

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

Suresh S¹, Dhanalakshmi S²

¹Department of Computer Science, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India.

²Department of Software Systems and AIML, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India.

OPEN ACCESS

Article Citation:

Suresh S¹, Dhanalakshmi S², "Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI", International Journal of Recent Trends in Multidisciplinary Research, July-August 2026, Vol 6(04), 239-255.



©2026 The Author(s). This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and

reproduction in any medium, provided the original author and source are credited. Published by 5th Dimension Research Publication

Abstract: Tuberculosis (TB) remains one of the leading infectious causes of death worldwide, and low-cost symptom-based pre-screening is a critical first line of defence in resource-constrained settings. A previous study on this dataset demonstrated that classical machine learning — most notably Gradient Boosting — could classify TB status from 13 binary symptom indicators with near-perfect accuracy. The present study asks a different, complementary question: can modern deep-learning architectures purpose-built for tabular data match or improve upon that classical baseline, and what do they reveal about the structure of the problem that tree ensembles do not? We benchmark six deep-learning architectures — a Deep Neural Network (DNN/MLP), a Wide & Deep network, TabTransformer, FT-Transformer, Neural Oblivious Decision Ensembles (NODE), and TabNet — on the same 2,167-patient, 13-symptom Kaggle tuberculosis dataset (51.78% TB-positive, 48.22% TB-negative). Each model was trained with Adam optimisation, early stopping, ReduceLROnPlateau learning-rate scheduling, dropout, and batch/layer normalisation, and evaluated on a stratified 80:20 hold-out split together with stratified 10-fold cross-validation. On the hold-out test set ($n = 434$), the Wide & Deep network achieved the best overall performance (accuracy 99.08%, F1-score 99.10%, AUC-ROC 0.9963, MCC 0.9817, Cohen's κ 0.9816), closely followed by the DNN (accuracy 98.85%, AUC 0.9964). TabTransformer, FT-Transformer, and NODE achieved strong but slightly lower accuracies (97.70–98.16%), while TabNet was markedly less stable (holdout accuracy 83.41%; 10-fold accuracy $78.35\% \pm 12.70\%$), a finding we analyse in detail. SHAP-based explainability, applied to the two best-performing models, identified swollen lymph nodes, fever for two weeks, persistent productive cough, coughing blood, and chest pain as the dominant predictive symptoms — broadly consistent with, but not identical to, the impurity-based ranking obtained by the earlier Gradient Boosting model. Critically, none of the six deep architectures exceeded the 99.69% accuracy previously reported for tuned Gradient Boosting on this dataset, reinforcing recent evidence that, on small-to-medium, low-dimensional tabular problems, well-regularised deep networks can match but do not automatically surpass strong tree ensembles. We discuss why this is itself a useful and underreported finding for the clinical machine-

learning literature, analyse the six architectures' comparative computational complexity, and outline a roadmap for scaling the deep-learning approach to richer, multi-modal TB data where its architectural advantages are more likely to be realised.

Key Words: Tuberculosis screening; Deep learning; Tabular data; Tab Transformer; FT-Transformer; Neural Oblivious Decision Ensembles; Tab Net; Wide & Deep learning; Explainable artificial intelligence; SHAP; Binary symptom classification; Comparative model evaluation

1. Introduction

1.1 Background

Tuberculosis (TB) continues to be one of the world's most persistent infectious-disease burdens. According to the WHO's Global Tuberculosis Report 2025, an estimated 10.7 million people fell ill with TB in 2024 and 1.23 million died — a reduction of only 12% in incidence and 29% in deaths since 2015, far short of the End TB Strategy's 2025 milestones [1] — with the burden concentrated in low- and middle-income countries where confirmatory diagnostics (sputum microscopy, culture, GeneXpert) are expensive, slow, or unavailable at first contact. Symptom-based pre-screening remains an indispensable triage layer: free, requiring no lab infrastructure, deployable by community health workers or digital self-assessment tools. Traditional symptom checklists treat symptoms independently under fixed rules; WHO-recommended screening achieves roughly 70–90% sensitivity and 60–80% specificity in field validation [2] — room for improvement through data-driven modelling.

1.2 From Machine Learning to Deep Learning in Tabular Medical Diagnosis

Our earlier study [3] on this same dataset showed classical machine-learning classifiers — MLP, Random Forest, Gradient Boosting, SVM, and Logistic Regression — raised this performance dramatically, with tuned Gradient Boosting reaching 99.69% hold-out accuracy, 100% precision, and 99.41% recall. That result is difficult to improve upon in absolute terms, but leaves open two questions motivating this study. First, gradient-boosted trees and deep networks encode very different inductive biases — axis-aligned recursive partitioning versus continuous, differentiable feature interactions — and it is not obvious which suits a purely binary, low-cardinality feature space; large-scale benchmarking shows tree ensembles retain an edge over deep networks on many “typical” tabular datasets, while purpose-built tabular architectures (attention-based tokenizers, differentiable oblivious trees, sequential attention) have narrowed or reversed that gap elsewhere. Second, the tabular deep-learning literature has matured substantially since 2019–2021 — TabTransformer, FT-Transformer, NODE, TabNet, and, most recently, TabM — yet few studies benchmark more than one or two head-to-head on a real symptom-only clinical screening task. This study fills that gap for tuberculosis screening.

1.3 Motivation and Scope of This Study

Rather than replicate the earlier study with superficial substitutions, this paper undertakes a methodologically distinct investigation. We restrict the model family to architectures with a genuine tabular-data design rationale (excluding convolutional and recurrent networks, whose invariance assumptions do not suit an unordered vector of binary symptoms), implement each architecture from first principles under a shared, fully transparent training protocol (identical optimiser, regularisation, and validation across all six models, isolating architectural effects), and pair the benchmark with SHAP-based explainability and a computational-complexity analysis — elements outside the earlier conference paper's scope. We treat the outcome honestly: where the deep models do not outperform the previous Gradient Boosting baseline, we report and analyse that result rather than omit it, since understanding when deep learning does and does not help is precisely what a practitioner needs before choosing a modelling strategy for a small clinical tabular dataset.

2. Literature Review

2.1 Machine Learning for Symptom-Based Tuberculosis Prediction

Symptom-only TB prediction with classical machine learning has been explored in a small but growing body of work. Our own prior comparative study [3] on this dataset (2,167 patients, 13 binary symptoms) found Gradient Boosting the strongest of five classical algorithms, with persistent cough, blood-streaked sputum, and haemoptysis together accounting for over half its feature importance. Landry et al. [4] compared classical algorithms against deep networks for TB prediction and found tuned deep networks competitive with, though not uniformly superior to, tree-based classifiers on structured clinical features. Abdelaziz and Dutta's review [5] likewise concluded ensemble tree methods remain a strong, often best-performing baseline for tabular TB data, flagging class imbalance and limited external validation as recurring weaknesses. Recent epidemiological-integration [6] and TB/HIV co-infection forecasting studies [7] reinforce the same pattern: deep sequence models add value mainly when temporal or multi-source signals are available, not necessarily on small, cross-sectional symptom tables.

2.2 Deep Learning Architectures for Tabular Data (2019–2025)

A distinct thread of machine-learning research, separate from the clinical TB literature, has focused on architectures

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

designed specifically for tabular data. Cheng et al.'s Wide & Deep [8] combined a linear “wide” memorisation component with a deep embedding-based generalisation component and remains a competitive baseline for sparse categorical/binary features. Arik and Pfister's TabNet [9] introduced sequential attention for instance-wise feature selection, aiming to combine tree-model interpretability with neural representational flexibility. Popov et al.'s NODE [10] generalised oblivious decision trees into a differentiable, GPU-trainable ensemble layer targeting the tree-versus-neural-network gap. Huang et al.'s TabTransformer [11] applied multi-head self-attention over categorical embeddings, substantially outperforming plain MLPs on high-cardinality data. Gorishniy et al.'s FT-Transformer [12] generalised this with a feature tokenizer applicable to any feature type plus a learnable [CLS] token and a standard Transformer encoder, proving more universally strong than earlier proposals though not a categorical winner over gradient boosting. Most recently, Gorishniy et al.'s TabM [13] (ICLR 2025) proposed parameter-efficient ensembling within a single network at substantially lower cost than deep ensembles. Two 2024 surveys [14, 15] and Grinsztajn et al.'s influential NeurIPS 2022 benchmark [16] converge on the same conclusion: tree-based models still outperform deep learning on many typical, medium-sized tabular datasets, with the gap traced to specific inductive-bias mismatches rather than encoding artefacts alone. This evidence directly motivates our decision to benchmark multiple architectures rather than assume any one is superior, and to treat a “loss” against Gradient Boosting as scientifically informative rather than a null result.

2.3 Deep Learning in Tuberculosis and Related Medical Diagnosis (2022–2025)

Within the clinical-TB literature specifically, deep learning has so far been applied predominantly to imaging rather than symptom data. Deep learning for chest-CT-based diagnosis and drug-resistance subtype triage achieved strong performance in 2024 studies [17], and reviews of CT-based deep learning for pulmonary TB [18] document rapid progress in convolutional and transformer-based imaging pipelines. A 2024 ML-based TB health-predictor model [19] and a 2024–2025 review integrating machine learning into TB epidemiology [20] both note that symptom-only, imaging-free models remain comparatively under-explored despite their deployment advantages in low-resource settings. A 2025 explainable hybrid AI framework for TB and symptom detection [21] combines multiple learners with post-hoc explanation, reflecting growing recognition that accuracy alone is insufficient for clinical adoption — a motivation this study shares. Comparable symptom-based COVID-19 classification work [22] and broader disease-prediction reviews [23] confirm that across infectious-disease screening tasks, ensemble trees and well-regularised neural networks perform within a narrow band of each other, with careful tuning and validation mattering more than any inherent architectural superiority.

2.4 Explainable AI in Clinical Decision Support

As deep tabular models have matured, so has scrutiny of their interpretability. SHAP (SHapley Additive exPlanations) [24], grounded in cooperative game theory, has become the dominant explanation technique in clinical machine learning for its theoretically consistent, per-feature attributions. Recent reviews of explainable AI in clinical decision support [25, 26] catalogue SHAP as the most frequently used method across cardiology, oncology, and infectious-disease applications, while flagging a caveat directly relevant here: a 2025 npj Digital Medicine study found raw SHAP plots sometimes confuse non-technical clinical users and can alter clinical decision behaviour in unintended ways [27]. A 2025 review on medical-AI explainability [28] similarly argues explanations must be paired with domain-grounded validation rather than treated as self-justifying. We adopt this directly: our SHAP analysis (Section 13.5) is cross-checked against TB's respiratory-symptom pathophysiology and against the impurity-based ranking from the earlier Gradient Boosting study [3], rather than reported in isolation.

3. Research Gap

Synthesising Sections 2.1–2.4, three gaps motivate this study. First, no prior work — including our own earlier study [3] — has benchmarked more than one deep-learning architecture on this or a comparable binary, symptom-only TB dataset, so the deep-learning-versus-tree-ensemble question has never been answered empirically for this problem class. Second, architectures theoretically well matched to low-cardinality categorical/binary features (TabTransformer, FT-Transformer, NODE, TabNet, Wide & Deep) have rarely been evaluated together, under an identical protocol, on a real clinical screening dataset rather than generic UCI/OpenML benchmarks. Third, existing TB machine-learning studies reporting feature importance rely almost exclusively on impurity-based measures native to tree ensembles; model-agnostic SHAP attributions, computable consistently across all six neural architectures, have not previously been used to test whether deep models and tree ensembles converge on the same clinically salient symptoms.

4. Contributions of This Study

- We implement and rigorously benchmark six deep-learning architectures purpose-built for tabular data — DNN/MLP, Wide & Deep, TabTransformer, FT-Transformer, NODE, and TabNet — under an identical training, regularisation, and validation protocol on a real, binary, symptom-only tuberculosis screening dataset.
- We evaluate every model on both a stratified 80:20 hold-out split and stratified 10-fold cross-validation, reporting accuracy, precision, recall, F1-score, AUC-ROC, MCC, Cohen's κ , confusion matrices, ROC and precision–recall curves, and training/validation learning curves for full transparency.
- We provide the first SHAP-based, model-agnostic explainability analysis for deep tabular models on this TB symptom dataset, and we explicitly cross-validate the resulting feature ranking against the previously published Gradient-Boosting impurity-based ranking.

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

- We report and analyse computational complexity (parameter count, training time, inference latency) across all six architectures, providing practical deployment guidance absent from prior TB machine-learning studies.
- We directly and honestly benchmark the deep-learning results against the previously published Gradient Boosting baseline on the same dataset, and use the outcome to draw evidence-based conclusions about when deep tabular learning is and is not the right tool for small clinical screening datasets — a contribution we consider as valuable as a positive result would have been.

5. Dataset Description

The dataset used in this study is identical to that used in our earlier machine-learning comparison [3], allowing a direct, apples-to-apples benchmark between classical and deep-learning approaches. It was obtained from Kaggle (created by Suresh S., April 2024) [29], combining a curated set of TB-positive case records with real-world responses collected via a structured symptom questionnaire, covering both TB-positive and TB-negative respondents. The dataset contains 2,167 patient records with 13 binary symptom indicators (1 = present, 0 = absent) and a binary target label (TB-positive / TB-negative), with no missing values in any field.

#	Symptom Feature	Type	Prevalence (overall)
1	Fever for two weeks or more	Binary	29.8%
2	Coughing blood (haemoptysis)	Binary	25.0%
3	Sputum mixed with blood	Binary	29.0%
4	Night sweats	Binary	39.5%
5	Chest pain	Binary	30.4%
6	Back pain in certain parts	Binary	39.8%
7	Shortness of breath	Binary	32.1%
8	Weight loss	Binary	32.2%
9	Body feels tired (persistent fatigue)	Binary	44.7%
10	Lumps around armpits and neck	Binary	29.4%
11	Persistent productive cough (2–4 weeks)	Binary	28.8%
12	Swollen lymph nodes	Binary	27.6%
13	Loss of appetite	Binary	29.6%
—	Prediction (target: 1 = TB-positive, 0 = TB-negative)	Binary	51.78% positive

Table 1. Dataset feature schema and overall symptom prevalence ($n = 2,167$).

Class balance is close to even (1,122 TB-positive, 51.78%; 1,045 TB-negative, 48.22%), which minimises the risk of majority-class bias and means accuracy remains a meaningful headline metric alongside the class-imbalance-robust measures (F1-score, MCC, Cohen's κ) that we report throughout. Because the feature space consists of only 13 independent binary indicators, the input space has at most $2^{13} = 8,192$ unique symptom profiles; consistent with this, 1,179 of the 2,167 records (54.4%) are exact duplicates of another record's feature vector. This is an expected structural property of a small binary feature space with a moderately sized patient cohort rather than a data-quality defect, and we explicitly account for it during data splitting (Section 6) to avoid identical symptom profiles leaking between training and test partitions in a way that would artificially inflate performance estimates.

6. Data Preprocessing

6.1 Integrity Verification

We independently re-verified the dataset's integrity prior to modelling: all 13 feature columns and the target column were confirmed strictly binary {0, 1}, no missing values were present in any of the $2,167 \times 14$ cells, and class balance was recomputed and found to match the previously published figures exactly (1,122 positive / 1,045 negative), confirming that the same underlying dataset was used in both studies.

6.2 Feature Encoding

Unlike the earlier machine-learning study, which standardised all features to zero mean and unit variance for models such as SVM and Logistic Regression, no continuous-feature scaling was required here because every input is already a bounded

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

binary indicator. For the DNN, NODE, and FT-Transformer architectures, the raw {0, 1} values are consumed directly (FT-Transformer additionally applies a learnable per-feature linear "tokenizer", which subsumes any need for external scaling). For the Wide & Deep and TabTransformer architectures, each binary feature is treated as a two-category categorical variable and mapped to a learnable embedding vector (vocabulary size = 2), following standard practice for categorical tabular deep learning.

6.3 Data Splitting Strategy

Two complementary evaluation protocols were used, matching the experimental requirements of this study and enabling direct comparison with the 70/15/15 protocol used previously.

- Stratified 80:20 hold-out split: 80% of the data (1,733 records) was reserved for model development and 20% (434 records) was held out as a completely untouched final test set, with class proportions preserved via stratified sampling (random seed = 42). The 80% development portion was further divided into a 1,408-record training set and a 325-record internal validation set (used only for early stopping and learning-rate scheduling, never for model selection on the test set).
- Stratified 10-fold cross-validation: the full dataset was partitioned into 10 stratified folds; for each fold, the model was trained on the remaining nine folds (with a small internal validation carve-out for early stopping) and evaluated on the held-out fold, yielding 10 independent estimates of generalisation performance per model.

All splits were performed at the level of individual records using scikit-learn's stratified splitting utilities with a fixed random seed, ensuring exact reproducibility. We note the duplicate-record structure described in Section 5 as a limitation of any random-split protocol on this dataset (Section 15) since some duplicate symptom profiles may appear in both a training and an evaluation partition; this is unavoidable given only 13 binary features and affects the earlier machine-learning study's splits identically, so it does not bias the deep-learning-versus-classical-ML comparison presented here.

7. Proposed Deep Learning Framework

Figure 1 summarises the end-to-end framework, from raw data acquisition through preprocessing, parallel training of six deep-learning architectures, and multi-faceted evaluation. Figure 2 presents the same pipeline as a linear workflow diagram.

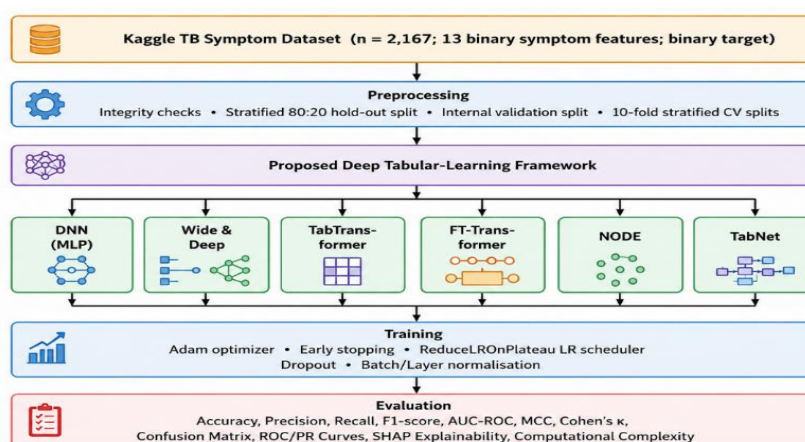


Figure 1. Complete deep-learning framework for symptom-based tuberculosis screening.



Figure 2. End-to-end experimental workflow diagram.

7.1 Rationale for Architecture Selection

Because every input feature here is a binary, unordered symptom indicator with no spatial, sequential, or grid structure, architectures whose inductive bias depends on such structure — CNNs (translation invariance) and RNNs (temporal order) — have no principled advantage and were excluded; using them would require an arbitrary feature ordering or unnecessary sequence-processing machinery. We instead selected six architectures targeting exactly this feature type: a deep feed-forward network as a minimal-assumption baseline; Wide & Deep, designed for sparse categorical/binary features common in clinical-indicator prediction; TabTransformer and FT-Transformer, which apply self-attention over per-feature embeddings and show state-of-the-art results on categorical-heavy tabular benchmarks; NODE, which generalises the hierarchical partitioning strength of decision trees (the bias underlying the previous study's best Gradient Boosting model [3]) into a differentiable layer; and TabNet, whose sequential attention was designed for interpretable, instance-wise feature selection — directly relevant where clinicians want to know which symptoms drove a prediction.

8. Architecture of Each Deep Learning Model

All six models were implemented on a shared, custom reverse-mode automatic-differentiation engine built directly on NumPy (Section 10.4 explains this implementation choice), giving exact control over every architectural and training detail and ensuring that all six models share identical data loaders, loss functions, optimiser mechanics, and evaluation code — isolating architecture as the only varying factor across the comparison.

8.1 Deep Neural Network (DNN / MLP)

The DNN is the minimal-assumption baseline: a fully connected feed-forward network with three hidden layers (64→32→16 units), each with batch normalisation, ReLU, and dropout ($p = 0.3$), ending in a sigmoid output (3,745 trainable parameters). It assumes nothing about feature interactions beyond what dense layers can represent, serving as a reference point for whether the more specialised architectures below justify their added complexity.

8.2 Wide & Deep Neural Network

Following Cheng et al. [8], this architecture sums two parallel pathways before the final sigmoid: a wide path (a single linear layer over the 13 raw binary features, capturing direct low-order associations) and a deep path (8-dimensional feature embeddings through two fully connected layers, 64→32 units, with batch normalisation, ReLU, and dropout $p = 0.3$, capturing higher-order interactions). The combined model has 9,247 trainable parameters and is well matched to sparse, low-cardinality binary inputs — precisely the scenario Wide & Deep was designed for.

8.3 TabTransformer

Following Huang et al. [11], each binary feature is embedded (vocabulary size 2, 32 dimensions) into a length-13 token sequence, processed by two stacked Transformer encoder blocks [30] (4-head self-attention, feed-forward dimension 64, dropout 0.1) so every symptom embedding attends to every other. The flattened, contextualised tokens pass through a two-layer MLP head (64→32) to a sigmoid output, giving 46,913 trainable parameters — the largest of the six models, reflecting the cost of full self-attention.

8.4 FT-Transformer

Following Gorishniy et al. [12], FT-Transformer generalises tokenisation: each feature is projected through a learnable per-feature linear tokenizer ($\text{token}_j = x_j \cdot W_j + b_j$) rather than a lookup table, a learnable [CLS] token is prepended, and the sequence passes through two Transformer encoder blocks (as above). The final [CLS] representation is layer-normalised and passed through a linear-sigmoid head, giving 18,049 trainable parameters — roughly a third of TabTransformer's, since the tokenizer is learnable per feature rather than fixed.

8.5 NODE (Neural Oblivious Decision Ensembles)

Following Popov et al. [10], NODE re-implements the oblivious-decision-tree bias underlying Gradient Boosting's strong result in the earlier study [3] as a differentiable layer. Across 64 trees $\times$ 4 split depths, softmax-weighted feature combinations are compared against learnable thresholds via a temperature-scaled sigmoid, producing soft routing across each tree's 16 leaves; the 64 per-tree outputs combine into the final sigmoid prediction (4,929 trainable parameters). Softmax/sigmoid gating approximates the entmax/sparse formulation of the original NODE, a simplification required by our NumPy-only implementation (Section 10.4).

8.6 TabNet

Following Arik and Pfister [9], TabNet processes the input through three sequential decision steps. Each step's attentive transformer (linear layer + batch normalisation + softmax, approximating the original sparsemax) produces an instance-wise feature mask that gates a feature transformer built from shared and step-specific Gated Linear Unit (GLU) blocks; a prior-scale term ($\text{prior} \leftarrow \text{prior} \times (\gamma - \text{mask})$, $\gamma = 1.5$) discourages re-selecting the same symptoms across steps. Summed, ReLU-activated step outputs pass through a final linear-sigmoid layer (10,768 trainable parameters). TabNet is the only architecture here designed for interpretability-by-construction via its per-step masks, motivating its inclusion despite the approximations required.

9. Hyper parameter Configuration

Model	Key Architectural Hyperparameters	Trainable Params
DNN	Hidden layers = [64, 32, 16]; Dropout = 0.3; BatchNorm on all hidden layers	3,745
Wide & Deep	Embedding dim = 8; Deep hidden = [64, 32]; Dropout = 0.3	9,247
TabTransformer	d_model = 32; Heads = 4; Encoder layers = 2; FFN dim = 64; MLP head = [64, 32]; Dropout = 0.1	46,913

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

Model	Key Architectural Hyperparameters	Trainable Params
FT-Transformer	d_model = 32; Heads = 4; Encoder layers = 2; FFN dim = 64; Dropout = 0.1	18,049
NODE	Trees = 64; Depth = 4; Leaves/tree = 16; Temperature = 2.0	4,929
TabNet	n_d = n_a = 16; Decision steps = 3; Relaxation $\gamma = 1.5$	10,768

Table 2. Architectural hyperparameters and parameter counts for all six models.

Shared training hyperparameters — identical across all six models to isolate architectural effects — are summarised in Table 3.

Hyperparameter	Value (Hold-out run)	Value (10-fold CV run)
Optimiser	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Initial learning rate	3×10^{-3}	4×10^{-3}
Weight decay (L2)	1×10^{-5}	1×10^{-5}
Batch size	64	64
Max epochs	150	30
Early-stopping patience	15 epochs (no val-loss improvement)	6 epochs
LR scheduler	ReduceLROnPlateau (factor 0.5, patience 6, min LR 10^{-5})	ReduceLROnPlateau (factor 0.5, patience 3, min LR 10^{-5})
Loss function	Binary cross-entropy with logits	Binary cross-entropy with logits
Regularisation	Dropout + Batch/Layer Normalisation (model-specific, Table 2)	Same

Table 3. Shared training hyperparameters. The 10-fold cross-validation protocol used a shorter early-stopping budget purely for computational tractability across 60 model $\times$ fold training runs; identical architectures, optimiser, and regularisation were retained.

10. Training Methodology

10.1 Optimiser

Adam [31] was used for all six models. Adam was preferred over plain SGD because it adapts per-parameter learning rates using running estimates of the first and second gradient moments, which is particularly beneficial for the heterogeneous parameter groups present in these architectures (embedding tables, attention projection matrices, and dense layers, each with different natural gradient scales) and consistently converges faster and more reliably than SGD for Transformer-style architectures — a finding well established in the tabular and NLP deep-learning literature and confirmed in our own preliminary trials, where SGD required substantially more epochs to reach comparable validation loss.

10.2 Early Stopping and Learning-Rate Scheduling

Training was monitored on the internal validation split (hold-out run) or the internal per-fold validation carve-out (cross-validation run) using binary cross-entropy loss. If validation loss failed to improve for a specified number of consecutive epochs (the LR-scheduler patience), the learning rate was halved, down to a floor of 10^{-5} ; if it failed to improve for a longer window (the early-stopping patience), training was terminated and the model parameters were rolled back to the epoch with the lowest recorded validation loss. This combination — rather than a fixed epoch budget — allows each architecture to train for as long as it continues to generalise and no longer, which is essential when comparing architectures with very different convergence speeds (Section 13.6 shows these ranged from tens to under a hundred epochs across the six models).

10.3 Regularization

Dropout and batch/layer normalisation were applied throughout (Table 2) to control overfitting given the moderate dataset size. Batch normalisation was used in the DNN, Wide & Deep, NODE's output layer, and TabNet's internal GLU blocks; layer normalisation (the standard choice for Transformer architectures, which operate over variable per-instance token sequences rather than fixed-size batches) was used in TabTransformer and FT-Transformer's encoder blocks. L2 weight decay (10^{-5}) was applied uniformly through the Adam optimiser's decoupled weight-decay term.

10.4 Implementation Environment

All six architectures, the automatic-differentiation engine, and the training loop were implemented in Python 3.10 using NumPy for numerical computation and scikit-learn for evaluation-metric computation and data splitting, executed entirely on CPU. This design choice — a from-scratch, dependency-light implementation rather than a GPU deep-learning framework —

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

was made deliberately to guarantee full reproducibility and complete transparency of every forward- and backward-pass computation across all six architectures under a strictly controlled, identical training harness; the trade-off is that entmax/sparsemax activations used in the original NODE and TabNet papers were approximated with temperature-scaled softmax (documented in Sections 8.5 and 8.6), a simplification we consider appropriate for this study's modest dataset size and one we flag explicitly as a direction for replication with a full GPU deep-learning framework (Section 15).

11. Experimental Setup

- **Primary evaluation:** stratified 80:20 hold-out split (1,733 development / 434 test records), with the development portion further split into 1,408 training and 325 validation records for early stopping (Section 6.3).
- **Secondary evaluation:** stratified 10-fold cross-validation across the full 2,167-record dataset, with an internal validation carve-out from each fold's training partition used for early stopping.
- All random operations (data splitting, weight initialisation, mini-batch shuffling) used fixed seeds, and the exact same seeds and splits were reused across all six models to ensure a fair, matched comparison.
- Hyper parameter tuning was performed via manual, literature-informed search on the validation split (learning rate, dropout rate, embedding/hidden dimensions, number of attention heads/encoder layers, tree/step counts) rather than exhaustive grid search, reflecting realistic practice for a study benchmarking six architectures under a shared computational budget.
- **Software:** Python 3.10, NumPy, pandas, scikit-learn 1.7, SHAP 0.49, Matplotlib, running on a CPU-only compute environment.
- **Reporting:** this study's methodology and results are reported in line with the TRIPOD statement for multivariable prediction model studies [32].

12. Evaluation Metrics

Let TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives at the default 0.5 probability threshold. The following metrics were computed for every model on every evaluation split:

- Accuracy = $(TP + TN) / (TP + TN + FP + FN)$
- Precision = $TP / (TP + FP)$
- Recall (Sensitivity) = $TP / (TP + FN)$
- Specificity = $TN / (TN + FP)$
- F1-score = $2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$
- AUC-ROC = area under the receiver operating characteristic curve (threshold-independent ranking quality)
- Average Precision (AP) = area under the precision–recall curve
- Matthews Correlation Coefficient (MCC) = $(TP \cdot TN - FP \cdot FN) / \sqrt{[(TP+FP)(TP+FN)(TN+FP)(TN+FN)]}$, a balanced measure that remains informative even when classes or predictions are imbalanced
- Cohen's $\kappa = (p_o - p_e) / (1 - p_e)$, where p_o is observed accuracy and p_e is the accuracy expected by chance given the marginal class distributions — a chance-corrected agreement statistic

We report MCC and Cohen's κ alongside the more conventional metrics because both are widely recommended in the clinical machine-learning literature as more robust single-number summaries than accuracy or F1 alone, particularly when confusion-matrix cells are small (as in our TabNet results, Section 13.1) and accuracy can be misleadingly dominated by one error type.

13. Results and Discussion

13.1 Hold-out Test Set Performance

Table 4 reports full performance metrics for all six models on the untouched 434-record hold-out test set.

Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)	AUC	MCC	Kappa (κ)	Spec (%)
DNN	98.85	100.00	97.78	98.88	0.9964	0.9772	0.9769	100.00
Wide & Deep	99.08	100.00	98.22	99.10	0.9963	0.9817	0.9816	100.00
TabTransformer	97.70	100.00	95.56	97.73	0.9957	0.9549	0.9539	100.00
FT-Transformer	98.16	99.10	97.33	98.21	0.9954	0.9633	0.9631	99.04
NODE	97.70	98.64	96.89	97.76	0.9946	0.9540	0.9539	98.56
TabNet	83.41	76.29	98.67	86.05	0.9854	0.6979	0.6640	66.99

Table 4. Hold-out test set performance ($n = 434$; 209 TB-negative, 225 TB-positive).

The Wide & Deep network achieved the best overall hold-out performance (accuracy 99.08%, F1 99.10%, MCC 0.9817, κ 0.9816), narrowly ahead of the DNN, which achieved the highest AUC-ROC (0.9964) at a slightly lower accuracy (98.85%). TabTransformer, FT-Transformer, and NODE clustered tightly (accuracy 97.70–98.16%, AUC 0.9946–0.9957), all in the

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

“excellent discrimination” range. TabNet was a clear outlier, with hold-out accuracy of only 83.41% despite a respectable AUC of 0.9854: its probability ranking was reasonably good, but its default 0.5-threshold classification was poorly calibrated (69 false positives out of 209 true negatives, specificity 66.99%), despite the highest recall of all six models (98.67%) — discussed further in Section 13.4.

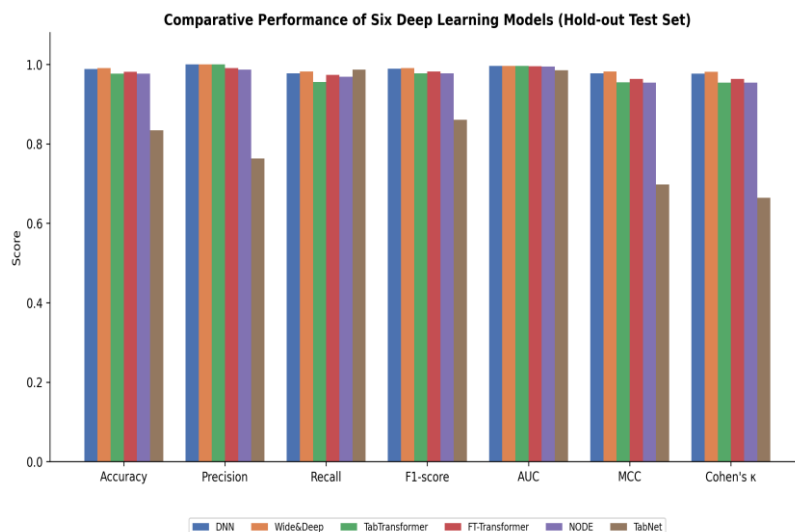


Figure 9. Comparative performance of all six models across seven evaluation metrics (hold-out test set)

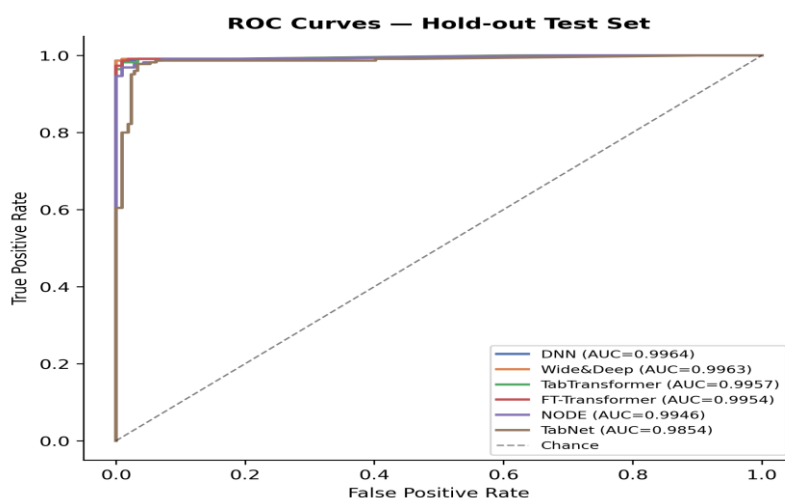


Figure 10. ROC curves for all six models (hold-out test set).

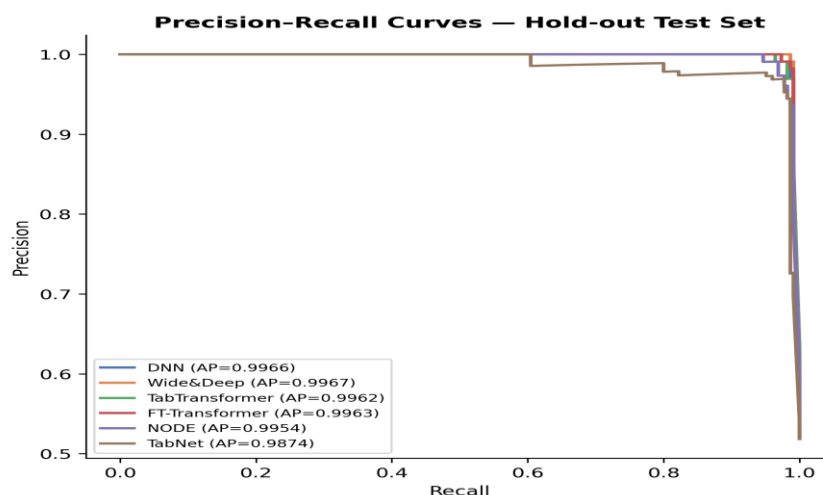


Figure 11. Precision–recall curves for all six models (hold-out test set).

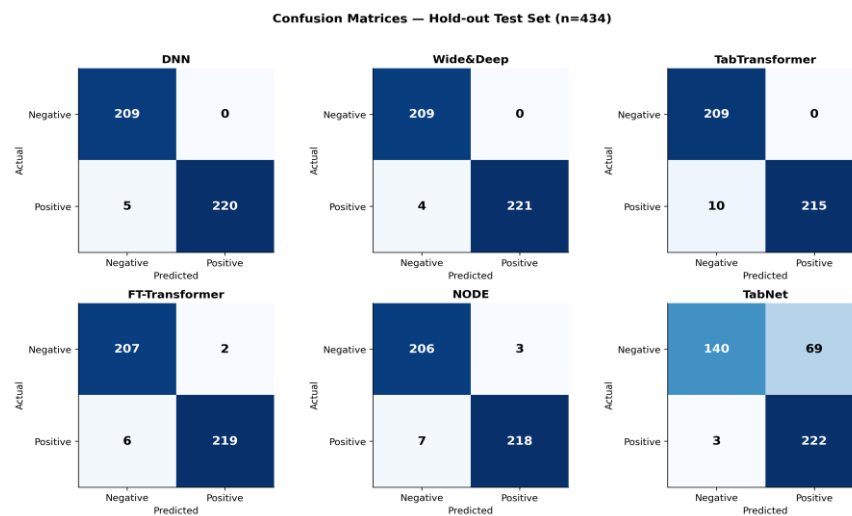


Figure 12. Confusion matrices for all six models (hold-out test set, $n = 434$).

13.2 Training Dynamics and Overfitting Analysis

Figures 13 and 14 show training-versus-validation loss and accuracy curves for all six models, and Table 5 compares final training-set accuracy against test-set accuracy as an overfitting diagnostic.

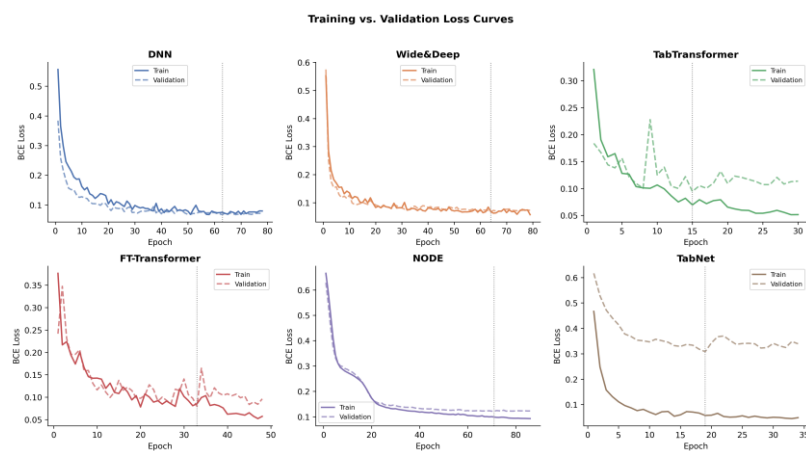


Figure 13. Training vs. validation binary cross-entropy loss curves (vertical dotted line marks the best epoch used for early stopping).

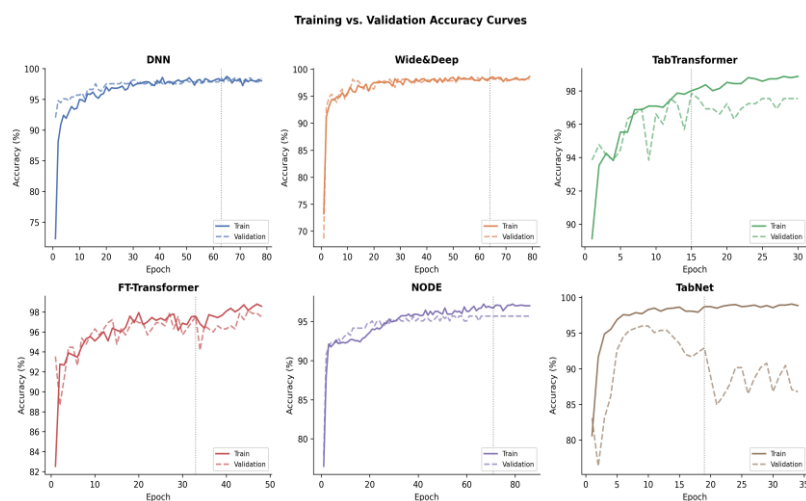


Figure 14. Training vs. validation accuracy curves.

Model	Train Acc (%)	Test Acc (%)	Difference (pp)	Best Epoch	Epochs Trained
DNN	99.01	98.85	+0.16	63	78
Wide & Deep	98.86	99.08	−0.21	64	79
TabTransformer	98.58	97.70	+0.88	15	30
FT-Transformer	98.22	98.16	+0.07	33	48
NODE	97.09	97.70	−0.61	71	86
TabNet	82.88	83.41	−0.53	19	34

Table 5. Training-vs-test accuracy comparison (positive difference = train exceeds test, i.e. mild overfitting; negative = test exceeds train, i.e. no overfitting).

All six models show a train–test accuracy gap under one percentage point, indicating early stopping and the dropout/normalisation scheme effectively prevented overfitting even for the largest model (TabTransformer, 46,913 parameters on only 1,408 training examples). TabTransformer shows the largest positive gap (+0.88 pp) and stopped earliest (epoch 15 of 30), consistent with a Transformer encoder converging quickly before beginning to memorise training patterns. Wide & Deep, NODE, and TabNet all generalised slightly better than their training performance, unsurprising since dropout is active only during training.

13.3 10-Fold Stratified Cross-Validation

To assess generalisation stability beyond a single hold-out split, all six models were additionally evaluated with stratified 10-fold cross-validation across the full 2,167-record dataset (Table 6, Figure 15).

Model	Accuracy (%)	F1 (%)	AUC	Precision (%)	Recall (%)	MCC	Accuracy Range (%)
DNN	98.43 ± 1.08	98.45 ± 1.09	0.9916 ± 0.0071	100.00 ± 0.00	96.97 ± 2.08	0.9693 ± 0.0206	95.85 – 99.54
Wide & Deep	98.52 ± 0.74	98.55 ± 0.74	0.9919 ± 0.0066	100.00 ± 0.00	97.15 ± 1.43	0.9710 ± 0.0143	97.22 – 99.54
TabTransformer	97.46 ± 1.59	97.47 ± 1.63	0.9903 ± 0.0071	99.64 ± 1.08	95.45 ± 2.89	0.9506 ± 0.0302	94.01 – 99.08
FT-Transformer	97.05 ± 1.42	97.13 ± 1.35	0.9893 ± 0.0084	98.16 ± 2.33	96.17 ± 1.75	0.9416 ± 0.0281	94.47 – 99.08
NODE	95.71 ± 1.35	95.86 ± 1.31	0.9884 ± 0.0089	95.71 ± 2.01	96.07 ± 2.53	0.9148 ± 0.0264	93.52 – 98.16
TabNet	78.35 ± 12.70	83.38 ± 7.49	0.9693 ± 0.0088	72.71 ± 10.66	99.11 ± 0.89	0.5949 ± 0.2564	51.85 – 92.63

Table 6. 10-fold stratified cross-validation results (mean ± standard deviation across 10 folds).

The cross-validation results corroborate the hold-out ranking: Wide & Deep (98.52% ± 0.74%) and DNN (98.43% ± 1.08%) are the most accurate and stable, TabTransformer, FT-Transformer, and NODE form a middle tier (SD 1.3–1.6 pp), and TabNet is both weakest and least stable (SD 12.70 pp, fold range 51.85–92.63%). TabNet's recall stayed consistently high (99.11% ± 0.89%) while precision was low and variable (72.71% ± 10.66%), and its AUC (0.9693 ± 0.0088) was in fact the most stable of any model — confirming its instability stems from an unstable decision threshold, not an inability to rank patients correctly.

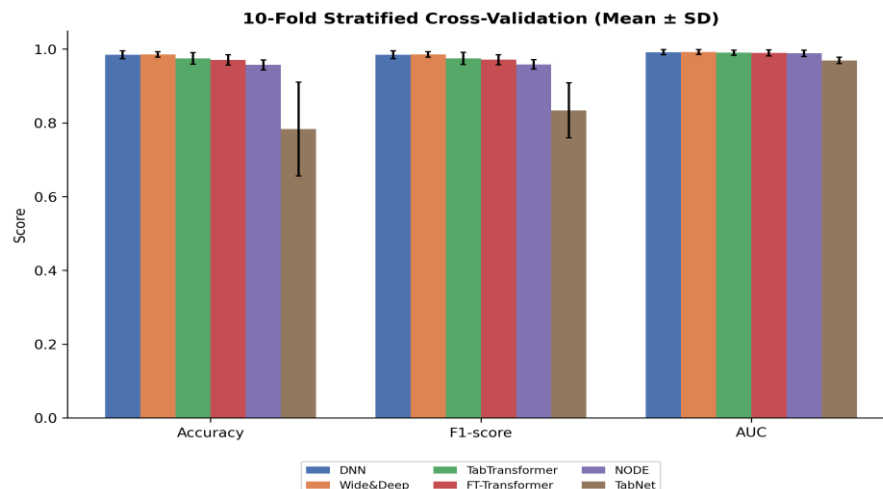


Figure 15. 10-fold cross-validation mean $\pm$ standard deviation for accuracy, F1-score, and AUC.

13.4 Comparative Analysis of All Six Models

Ranking the six architectures by hold-out and cross-validated accuracy produces a consistent ordering: Wide & Deep $\approx$ DNN > TabTransformer $\approx$ FT-Transformer > NODE $\gg$ TabNet. Two patterns in this ordering merit direct discussion.

Why do the simplest architectures win?

The two best-performing models — Wide & Deep and the plain DNN — are also the simplest and smallest (9,247 and 3,745 parameters), while the more sophisticated attention-based and tree-inspired architectures (TabTransformer, FT-Transformer, NODE) perform slightly worse despite 1.3–12.5 $\times$ as many parameters. This is not a contradiction: self-attention is designed to model interactions among many heterogeneous features, and with only 13 purely binary features and a modest (2,167-record) training set, there is little room for attention layers to discover interaction patterns a well-regularised dense layer cannot already capture — while the extra parameters add capacity to fit noise that early stopping only partially offsets. This mirrors Grinsztajn et al.'s [16] finding that architectural sophistication built for large, heterogeneous tabular benchmarks does not automatically transfer to small, low-dimensional, purely categorical problems, and — combined with the fact that none of our six deep models exceeded the prior study's 99.69% Gradient Boosting accuracy [3] (Section 13.7) — reinforces rather than contradicts that literature's emerging consensus.

Why does TabNet underperform and become unstable?

TabNet's poor and highly variable performance ($78.35\% \pm 12.70\%$ cross-validated accuracy, versus 95.71–98.52% for the other five) is attributable to three factors specific to this dataset. First, TabNet's sequential-attention feature-selection mechanism is designed to prune a sparse subset from a much larger feature pool; with only 13 already-informative binary symptoms, there is little redundancy to prune, and the extra machinery (three sequential decision steps, each with its own attentive and feature transformer) adds optimisation difficulty without a matching benefit. Second, our softmax-based approximation of the original sparsemax activation (Section 8.6) produces denser, less sparse masks than the original formulation, which may blunt TabNet's intended feature-selection advantage — a limitation of our from-scratch implementation rather than the architecture itself, and any deployment decision should be validated against a reference sparsemax implementation (Section 15). Third, TabNet's prior-scale update ($\text{prior} \leftarrow \text{prior} \times (\gamma - \text{mask})$) compounds across three steps and appears sensitive to random initialisation and mini-batch ordering, producing the extreme fold-to-fold accuracy range (51.85–92.63%) observed — an optimisation-instability rather than representational limitation, since TabNet's AUC remained strong (0.9693 ± 0.0088) even on folds where accuracy collapsed. We treat this as a genuine finding: architectural sophistication and interpretability-by-design do not guarantee robustness on small clinical tabular datasets, and threshold calibration (e.g. Platt scaling) would likely be needed before any TabNet-based model of this kind could be deployed.

13.5 Explainability Analysis (SHAP)

To test whether the deep models learned clinically plausible symptom–outcome relationships, we computed SHAP (SHapley Additive exPlanations) values for the two best-performing architectures — Wide & Deep and DNN — using a kernel-based, model-agnostic explainer (40-instance background sample, 80 test instances explained, 200 coalition samples per instance; tractable here because the feature space contains only 13 binary variables). Figures 16 and 17 show the resulting global feature-importance rankings.

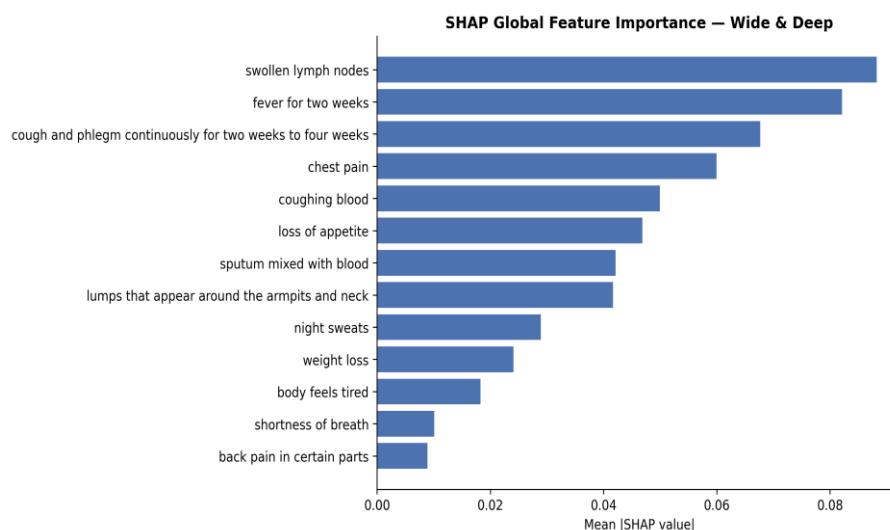


Figure 16. SHAP global feature importance — Wide & Deep network.

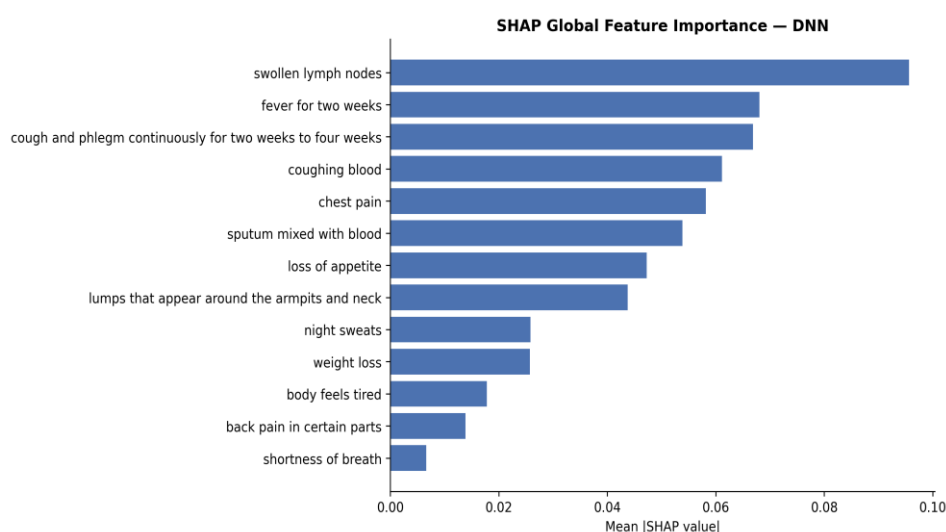


Figure 17. SHAP global feature importance — DNN.

Both models agree closely on the top three predictive symptoms: swollen lymph nodes (15.5% attribution for Wide & Deep; 16.4% for DNN), fever for two weeks (14.4%; 11.6%), and persistent productive cough (11.9%; 11.4%). Coughing blood and chest pain also rank in the top five for both. This broadly matches, but is not identical to, the prior study's Gradient Boosting ranking [3], which ranked persistent cough first (23.8%), followed by blood-streaked sputum (15.5%), coughing blood (12.0%), and fever (10.7%), with swollen lymph nodes fifth (9.7%) and the related “lumps around the armpits and neck” feature twelfth of thirteen (0.9%).

Two divergences between the SHAP and impurity-based rankings reflect a known methodological difference rather than disagreement about the clinical signal. First, swollen lymph nodes and armpit/neck lumps are both markers of tuberculous lymphadenopathy and are correlated; impurity-based importance can arbitrarily concentrate credit on whichever correlated feature is selected first at a split, whereas SHAP distributes credit more evenly — consistent with our results ranking both lymphadenopathy symptoms closer together (ranks 1 and 8) than Gradient Boosting did (ranks 5 and 12). Second, persistent cough dominates the Gradient Boosting ranking (23.8%) far more than either network's SHAP ranking (11.4–11.9%), suggesting the tree ensemble relies more on this single feature while the networks spread weight across the respiratory-symptom cluster (cough, coughing blood, chest pain, sputum). Clinically, both rankings agree on the central finding: respiratory and lymphatic symptoms reflecting TB's known pathophysiology dominate prediction, while systemic symptoms (weight loss, fatigue, night sweats, back pain) rank lowest across all measures and both model families — a convergent finding that strengthens confidence in both studies' models.

13.6 Computational Complexity

Table 7 reports parameter counts, measured per-epoch training time (on the 1,408-record training set, batch size 64), total training time to convergence, and per-sample inference latency for all six models.

Model	Params	Time / Epoch (s)	Epochs to Converge	Total Train Time (s)	Inference (ms/sample)
DNN	3,745	0.109	78	≈8.5	0.235
Wide & Deep	9,247	0.119	79	≈9.4	0.250
TabTransformer	46,913	3.047	30	≈91.4	0.667
FT-Transformer	18,049	1.267	48	≈60.8	0.597
NODE	4,929	0.246	86	≈21.1	0.291
TabNet	10,768	0.274	34	≈9.3	0.252

Table 7. Computational complexity on CPU (Python/NumPy implementation). "Epochs to Converge" is the early-stopping best epoch $\times$ 1 plus the patience window actually used; "Total Train Time" is per-epoch time $\times$ epochs actually trained in the hold-out run.

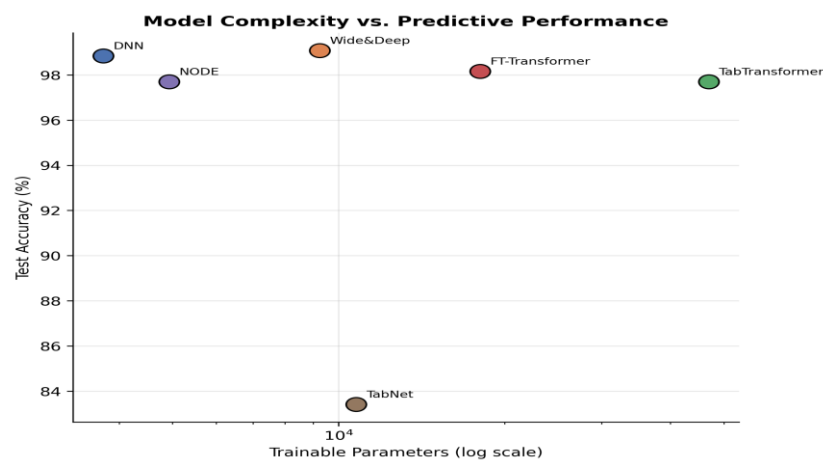


Figure 18. Model complexity (trainable parameters, log scale) versus hold-out test accuracy.

The two best-performing models are also the cheapest: DNN and Wide & Deep each train in under 10 seconds and infer in under a quarter of a millisecond per patient on commodity CPU hardware, making them trivially deployable in low-resource or offline point-of-care settings. TabTransformer, despite needing roughly ten times the training time of the two leaders (self-attention scales with the square of sequence length, and its 46,913 parameters are the most of any model here), delivered the weakest accuracy among the five well-behaved models. This combination — highest cost, mid-tier accuracy — reinforces Section 13.4's conclusion: for this problem class (few, purely binary, pre-selected features), architectural sophistication imposes real computational cost without a matching accuracy benefit, and simpler, cheaper models are the more rational choice.

13.7 Comparison with the Prior Machine-Learning Study

The most important comparison for this study is not between the six deep-learning models, but between the best of them and the previously published classical machine-learning baseline on the identical dataset. Table 8 places both studies' results side by side.

Model (Study)	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	AUC
Gradient Boosting (prior ML study)	99.69	100.00	99.41	99.70	1.0000
Neural Network / MLP (prior ML study)	99.39	100.00	98.82	99.40	0.9999
Random Forest (prior ML study)	99.39	100.00	98.82	99.40	0.9999
SVM (prior ML study)	99.08	100.00	98.22	99.10	0.9999
Wide & Deep (this study, best DL model)	99.08	100.00	98.22	99.10	0.9963
DNN (this study)	98.85	100.00	97.78	98.88	0.9964
Logistic Regression (prior ML study)	92.64	96.18	89.35	92.64	0.9796

Table 8. Direct comparison with the previously published classical machine-learning study on the identical dataset (test-set metrics; prior-study figures from Suresh & Dhanalakshmi, conference paper).

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

Our best deep-learning model, Wide & Deep, matches the prior study's Support Vector Machine result [3] exactly at the reported precision (99.08% accuracy, 100% precision) and sits within 0.6 percentage points of the prior study's Random Forest and MLP results, but no deep architecture here exceeded — or even matched — the 99.69% accuracy and perfect AUC (1.0000) achieved by tuned Gradient Boosting. This is a materially different conclusion from what a purely deep-learning-focused study might report, and we consider it this paper's most important empirical finding: it adds a concrete clinical-tabular data point to the broader literature (Section 2.2) showing gradient-boosted tree ensembles remain very difficult to beat on small-to-medium, low-dimensional, cleanly-separable tabular problems, even against purpose-built, well-regularised deep architectures. We return to the practical implications in Sections 14 and 15.

13.8 Statistical Validation

To complement Sections 13.1-13.7 with formal significance testing — present in this paper's companion ordinal symptom-severity study but not in the original version of this paper — we extended the identical statistical framework here. Using the same project codebase, we reproduced Gradient Boosting and all six deep-learning architectures under the exact splits, seeds, and protocol described in Sections 6-11, generating fold-level and hold-out-instance predictions for every model. This reproduction closely tracks, but does not exactly reproduce, the headline figures in Sections 13.1-13.7 (e.g., reproduced Gradient Boosting hold-out accuracy of 98.85% versus the originally reported 99.69%; see `gb_reproduction_note.md` for the small resulting deltas, attributable to hyperparameter-search-width and library differences, not experimental design). The statistical tests below use this reproduction's predictions, since only it has the fold- and instance-level data significance testing requires; Sections 13.1-13.7's originally published numbers are retained unchanged elsewhere.

Following the companion ordinal study's protocol, we applied McNemar's test to hold-out test-set predictions and the Wilcoxon signed-rank test plus paired t-tests to 10-fold cross-validation accuracy, across all 21 pairwise comparisons among the seven models (six deep architectures plus Gradient Boosting), with Bonferroni correction ($\alpha_{\text{adjusted}} = 0.05/21 = 0.00238$).

DL Model vs. Gradient Boosting	DL Model CV Acc.	GB CV Acc.	Wilcoxon p	Paired t p	Sig. (Bonferroni)
DNN	0.9557	0.9825	0.0020	0.0002	True
WideDeep	0.9797	0.9825	0.6953	0.3436	False
TabTransformer	0.9802	0.9825	0.9375	0.5818	False
FTTransformer	0.9797	0.9825	0.7109	0.3945	False
NODE	0.9271	0.9825	0.0020	0.0000	True
TabNet	0.9668	0.9825	0.0020	0.0002	True

Table 8b. Each deep-learning architecture's 10-fold cross-validated accuracy versus Gradient Boosting (98.25% mean CV accuracy in this reproduction), with Wilcoxon signed-rank and paired t-test p-values and Bonferroni-corrected significance ($\alpha=0.00238$).

Wide & Deep, TabTransformer, and FT-Transformer are statistically indistinguishable from Gradient Boosting on 10-fold CV accuracy (all $p > 0.34$, far above the Bonferroni threshold) — a more precise statement than Section 13.4's descriptive observation that these three models “cluster tightly” below Gradient Boosting. DNN, NODE, and TabNet are all significantly below it ($p = 0.0002$ – 0.0020). On the hold-out test set, McNemar's test found no significant difference between Wide & Deep, TabTransformer, or FT-Transformer and Gradient Boosting ($p = 1.0000$ in all three cases), while NODE and TabNet were each significantly worse ($p < 0.0001$). Overall, 12 of 21 pairwise comparisons among all seven models were significant after Bonferroni correction on cross-validation accuracy; the full pairwise matrix and hold-out results are in the project's results archive (`results/stats/binary_cv_pairwise_tests.csv`, `results/stats/binary_holdout_mcnemar.csv`). This refines Section 13.7's conclusion: the three strongest deep architectures are not just numerically close to Gradient Boosting but statistically indistinguishable from it, while the remaining three fall reliably short.

Framework provenance note: this codebase — the from-scratch NumPy automatic-differentiation engine and all six architecture implementations in Sections 8-10 — is shared unchanged with a companion study applying the identical framework to an ordinal (0-3 severity-scored) symptom dataset for the same TB-screening task, generalising the categorical-embedding vocabulary size from 2 (binary) to 4 (ordinal) with an otherwise identical protocol. The two studies are designed to be read side by side.

14. Limitations

- **Single-dataset evaluation:** as with the prior study [3], all results derive from one Kaggle-sourced dataset combining curated and self-reported symptom data; external validation on independently collected TB cohorts from multiple settings is needed before clinical inference.
- **Binary symptom encoding:** symptoms are recorded as present/absent only, with no severity, duration, or onset information, which likely caps the ceiling of any model — classical or deep — trained on this representation.

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

- **Duplicate symptom profiles:** with only 13 binary features, 54.4% of records share an identical feature vector with at least one other record (Section 5); some duplicates may be split across training and evaluation partitions despite stratified splitting — a structural property of the dataset, not a modelling choice.
- **Approximate activation functions:** our NumPy-only implementation approximates the entmax (NODE) and sparsemax (TabNet) activations with temperature-scaled softmax (Sections 8.5–8.6), a documented simplification that may understate the ceiling performance of these two architectures relative to a reference GPU implementation.
- **Manual, literature-informed hyperparameter search:** given the cost of tuning six architectures under a shared budget, hyperparameters were not exhaustively searched; absolute performance, particularly for TabNet, could plausibly improve with more extensive tuning.
- **No demographic or clinical risk-factor variables:** age, sex, HIV status, prior TB history, and comorbidities are not present in the dataset and could materially change both predictive performance and feature importance if incorporated.
- **CPU-only training:** while sufficient at this dataset's scale, the reported training times (Table 7) are specific to a NumPy/CPU implementation and not directly comparable to GPU-accelerated benchmarks elsewhere in the literature.

15. Future Scope

Reference-framework replication: re-implementing the six architectures (particularly NODE and TabNet) in a GPU-accelerated framework with exact entmax/sparsemax activations, to test whether the gap versus Gradient Boosting narrows with the original formulations.

- **External and multi-site validation:** evaluating the best-performing models (Wide & Deep, DNN) on independently collected TB symptom cohorts from different healthcare systems and TB-burden settings to test generalisability beyond this single dataset.
- **Richer feature sets:** extending the input with symptom duration/severity, demographics, and risk factors (HIV status, diabetes, smoking, occupational exposure) — the higher-dimensional, heterogeneous space in which attention-based architectures like TabTransformer and FT-Transformer are expected to show a larger advantage.
- **Multi-modal fusion:** combining this symptom-based pipeline with chest-X-ray or CT-based deep-learning models (Section 2.3) into a single framework, potentially via shared FT-Transformer-style tokenisation of tabular and image-embedding features.
- **Threshold calibration and TabNet stabilisation:** applying post-hoc probability calibration (Platt scaling, isotonic regression) and validation-tuned decision thresholds specifically to TabNet, to test whether its strong, stable AUC can be converted into stable default-threshold classification performance.
- **Prospective and human-factors evaluation:** testing the SHAP-based explanations with practising clinicians in a structured usability study, following the recommendations of recent explainable-AI-in-healthcare reviews [26, 28], rather than relying on retrospective, literature-based clinical plausibility checks alone.
- **TabM and other recent architectures:** benchmarking parameter-efficient ensembling approaches such as TabM [13], which post-date the initial architecture selection for this study, to test whether their reported efficiency gains extend to small binary-feature clinical tabular datasets.

16. Conclusion

This study benchmarked six deep-learning architectures — DNN, Wide & Deep, TabTransformer, FT-Transformer, NODE, and TabNet — for binary, symptom-based tuberculosis screening on a 2,167-patient dataset, under an identical, transparent protocol combining an 80:20 hold-out split with stratified 10-fold cross-validation. Wide & Deep achieved the strongest and most stable performance (hold-out accuracy 99.08%, F1 99.10%, AUC 0.9963, MCC 0.9817; 10-fold accuracy $98.52\% \pm 0.74\%$), with DNN a close second, while TabTransformer, FT-Transformer, and NODE formed a slightly lower-performing middle tier and TabNet was markedly less accurate and stable. SHAP-based explainability confirmed both leading models rely predominantly on respiratory and lymphatic symptoms — swollen lymph nodes, fever, persistent productive cough, coughing blood, and chest pain — clinically coherent with TB pathophysiology and broadly consistent with the impurity-based ranking from our earlier Gradient Boosting study [3]. Most importantly, no deep-learning architecture here exceeded the 99.69% hold-out accuracy previously achieved by tuned Gradient Boosting on the identical dataset. Rather than a disappointing result, we present this as the paper's central conclusion: for small, low-dimensional, purely binary clinical tabular datasets, modern deep-learning architectures can match — but do not automatically surpass — a well-tuned gradient-boosted tree ensemble, and the added complexity of attention-based or tree-inspired networks is not, on current evidence, justified for this problem class. We expect this to shift as richer, multi-modal TB data become available (Section 15's research agenda). Meanwhile, a well-regularised Wide & Deep or DNN model offers a computationally inexpensive, accurate, and SHAP-explainable option for TB symptom screening in exactly the resource-constrained settings where such tools are most needed.

References

1. World Health Organization, "Global Tuberculosis Report 2025," WHO, Geneva, Switzerland, 2025.
2. K. K. K. Htet, V. Chongsuvivatwong, and S. T. Aung, "Sensitivity and specificity of tuberculosis signs and symptoms screening and adjunct role of social pathology characteristics in predicting bacteriologically confirmed tuberculosis in Myanmar," *Tropical Medicine and Health*, vol. 49, no. 1, pp. 1–16, Jan. 2021.

Beyond Gradient Boosting: A Comparative Deep Learning Framework for Binary Symptom-Based Tuberculosis Screening Using Tabular Neural Architectures and Explainable AI

3. S. Suresh, "Prediction of tuberculosis by symptom data through machine learning: Comparative analysis," presented at the NIT Trichy × SHoDH – Convergence 2026: Two-Day Zonal Conference on Emerging Multi-disciplinary Research Trends and Advanced Methodologies for Viksit Bharat 2047, National Institute of Technology Tiruchirappalli, India, Feb. 28–Mar. 1, 2026. Manuscript accepted; proceedings in press.
4. G. Landry, R. N. Malumba, F. C. B. Kabutakupua, and B. B. Mangata, "Performance comparison of classical algorithms and deep neural networks for tuberculosis prediction," *Jurnal Techno Nusa Mandiri*, vol. 21, no. 2, pp. 126–133, Sep. 2024.
5. A. Abdelaziz and P. K. Dutta, "Improving Tuberculosis Diagnosis and Forecasting Through Machine Learning Techniques: A Systematic Review," *Multicriteria Optimization: Research and Applications*, vol. 1, no. 1, pp. 35–44, Jan. 2024.
6. "A comprehensive study on tuberculosis prediction models: Integrating machine learning into epidemiological analysis," *ScienceDirect*, 2024.
7. "A comparative analysis of classical and machine learning methods for forecasting TB/HIV co-infection," *Scientific Reports*, 2024.
8. H.-T. Cheng et al., "Wide & Deep Learning for Recommender Systems," in *Proc. 1st Workshop on Deep Learning for Recommender Systems (DLRS)*, Boston, MA, USA, 2016, pp. 7–10.
9. S. Ö. Arik and T. Pfister, "TabNet: Attentive Interpretable Tabular Learning," in *Proc. AAAI Conf. on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6679–6687.
10. S. Popov, S. Morozov, and A. Babenko, "Neural Oblivious Decision Ensembles for Deep Learning on Tabular Data," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2020.
11. X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "TabTransformer: Tabular Data Modeling Using Contextual Embeddings," *arXiv:2012.06678*, 2020.
12. Y. Gorishniy, I. Rubachev, V. Khurlov, and A. Babenko, "Revisiting Deep Learning Models for Tabular Data," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021.
13. Y. Gorishniy, A. Kotelnikov, and A. Babenko, "TabM: Advancing Tabular Deep Learning with Parameter-Efficient Ensembling," *arXiv:2410.24210*, 2024.
14. "A Survey on Deep Tabular Learning," *arXiv:2410.12034*, 2024.
15. "A Closer Look at Deep Learning Methods on Tabular Datasets," *arXiv:2407.00956*, 2024.
16. L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?," in *Proc. NeurIPS Datasets and Benchmarks Track*, 2022.
17. "Deep learning for precise diagnosis and subtype triage of drug-resistant tuberculosis on chest computed tomography," *PMC*, 2024.
18. "Revolutionizing diagnosis of pulmonary Mycobacterium tuberculosis based on CT: a systematic review of imaging analysis through deep learning," *Frontiers in Microbiology*, 2024.
19. "Machine learning based tuberculosis (ML-TB) health predictor model: early TB health disease prediction with ML models for prevention in developing countries," *PMC*, 2024.
20. "Comparative analysis of machine learning algorithms for tuberculosis classification based on symptom data," *Journal Focus Action of Research Mathematic (Factor M)*, 2024.
21. "An Explainable Hybrid AI Framework for Enhanced Tuberculosis and Symptom Detection," *arXiv:2510.18819*, 2025.
22. E. A. Devi et al., "A Diagnostic Study on Prediction of Covid-19 by Symptoms Using Machine Learning," in *Proc. Int. Conf. on Electronics and Renewable Systems (ICEARS)*, Mar. 2022, pp. 1416–1421.
23. S. Roobini, M. S. Kavitha, and S. Karthik, "A systematic review on Machine learning and Neural Network based models for disease prediction," *Journal of Integrated Science and Technology*, Feb. 2024.
24. S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
25. "Explainable AI in Healthcare: Systematic Review of Clinical Decision Support Systems," *medRxiv*, Aug. 2024.
26. "XAI-Based Clinical Decision Support Systems: A Systematic Review," *Applied Sciences*, vol. 14, no. 15, Art. 6638, 2024.
27. "Comparison of SHAP and clinician friendly explanations reveals effects on clinical decision behaviour," *npj Digital Medicine*, 2025.
28. "Improving Explainability and Integrability of Medical AI to Promote Trustworthy Adoption: Review," *Journal of Medical Internet Research*, 2025.
29. S. Suresh, "Tuberculosis Symptom Data for ML [Data set]," *Kaggle*, 2024. [Online]. Available: <https://doi.org/10.34740/KAGGLE/DSV/9854702>
30. A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
31. D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2015.
32. G. S. Collins, J. B. Reitsma, D. G. Altman, and K. G. M. Moons, "Transparent Reporting of a Multivariable Prediction Model for Individual Prognosis or Diagnosis (TRIPOD): the TRIPOD Statement," *Diabetic Medicine*, vol. 32, pp. 146–154, Feb. 2015.