664 “Pathogenic/Likely pathogenic” variants to Likely Pathogenic. The rule is asymmetric: it rounds every compound label toward the less severe class, which blurs the two boundaries the models already find hardest (Benign versus Likely Benign, and Likely Pathogenic versus Pathogenic) and starves the Pathogenic class, since the 5,664 “Pathogenic/Likely pathogenic” variants land in Likely Pathogenic rather than Pathogenic. This choice is harmless for the binary formulation but weakens the four-class Pathogenic recall reported throughout.

The second is the amino-acid encoding. In the 31×15 matrix, the first channel stores the identity of each wild-type residue as an integer from 1 to 20. A convolutional filter reads this channel as a continuous number, so it treats the amino acids as if they lay on a scale: residue 20 appears ten times larger than residue 2, and a substitution from residue 5 to residue 6 looks smaller than one

from residue 5 to residue 20. This ordering is an artefact of alphabetical numbering and has no biochemical basis, since two amino acids with adjacent indices can differ sharply in size, charge, or hydrophobicity. The network therefore cannot read residue identity as a meaningful category from this channel. A one-hot encoding, which gives each amino acid its own binary channel, or a learned embedding, which places biochemically similar residues near each other, would remove the false ordering and let the model treat each residue as a distinct token. In the present design the substitution itself is instead described numerically in the dense vector through the BLOSUM score and the mass, hydropathy, and charge deltas, which is one reason the MLP branch carries most of the predictive signal.

The third is the window size. The window-size experiment retrained the VGG CNN + MLP model at windows of 11, 21, 31, 41, and 51 residues under identical conditions. Macro-F1 varied by less than 0.01 across the whole range (0.6034 to 0.6122), so 31 residues is a reasonable central choice rather than a tuned optimum. The insensitivity is consistent with the ablation finding: the signal is local and dominated by the eight tabular features, so widening the structural window adds little.

The fourth concerns the comparison with published predictors. The binary AUC values placed alongside this model (Figure 5.20) come from each tool's own publication on a different dataset, not from a shared held-out benchmark, so the comparison is indicative rather than head-to-head. AlphaMissense illustrates the gap: it is trained at proteome scale on every possible human missense substitution, drawing on AlphaFold2 structure and deep multiple sequence alignments, and it is not fitted to ClinVar labels, whereas this model is trained and tested on a single 23,569-variant ClinVar split. The two are therefore evaluated under different data, different label sources, and different train/test conditions. A fair comparison would require running every predictor on the same held-out test set, which is left to future work.

5.11 Summary

The results show that combining variant-level features with the protein window classifies variants better than either input on its own. The VGG CNN + MLP model performs best, with a Macro F1 of 0.6179 and an accuracy of 66.77% (Table 5.13). The ablations show where the signal sits: the eight variant-level features alone reach a Macro F1 of 0.5693, while the protein window image alone falls close to random. Adding the image to the features gives a further gain, but the features carry most of the information. Across every model, separating Likely Pathogenic from Pathogenic stayed the hardest part of the task, and it is the

clearest direction for future work.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis set out to classify germline missense variants by combining clinical labels from ClinVar with per-residue protein descriptors from DescribePROT. Each variant was represented in two ways: a 31×15 matrix of the protein window around the mutation, and an eight-feature vector describing the variant itself. Five deep learning models were trained and compared on the same 23,569-variant test set.

The dual-input VGG CNN with an MLP branch performed best, reaching a Macro F1 of 0.6179, an accuracy of 66.77% and a Macro ROC-AUC of 0.8910. On an independent set of 34,146 ClinVar variants added after the training data was collected, and never used during training or tuning, the same model held a Macro ROC-AUC of 0.8811.

The ablation study locates the predictive signal. The eight variant-level features alone reach a Macro F1 of 0.5693, which is 92% of the best fusion model, while the DeepInsight image alone falls close to random with a Macro ROC-AUC of 0.5999. Adding the protein window to the features helps, but the features carry most of the information. A deeper backbone does not compensate for an input that holds little class signal: the DenseNet121 model, with far more parameters than the VGG branch, did not overtake it.

