Is Hyperbolic Learning the Key to Better KAN?

Evaluating HyperKAN for Lead Scoring

Jia Yee Lee1, a), Chi Wee Tan1, b), Chuk Fong Ho1, c) and Noor Aida Husaini1, d)

1Faculty of Computing and Information Technology, Tunku Abdul Rahman University of Management and Technology, Jalan Genting Kelang, Setapak, 53300 Kuala Lumpur, Malaysia.

*a) Corresponding author: jylee-wm19@student.tarc.edu.my  
b) chiwee@tarc.edu.my*

*c) cfho@tarc.edu.my*

*d) nooraida@tarc.edu.my*

**Abstract.** Hyperbolic learning has recently gained attention for its effectiveness in deep learning models, particularly due to its tree-like structure, which is well-suited for addressing hierarchical problems compared to traditional Euclidean space. To tackle the challenges of high-dimensional data in lead scoring, this paper proposes HyperKAN, a novel integration of hyperbolic learning into the Kolmogorov-Arnold Networks (KAN) framework. The model's performance was evaluated on a publicly available dataset across different combinations of data split ratios and random states. Results indicate that the HyperKAN model achieved strong recall and F1 scores, demonstrating improved consistency over the standalone KAN model. After fine-tuning, the best configuration of the HyperKAN model, using a 60:20:20 train-test-validation split and a random state of 7, achieved 93.94% of accuracy, 93.93% of precision, 93.94% of recall, and 93.93% of F1 score. It outperformed the untuned KAN model in precision, recall, and F1 score, though not in accuracy. However, the HyperKAN model was surpassed by a neural network variant that incorporated hyperbolic learning, early stopping, and a learning rate of 0.0001, achieving higher performance across all metrics while also training more efficiently. These findings highlight a trade-off between performance gains and computational efficiency when applying hyperbolic learning within the KAN framework. While the HyperKAN model reveals potential improvements over the standalone KAN model in specific metrics, it is not an all-rounder model for the specific dataset. Further work is needed to refine its architecture, reduce training time without sacrificing performance, and assess its generalizability across different datasets and domains.

# Introduction

Lead scoring is a critical process in sales and marketing, where businesses assess and classify potential customers (known as leads) based on their likelihood of being converted into paying customers [1], [2]. Over time, the literature on lead scoring has evolved significantly, from traditional to advanced methodologies, to enhance the accuracy and efficiency of this process [3]. In previous lead scoring studies [4], [5], [6], [7], [8], [9], [10], it is generally assumed that features are represented in a flat Euclidean space when using basic Neural Network (NN) or Deep Learning (DL) models, unless otherwise specified. However, real-world data in lead scoring often exhibit complex, higher-dimensional, graph-structured, or hierarchical structures [11] that standard Euclidean models may struggle to capture effectively [12]. