Mathematical Optimization of Loan Approval Neural Networks via First-Order Taylor Pruning and Affine Quantization

Jakub Dras[1]

[1] *Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, POLAND*
Email: jd309975@student.polsl.pl

Abstract

The deployment of Deep Learning models in financial risk assessment is often hindered by computational constraints and the requirement for interpretability. This paper presents a rigorously derived optimization pipeline for a Multi-Layer Perceptron (MLP) designed for loan approval prediction. I propose a dual-stage compression strategy rooted in explainability theory and numerical approximation. Firstly, I apply SHAP (SHapley Additive exPlanations) values based on cooperative game theory to reduce the input space dimensionality. Secondly, I derive a structural pruning criterion based on the first-order Taylor expansion of the loss function, which approximates the change in error induced by removing specific neurons. This method allows for the identification and elimination of redundant latent structures without retraining. Finally, I implement Post-Training Dynamic Quantization, mapping continuous 32-bit floating-point weights to discrete 8-bit integers using an affine transformation scheme. Experimental results confirm the efficacy of this approach: the model size was reduced by a factor of 3.35 (from 63.13 KB to 18.83 KB) with a negligible accuracy drop (from 0.912 to 0.897), demonstrating that theoretical pruning criteria can effectively guide practical model compression.

Keywords

Neural Networks, Taylor Expansion, Structural Pruning, Affine Quantization, Explainable AI (XAI), SHAP, Financial Modeling

1. Introduction

In a different domain, the transition from linear statistical models (e.g., Logistic Regression) [1, 2] to non-linear approximations via Artificial Neural Networks (ANNs) [3] has significantly improved predictive accuracy. Such an example is quantitative finance [4]. However, this performance gain comes at the cost of model complexity. Deep neural networks are notoriously over-parameterized, often containing far more degrees of freedom than necessary to solve the target task. It leads to excessive system resources consumption (RAM, CPU/GPU) and limits performance, which are critical bottlenecks in modern fintech applications requiring rapid credit decisions and requires processing a huge number of transactions.

Furthermore, the "black box" nature of ANNs causes a regulatory challenge. Financial institutions are mandated to provide transparent justifications for credit decisions. Consequently, achieving high predictive performance often comes at the expense of model interpretability and their performance.

To address these challenges, this study introduces a mathematically grounded optimization framework. I leverage Explainable AI (XAI) to inform the pruning process, transforming interpretation tools into a powerful mechanism for architectural compression. By utilizing the Taylor series expansion of the cost function, I derive a "saliency" metric for hidden neurons, allowing me to perform structural pruning that is theoretically justified rather than heuristic [5]. I combine this with integer quantization to minimize the bit-width of the remaining parameters [6]. The contribution of this paper is a comprehensive, end-to-end pipeline that transforms a dense, redundant neural network into a compact, efficient, and interpretable estimator.

IVUS 2026: Information Society and University Studies 2026, May 12, Kaunas, Lithuania

jd309975@student.polsl.pl (J. Dras)

Copyright © 2026 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-06

CEUR-WS.org/Vol-4317/paper28.pdf

2. Methodology

My methodology relies on a sequential reduction of complexity, starting from the input features and progressing to the latent space (neurons) and finally the parameter space (bits).

2.1. Neural Network Formulation

I define the Multi-Layer Perceptron as a composition of affine transformations and non-linear activations. Let L be the number of layers. The output of the l-th layer, $h^{(l)}$, is given by:

$$h^{(l)} = f\left(W^{(l)}h^{(l-1)} + b^{(l)}\right) \tag{1}$$

where $W^{(l)}$ is the weight matrix, $b^{(l)}$ is the bias vector, and $f(\cdot)$ is the activation function. I employ the LeakyReLU activation [7] to mitigate the vanishing gradient problem:

$$\text{LeakyReLU}(x) = \max(0, x) + 0.1\min(0, x) \tag{2}$$

The final output $\hat{y}$ represents the probability of loan approval, modelled by the Sigmoid function:

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

2.2. Feature Pruning via Cooperative Game Theory (SHAP)

Before optimizing the network structure, I address input redundancy using SHAP (SHapley Additive exPlanations) [8]. This method treats feature attribution as a cooperative game where input features are players, and the model prediction is the payout. The contribution ϕ_i of feature i is calculated as the Shapley value:

$$\phi_i(f, x) = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!}\left[f_S(x_S \cup \{i\}) - f_S(x_S)\right] \tag{4}$$

where F is the set of all features, S is a subset of features excluding i, and f_S is the model trained only on subset features. Due to the computational complexity of calculating exact Shapley values $\left(2^{|F|}\right)$, I utilize DeepSHAP, which approximates these values by combining DeepLIFT with Shapley values. Features with mean absolute SHAP values close to zero ($\mathbb{E}[|\phi_i|] \approx 0$) are considered non-informative and are pruned from the input layer.

2.3. Structural Pruning via Taylor Expansion

For the hidden layers, I want to estimate the change in loss $\Delta\mathcal{L}$ caused by setting a specific parameter or neuron to zero. Consider the Taylor series expansion of $\mathcal{L}$ around the current weights W, inspired by optimal brain damage theory [9]:

$$\mathcal{L}(W + \Delta W) = \mathcal{L}(W) + \nabla\mathcal{L}(W)^T\Delta W + \frac{1}{2}\Delta W^T H\mathcal{L}(W)\Delta W + \mathcal{O}(\|\Delta W\|^3) \tag{5}$$

where $\nabla\mathcal{L}$ is the gradient and H is the Hessian matrix. The change in loss is:

$$\Delta\mathcal{L} = \mathcal{L}(W + \Delta W) - \mathcal{L}(W) \approx \nabla\mathcal{L}(W)^T\Delta W \tag{6}$$

I ignore the second-order term to reduce computational cost (First-Order Oracle) [5]. For a structural pruning of a neuron i in layer l, the "removal" is equivalent to setting its activation output

$a_i^{(l)}$ to zero. Thus, ΔW is effectively represented by the perturbation of the activation. The importance score S_i is defined as the absolute change in loss:

$$S_i^{(l)} = |\Delta \mathcal{L}| \approx \left| \frac{\partial \mathcal{L}}{\partial a_i^{(l)}} \cdot a_i^{(l)} \right| \tag{7}$$

This score captures both the magnitude of the neuron's output (a_i) and the sensitivity of the network to that output $\left(\frac{\partial \mathcal{L}}{\partial a_i}\right)$. Neurons with small S_i contribute minimally to the loss gradient and are safe candidates for pruning.

2.4. Affine Quantization (Int8)

After structural pruning, I apply dynamic quantization [6]. This process maps floating-point values $r \in \mathbb{R}$ (weights) to integers $q \in \mathbb{Z}$ (8-bit, range [-128, 127]). I use an affine mapping defined by a scale factor S and a zero-point Z :

$$q = \text{round}\left(\frac{r}{S} + Z\right) \tag{8}$$

where round $(\cdot)$ denotes rounding to the nearest integer. The de-quantization (approximation of the original value) is given by:

$$\tilde{r} = S(q - Z) \tag{9}$$

The scale S and zero-point Z are determined by the range of the tensor elements $[r_{\min}, r_{\max}]$:

$$S = \frac{r_{\max} - r_{\min}}{q_{\max} - q_{\min}}, Z = \text{round}\left(q_{\min} - \frac{r_{\min}}{S}\right) \tag{10}$$

This transformation reduces the memory footprint of weights by a factor of 4 , significantly compressing the model.

Table 1
Dataset Schema and Preprocessing Strategy

Feature	Type	Description	Preprocessing
annual_income	Numerical	Applicant's yearly income	$\ln(x+1)$,Scaled
loan_amount	Numerical	Requested loan principal	$\ln(x+1)$, Scaled
cibil_score	Numerical	Credit score indicator	Scaled
term	Categorical	Loan duration (Short/Long)	One-Hot Encoding
int_rate	Numerical	Interest rate	Scaled
installment	Numerical	Monthly payment	Scaled
dti	Numerical	Debt-to-Income ratio	Scaled
loan_intent	Categorical	Purpose (Edu, Med, Venture, etc.)	One-Hot Encoding
...	...	...	...

3. Experiments

3.1. Dataset Description

The experiments utilized the Loan Approval dataset (2025). The feature set comprises both numerical (continuous) and categorical variables. Table 1 details the variable types and transformations.

3.2. Baseline Architecture Specification

I define the exact baseline topology of the unpruned network, visualized in Figure ??. The model is a fully connected MLP designed to process the input vector of size N_{in}. The architecture is organized into sequential blocks to improve training stability and prevent overfitting.

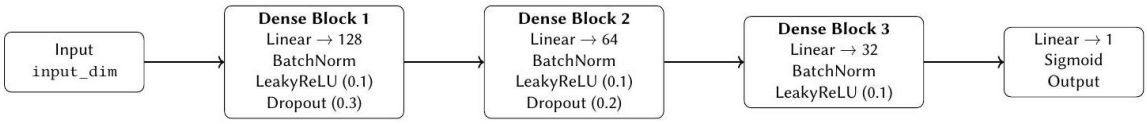


Figure 1: Baseline architecture specification

The architecture incorporates specific components to ensure convergence and generalization:

- Batch Normalization [10]: Applied after each linear layer, this technique normalizes the inputs of each layer to have a mean of zero and variance of one. It significantly stabilizes the learning process by reducing internal covariate shift, allowing for higher learning rates.
- Dropout [11]: Included in the first two dense blocks with probabilities $p = 0.3$ and $p = 0.2$ respectively. During training, it randomly zeroes out a fraction of neurons, preventing the network from relying too heavily on any single feature reducing the probability of overfitting.

It is important to note now, that the structural pruning mechanism described in Section 3.5 targets only the hidden linear layers. The final classification layer remains intact to preserve the logical mapping to the binary output space.

3.3. Training Configuration and Optimization

The model optimization was performed using PyTorch Lightning [12]. The training process is governed by the following mathematical formulations:

Objective Function I minimize the Binary Cross-Entropy (BCE) loss, which measures the dissimilarity between the true label distribution y and predicted probabilities $\hat{y}$:

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{11}$$

Optimizer (AdamW) I employ AdamW (Adam with Decoupled Weight Decay) [13], which corrects the L2 regularization implementation in standard Adam. The parameter update rule at step t for weights θ is:

$$g_t = \nabla_\theta \mathcal{L}_t(\theta_{t-1}) \tag{12}$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{13}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{14}$$

$$\theta_t = \theta_{t-1} - \eta \left(\frac{m_t}{\sqrt{v_t} + \epsilon} + \lambda \theta_{t-1} \right) \tag{15}$$

where $\eta = 10^{-3}$ is the learning rate, λ is the weight decay coefficient, and β_1, β_2 are moment decay rates. A ReduceLROnPlateau scheduler is applied to dynamically anneal η when the validation loss plateaus.

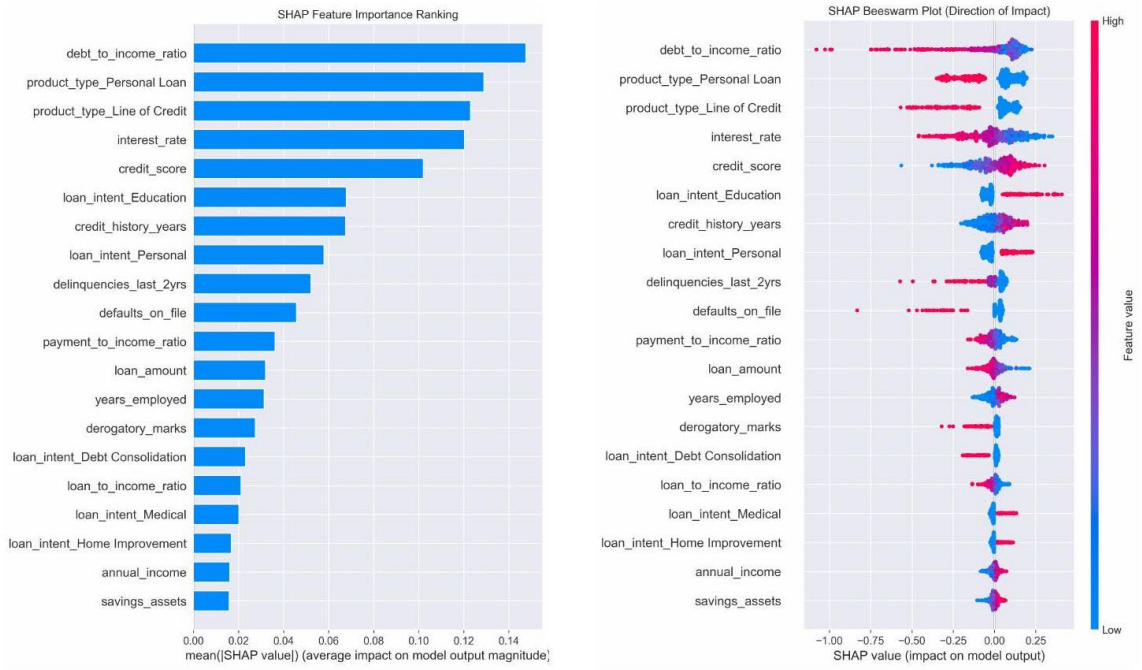
3.4. Phase 1: Feature Pruning via SHAP

Before architectural compression, I analyzed the global feature importance. Figure 2 presents the SHAP analysis. Features with negligible impact on the model output, such as savings assets,Piotr Pilszczek (pp310219) were identified and pruned, reducing the input dimension from 24 to 18.

3.5. Phase 2: Structural Pruning

I applied the Taylor-based pruning to the hidden layers. The importance of neurons was evaluated using the First-Order Oracle criterion derived in Section 2.3. To visualize the impact of pruning, I compare the architecture of the baseline model against the physically compressed model.

Table 2 details the reduction in the number of neurons for each fully connected layer. The baseline MLP followed a standard decreasing pyramid structure ($128 \rightarrow 64 \rightarrow 32$). The pruning



(a) SHAP Feature Importance Ranking (b) SHAP Beeswarm Plot (Direction of Impact)

Figure 2: SHAP Analysis. Features at the bottom of the ranking were pruned due to low contribution.

Table 2
Architectural Comparison: Baseline vs. Pruned Network Structure

Layer Type	Original Neurons	Pruned Neurons	Reduction Ratio
Input Layer	18	18	0%
Hidden Layer 1	128	90	29.7%
Hidden Layer 2	64	45	29.7%
Hidden Layer 3	32	23	28.1%
Output Layer	1	1	0%
Total Params	~14,000	~7,200	~48.5%

process removed approximately 30% of the least salient neurons from each hidden layer, resulting in a significantly sparser topology ($90 \rightarrow 45 \rightarrow 23$).

This physical removal of neurons-as opposed to merely zeroing out weights (masking)-is crucial for achieving actual reduction of system resources usage. Masked weights still require memory and computation (multiplication by zero), whereas physically removing rows and columns from the weight matrices reduces the FLOPs count quadratically.

3.6. Phase 3: Final Quantization and Performance Trade-off

The final stage involved dynamic quantization to Int8. The cumulative results of the optimization pipeline are presented in Table 3 and Figure 3.

Table 3
Quantitative Results of the Optimization Pipeline

Model Stage	Size (KB)	Accuracy	F1 Score	Reduction
Baseline (FP32)	63.13	0.912	0.920	-
Pruned (FP32)	35.94	0.897	0.906	$1.75\times$
Quantized (Int8)	**18.83**	0.897	0.906	$\mathbf{3.35}\times$

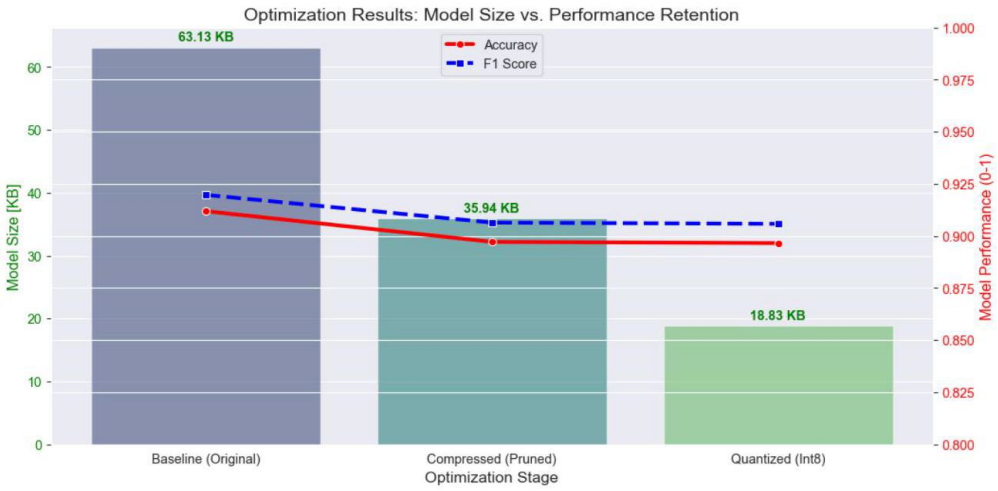


Figure 3: Optimization Trade-off: Model Size vs. Predictive Quality. The final model is significantly smaller with maintained accuracy.

The analysis reveals a highly favorable trade-off. The pruning phase incurred a minor accuracy drop (1.5%), which is expected when removing 30% of the network's capacity. However, the subsequent quantization phase reduced the model size by nearly 50% (relative to the pruned model) with zero additional loss in accuracy or F1 Score. This suggests that the pruned architecture was robust enough to withstand the precision reduction from 32-bit to 8-bit.

4. Discussion

4.1. Theoretical Complexity and Speedup Analysis

Beyond the storage reduction visualized in the experiments, it is crucial to quantify the theoretical computational gain. The processing cost of a fully connected layer is dominated by matrix-vector multiplication. For a layer with N_{in} input neurons and N_{out} output neurons, the number of Floating Point Operations (FLOPs) is given by:

$$\text{FLOPs} \approx 2 \cdot N_{\text{in}} \cdot N_{\text{out}} \tag{16}$$

(accounting for one multiplication and one addition per weight). Let us consider the second hidden layer, which was reduced from $N_{\text{in}} = 128, N_{\text{out}} = 64$ in the baseline to $N'_{\text{in}} = 90, N'_{\text{out}} = 45$ in the pruned model. The theoretical reduction ratio R_{ops} for this specific layer is:

$$R_{ops} = 1 - \frac{2 \cdot 90 \cdot 45}{2 \cdot 128 \cdot 64} = 1 - \frac{8100}{16384} \approx 50.5\% \tag{17}$$

This calculation demonstrates that structural pruning provides a quadratic reduction in computational complexity when neurons are removed from consecutive layers.

The combined effect of R_{ops} (from pruning) and bit-width reduction (from quantization) implies that the theoretical latency τ of the optimized model satisfies:

$$\tau_{\text{final}} \ll \tau_{\text{baseline}} \tag{18}$$

This reduction is particularly valuable for scalable cloud-based credit scoring systems, where minimizing processing time directly translates to lower server costs and improved user experience in real-time "Buy Now, Pay Later" (BNPL) services.

4.2. Financial Risk and Ethical Implications

In credit scoring, predictive accuracy must be balanced against financial risk types. A False Positive corresponds to approving a loan for a defaulter, leading to direct capital loss. A False Negative corresponds to rejecting a solvent applicant, resulting in opportunity cost (lost interest revenue).

My experimental results indicate that the optimized model maintains a high F1 Score (0.906), suggesting a stable balance between Precision and Recall. Importantly, the quantization process did not introduce a bias shift; the decision boundary remained robust. This stability is critical for regulatory compliance.

Under regulations such as the EU AI Act or GDPR, financial institutions must provide "meaningful information about the logic involved" in automated decisions. The application of Taylor-based pruning enhances this interpretability. By physically removing 30% of neurons that had low saliency scores, I effectively filtered out "noise" from the decision process. The resulting compact model works faster using less resources and, more importantly, is potentially less prone to overfitting on false correlations, aligning with the principles of Trustworthy AI.

5. Conclusion

This study presented a mathematical framework for neural network compression. By deriving pruning criteria from the Taylor series expansion of the loss function [5], I avoided heuristic methods, ensuring that the removed neurons were indeed statistically redundant. The combination of this structural pruning with affine quantization yielded a model that is 3.35 times smaller than the baseline (reducing storage from 63 KB to 19 KB) while retaining a high classification accuracy of nearly 90%.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] D. Zhong, C. Huang, W. Ma, L. Deng, J. Zhou, Y. Xia, Enhanced prediction of pipe failure through transient simulation-aided logistic regression, Reliability Engineering & System Safety 260 (2025) 110913.
[2] A. Ostovar, D. D. Davari, M. Dzikuć, Determinants of design with multilayer perceptron neural networks: A comparison with logistic regression, Sustainability 17 (2025) 2611.
[3] A. Jaszcz, D. Połap, Aimm: Artificial intelligence merged methods for flood ddos attacks detection, Journal of King Saud University-Computer and Information Sciences 34 (2022) 8090-8101.
[4] S. K. Sahu, A. Mokhade, N. D. Bokde, An overview of machine learning, deep learning, and reinforcement learning-based techniques in quantitative finance: recent progress and challenges, Applied Sciences 13 (2023) 1956.
[5] P. Molchanov, S. Tyree, T. Karras, T. Aila, J. Kautz, Pruning convolutional neural networks for resource efficient inference, 2017. URL: https://arxiv.org/abs/1611.06440. arXiv:1611.06440.
[6] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, D. Kalenichenko, Quantization and training of neural networks for efficient integer-arithmetic-only inference, 2017. URL: https://arxiv.org/abs/1712.05877. arXiv:1712.05877.
[7] A. L. Maas, Rectifier nonlinearities improve neural network acoustic models, 2013. URL: https://api.semanticscholar.org/CorpusID:16489696.
[8] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/ 8a20a8621978632d76c43dfd28b67767-Paper.pdf.
[9] Y. LeCun, J. Denker, S. Solla, Optimal brain damage, in: D. Touretzky (Ed.), Advances in Neural Information Processing Systems, volume 2, MorganKaufmann, 1989. URL:

https://proceedings.neurips.cc/paper_files/paper/1989/file/ 6c9882bbac1c7093bd25041881277658-Paper.pdf.
[10] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. URL: https://arxiv.org/abs/1502.03167. arXiv:1502.03167.
[11] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Learning Research 15 (2014) 1929-1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.
[12] W. Falcon, The PyTorch Lightning Team, Pytorch lightning, 2019. URL: https://github.com/Lightning-AI/lightning, software available from https://github.com/LightningAI/lightning.
[13] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, 2019. URL: https://openreview.net/forum?id= Bkg6RiCqY7.