Two findings hold across every model. Benign categories are predicted more accurately than pathogenic ones, and the boundary between Likely Pathogenic and Pathogenic is where the models fail most often. Pathogenic had the lowest recall of the four classes in every architecture tested.

The contribution of this work is the representation rather than the architecture. Per-residue structural and functional descriptors, already computed for whole proteomes and stored in DescribePROT, can be arranged as a spatial window and combined with variant-level annotation to predict pathogenicity, without the

multiple sequence alignments or solved structures that most existing tools require.

6.2 Future Work

Several directions follow from these results.

Because VUS are unlabelled, they cannot be used through supervised learning; future work could instead label them with functional-assay data such as deep mutational scanning, or exploit them through semi-supervised and self-supervised learning.

The amino-acid identity, encoded here as a single integer channel, could be replaced by a one-hot encoding or a learned embedding so that the network reads each residue as a distinct token rather than a magnitude. Because the ablation study showed the DeepInsight image branch added little, future models should invest in richer feature integration rather than image transforms, for example using protein language model embeddings such as ESM in place of, or alongside, the hand-crafted features.

The Pathogenic class remained the hardest to predict across all models, so techniques that target minority-class recall, such as focal loss or resampling, are worth exploring, together with calibrated probability outputs that would let predictions carry a confidence suitable for clinical triage.

Future work could also include three-dimensional protein representations generated by AlphaFold and graph neural network architectures capable of modelling residue interactions and structural neighborhoods, and support alternative splice isoforms rather than relying exclusively on canonical UniProt sequences. These extensions would improve coverage of clinically relevant variants and reduce dataset exclusion.

Bibliography

- [1] R. L. Nussbaum, R. R. McInnes, and H. F. Willard, *Thompson & Thompson Genetics in Medicine*, 8th ed. Elsevier, 2015.
- [2] K. Eilbeck, A. Quinlan, and M. Yandell, “Settling the score: variant prioritization and Mendelian disease,” *Nature Reviews Genetics*, vol. 18, no. 10, pp. 599–612, 2017.
- [3] M. J. Landrum *et al.*, “Clinvar: improving access to variant interpretations and supporting evidence,” *Nucleic Acids Research*, vol. 46, no. D1, pp. D1062–D1067, 2018.
- [4] M. J. Landrum, S. Chitipiralla, G. R. Brown, C. Chen, B. Gu, J. Hart, D. Hoffman, W. Jang, K. Kaur, C. Liu, V. Lyoshin, Z. Maddipatla, R. Maiti, J. Mitchell, N. O’Leary, G. R. Riley, W. Shi, G. Zhou, V. Schneider, D. Maglott, J. B. Holmes, and B. L. Kattman, “Clinvar: updates to support classifications of both germline and somatic variants,” *Nucleic Acids Research*, vol. 53, no. D1, pp. D1313–D1321, 2025.
- [5] L. M. Starita, N. Ahituv, M. J. Dunham, J. O. Kitzman, F. P. Roth, G. Seelig, J. Shendure, and D. M. Fowler, “Variant interpretation: Functional assays to the rescue,” *American Journal of Human Genetics*, vol. 101, no. 3, pp. 315–325, 2017.
- [6] P. C. Ng and S. Henikoff, “Sift: Predicting amino acid changes that affect protein function,” *Nucleic Acids Research*, vol. 31, no. 13, pp. 3812–3814, 2003.
- [7] I. A. Adzhubei *et al.*, “A method and server for predicting damaging missense mutations,” *Nature Methods*, vol. 7, no. 4, pp. 248–249, 2010.
- [8] J. Cheng, G. Novati, J. Pan, C. Bycroft, A. Žemgulytė, T. Applebaum, A. Pritzel, L. H. Wong, M. Zielinski, T. Sargeant, R. G. Schneider, A. W.

- Senior, J. Jumper, D. Hassabis, P. Kohli, and Ž. Avsec, “Accurate proteome-wide missense variant effect prediction with AlphaMissense,” *Science*, vol. 381, no. 6664, p. eadg7492, 2023.
- [9] N. Brandes, G. Goldman, C. H. Wang, C. J. Ye, and V. Ntranos, “Genome-wide prediction of disease variant effects with a deep protein language model,” *Nature Genetics*, vol. 55, no. 9, pp. 1512–1522, 2023.
- [10] S. Basu, B. Zhao, B. Biró, E. Faraggi, J. Gsponer, G. Hu, A. Kloczkowski, N. Malhis, M. Mirdita, J. Söding, M. Steinegger, D. Wang, K. Wang, D. Xu, J. Zhang, and L. Kurgan, “DescribePROT in 2023: more, higher-quality and experimental annotations and improved data download options,” *Nucleic Acids Research*, vol. 52, no. D1, pp. D426–D433, 01 2024.
- [11] J. Zhou *et al.*, “Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk,” *Nature Genetics*, vol. 50, pp. 1171–1179, 2018.
- [12] S. Richards *et al.*, “Standards and guidelines for the interpretation of sequence variants,” *Genetics in Medicine*, vol. 17, no. 5, pp. 405–424, 2015.
- [13] K. J. Karczewski *et al.*, “The mutational constraint spectrum quantified from variation in 141,456 humans,” *Nature*, vol. 581, pp. 434–443, 2020.
- [14] D. Taliun *et al.*, “Sequencing of 53,831 diverse genomes from the nhlbi topmed program,” *Nature*, vol. 590, pp. 290–299, 2021.
- [15] 1000 Genomes Project Consortium, “A global reference for human genetic variation,” *Nature*, vol. 526, pp. 68–74, 2015.
- [16] S. Henikoff and J. G. Henikoff, “Amino acid substitution matrices from protein blocks,” *Proceedings of the National Academy of Sciences*, vol. 89, no. 22, pp. 10 915–10 919, 1992.
- [17] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, “DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture,” *Scientific Reports*, vol. 9, no. 1, p. 11399, 2019.
- [18] E. S. Lander *et al.*, “Initial sequencing and analysis of the human genome,” *Nature*, vol. 409, no. 6822, pp. 860–921, 2001.
- [19] C. Branden and J. Tooze, *Introduction to Protein Structure*, 2nd ed. Garland Publishing, 1999.

- [20] B. Alberts *et al.*, *Molecular Biology of the Cell*, 6th ed. Garland Science, 2014.
- [21] N. Orr and S. Chanock, “Common genetic variation and human disease,” *Advances in Genetics*, vol. 62, pp. 1–32, 2008.
- [22] The UniProt Consortium, “Uniprot: the universal protein knowledgebase in 2023,” *Nucleic Acids Research*, vol. 51, no. D1, pp. D523–D531, 2023.
- [23] B. Zhao, A. Katuwawala, C. J. Oldfield, A. K. Dunker, E. Faraggi, J. Gsponer, A. Kloczkowski, N. Malhis, M. Mirdita, Z. Obradovic, and L. Kurgan, “DescribePROT: database of amino acid-level protein structure and function predictions,” *Nucleic Acids Research*, vol. 49, no. D1, pp. D298–D308, 01 2021.
- [24] D. G. MacArthur *et al.*, “Guidelines for investigating causality of sequence variants in human disease,” *Nature*, vol. 508, no. 7497, pp. 469–476, 2014.
- [25] NHLBI GO Exome Sequencing Project, “Exome variant server, esp6500si-v2 release,” <https://evs.gs.washington.edu/EVS/>, 2012, 6,503 samples: 4,300 European American and 2,203 African American.
- [26] NHLBI Exome Sequencing Project, “Guidelines for large-scale sequence-based complex trait association studies: Lessons learned from the nhlbi exome sequencing project,” *The American Journal of Human Genetics*, vol. 99, no. 4, pp. 791–801, 2016.
- [27] M. Lek *et al.*, “Analysis of protein-coding genetic variation in 60,706 humans,” *Nature*, vol. 536, pp. 285–291, 2016.
- [28] D. T. Jones, “Protein secondary structure prediction based on position-specific scoring matrices,” *Journal of Molecular Biology*, vol. 292, no. 2, pp. 195–202, 1999.
- [29] P. E. Wright and H. J. Dyson, “Intrinsically disordered proteins in cellular signalling and regulation,” *Nature Reviews Molecular Cell Biology*, vol. 16, no. 1, pp. 18–29, 2015.
- [30] G. Hu *et al.*, “fldpnn: Accurate intrinsic disorder prediction with putative propensities of disorder functions,” *Nature Communications*, vol. 12, p. 4438, 2021.
- [31] F. Meng and L. Kurgan, “Dfipred: High-throughput prediction of disordered flexible linker regions in protein sequences,” *Bioinformatics*, vol. 32, no. 12, pp. i341–i350, 2016.

- [32] V. N. Uversky, “The triple power of D³: Protein intrinsic disorder in degenerative diseases,” *Frontiers in Bioscience*, vol. 19, no. 2, pp. 181–258, 2014.
- [33] B. Rost and C. Sander, “Conservation and prediction of solvent accessibility in protein families,” *Proteins: Structure, Function, and Bioinformatics*, vol. 20, no. 3, pp. 216–226, 1994.
- [34] C. Savojardo, M. Manfredi, P. L. Martelli, and R. Casadio, “Solvent accessibility of residues undergoing pathogenic variations in humans: From protein structures to protein sequences,” *Frontiers in Molecular Biosciences*, vol. 7, p. 626363, 2021.
- [35] S. Jones and J. M. Thornton, “Principles of protein-protein interactions,” *Proceedings of the National Academy of Sciences*, vol. 93, no. 1, pp. 13–20, 1996.
- [36] E. Faraggi, T. Zhang, Y. Yang, L. Kurgan, and Y. Zhou, “Accurate single-sequence prediction of solvent accessible surface area using automated neural networks,” *The Open Bioinformatics Journal*, vol. 8, no. 1, pp. 1–9, 2014.
- [37] J. A. Marsh and S. A. Teichmann, “Structure, dynamics, assembly, and evolution of protein complexes,” *Annual Review of Biochemistry*, vol. 84, pp. 551–575, 2015.
- [38] N. Malhis, M. Jacobson, and J. Gsponer, “Morfchibi system: software tools for the identification of morfs in protein sequences,” *Nucleic Acids Research*, vol. 44, no. W1, pp. W488–W493, 2016.
- [39] J. Reimand, O. Wagih, and G. D. Bader, “Evolutionary constraint and disease associations of post-translational modification sites in human genomes,” *PLOS Genetics*, vol. 11, no. 1, p. e1004919, 2015, PMID: 25611800.
- [40] M. Gribskov, R. Lüthy, and D. Eisenberg, “Profile analysis,” *Methods in Enzymology*, vol. 183, pp. 146–159, 1990.
- [41] M. Steinegger and J. Söding, “Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets,” *Nature Biotechnology*, vol. 35, no. 11, pp. 1026–1028, 2017.
- [42] A. L. Lehninger, D. L. Nelson, and M. M. Cox, *Principles of Biochemistry*, 7th ed. W. H. Freeman, 2017.

- [43] A. Niroula and M. Vihinen, “Variation interpretation predictors: Principles, types, performance, and choice,” *Human Mutation*, vol. 37, no. 6, pp. 579–597, 2016.
- [44] J. Kyte and R. F. Doolittle, “A simple method for displaying the hydropathic character of a protein,” *Journal of Molecular Biology*, vol. 157, no. 1, pp. 105–132, 1982.
- [45] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009.
- [46] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
- [47] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [48] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [49] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
- [50] F. Chollet, *Deep Learning with Python*, 2nd ed. Manning Publications, 2021.
- [51] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [52] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for activation functions,” 2017, arXiv:1710.05941.
- [53] D. Scherer, A. Müller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in *Artificial Neural Networks–ICANN 2010*. Springer, 2010, pp. 92–101.
- [54] M. Lin, Q. Chen, and S. Yan, “Network in network,” 2013, arXiv:1312.4400.
- [55] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations (ICLR)*, 2015, arXiv:1409.1556.
- [56] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.
- [57] L. van der Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

- [58] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi *et al.*, “KerasTuner: A hyperparameter tuning library for Keras,” <https://github.com/keras-team/keras-tuner>, 2019.
- [59] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A novel bandit-based approach to hyperparameter optimization,” *Journal of Machine Learning Research*, vol. 18, no. 185, pp. 1–52, 2018.
- [60] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.
- [61] R. Vaser, S. Adusumalli, S. N. Leng, M. Sikic, and P. C. Ng, “Sift missense predictions for genomes,” *Nature Protocols*, vol. 11, no. 1, pp. 1–9, 2016.
- [62] D. G. Grimm *et al.*, “The evaluation of tools used to predict the impact of missense variants is hindered by two types of circularity,” *Human Mutation*, vol. 36, no. 5, pp. 513–523, 2015.
- [63] J. M. Schwarz, C. Rödelberger, M. Schuelke, and D. Seelow, “Mutationtaster: software for rapid assessment of the disease-causing potential of sequence alterations,” *Nature Methods*, vol. 7, no. 8, pp. 575–576, 2010.
- [64] H. A. Shihab, J. Gough, D. N. Cooper, P. D. Stenson, G. L. A. Barker, K. J. Edwards, I. N. M. Day, and T. R. Gaunt, “Predicting the functional, molecular, and phenotypic consequences of amino acid substitutions using hidden markov models,” *Human Mutation*, vol. 34, no. 1, pp. 57–65, 2013.
- [65] M. Kircher *et al.*, “A general framework for estimating the relative pathogenicity of human genetic variants,” *Nature Genetics*, vol. 46, no. 3, pp. 310–315, 2014.
- [66] N. M. Ioannidis *et al.*, “Revel: An ensemble method for predicting the pathogenicity of rare missense variants,” *American Journal of Human Genetics*, vol. 99, no. 4, pp. 877–885, 2016.
- [67] B.-J. Feng, “PERCH: A unified framework for disease gene prioritization,” *Human Mutation*, vol. 38, no. 3, pp. 243–251, 2017.
- [68] N. Alirezaie, K. D. Kernohan, T. Hartley, J. Majewski, and T. D. Hocking, “Clinpred: Prediction tool to identify disease-relevant nonsynonymous single-nucleotide variants,” *American Journal of Human Genetics*, vol. 103, no. 4, pp. 474–483, 2018.

- [69] B. J. Livesey and J. A. Marsh, “Updated benchmarking of variant effect predictors using deep mutational scanning,” *Molecular Systems Biology*, vol. 19, p. e11474, 2023.
- [70] L. Sundaram *et al.*, “Predicting the clinical impact of human mutation with deep neural networks,” *Nature Genetics*, vol. 50, no. 8, pp. 1161–1170, 2018.
- [71] J. Frazer *et al.*, “Disease variant prediction with deep generative models of evolutionary data,” *Nature*, vol. 599, no. 7883, pp. 91–95, 2021.
- [72] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus, “Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences,” *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, p. e2016239118, 2021.
- [73] F. Ryan-Phillips, L. Henahan, S. Ramdas, J. Palace, D. Beeson, and Y. Y. Dong, “Assessing the utility of ColabFold and AlphaMissense in determining missense variant pathogenicity for congenital myasthenic syndromes,” *Biomedicines*, vol. 12, no. 11, p. 2549, 2024.
- [74] G. M. Şilbir and B. Kurt, “Using convolutional neural networks with image-based representations of amino acid sequences for predicting the effects of genetic variants,” *The EuroBiotech Journal*, vol. 9, no. 4, pp. 247–256, 2025.
- [75] G. Ramakrishnan, C. Baakman, S. Heijl, B. Vroiling, R. van Horck, J. Hiraki, L. C. Xue, and M. A. Huynen, “Understanding structure-guided variant effect predictions using 3d convolutional neural networks,” *Frontiers in Molecular Biosciences*, vol. 10, p. 1204157, 2023.
- [76] NCBI Variation Services, “Ncbi variation services api,” <https://api.ncbi.nlm.nih.gov/variation/v0/>, 2026.
- [77] S. Raschka, “Model evaluation, model selection, and algorithm selection in machine learning,” *arXiv preprint*, 2018.
- [78] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

Appendices

Appendix A

List of Abbreviations

Genetics and Molecular Biology

ACMG American College of Medical Genetics and Genomics

AMP Association for Molecular Pathology

B Benign

BRCA1 Breast Cancer gene 1

DNA Deoxyribonucleic Acid

GRCh37/38 Genome Reference Consortium Human (builds 37 and 38)

HGVS Human Genome Variation Society

IDR Intrinsically Disordered Region

LB Likely Benign

LP Likely Pathogenic

mRNA messenger Ribonucleic Acid

P Pathogenic

PAH Phenylalanine Hydroxylase

PTM Post-Translational Modification

RNA Ribonucleic Acid

rsID Reference SNP cluster Identifier

SPDI Sequence Position Deletion Insertion

VUS Variant of Uncertain Significance

Protein Structure and Sequence Features

AF Allele Frequency

ASAquick Accessible Surface Area predictor

BLOSUM62 BLOcks SUBstitution Matrix

DFLpred Disordered Flexible Linker predictor

DRNAPred DNA- and RNA-binding residue predictor

fIDPnn intrinsic Disorder Predictor

MMseqs2 Many-against-Many sequence searching

MoRF Molecular Recognition Feature

MSA Multiple Sequence Alignment

PSIPRED PSI-blast based secondary structure PREDiction

PSSM Position-Specific Scoring Matrix

SCRIBER protein-binding residue predictor

Variant Effect Prediction Tools

CADD Combined Annotation-Dependent Depletion

dbNSFP database for
Nonsynonymous SNPs' Functional
Predictions

ESM-1b Evolutionary Scale Modeling
(protein language model)

EVE Evolutionary model of Variant
Effect

FATHMM Functional Analysis
Through Hidden Markov Models

GERP Genomic Evolutionary Rate
Profiling

LRT Likelihood Ratio Test

PolyPhen-2 Polymorphism
Phenotyping version 2

PROVEAN Protein Variation Effect
Analyzer

REVEL Rare Exome Variant
Ensemble Learner

SIFT Sorting Intolerant From Tolerant

VEST Variant Effect Scoring Tool

Databases and Resources

ESP Exome Sequencing Project

ExAC Exome Aggregation Consortium

gnomAD Genome Aggregation
Database

GoESP Grand Opportunity Exome
Sequencing Project

HGMD Human Gene Mutation
Database

NCBI National Center for
Biotechnology Information

NHLBI National Heart, Lung, and
Blood Institute

TOPMED Trans-Omics for Precision
Medicine

UniProt Universal Protein Resource

Machine Learning and Deep Learning

AUC Area Under the ROC Curve

CNN Convolutional Neural Network

DenseNet Densely Connected
Convolutional Network

FN False Negative

FP False Positive

GAP Global Average Pooling

KL Kullback–Leibler divergence

LLR Log-Likelihood Ratio

MCC Matthews Correlation
Coefficient

ML Machine Learning

MLP Multilayer Perceptron

PR-AUC Precision–Recall Area
Under the Curve

ReLU Rectified Linear Unit

ROC Receiver Operating
Characteristic

SMOTE Synthetic Minority
Over-sampling Technique

SVM Support Vector Machine

t-SNE t-Distributed Stochastic
Neighbor Embedding

TN True Negative

TP True Positive

VAE Variational Autoencoder

VGG Visual Geometry Group

Data Formats and Interfaces

API Application Programming
Interface

JSON JavaScript Object Notation

REST Representational State Transfer

TSV Tab-Separated Values

VCF Variant Call Format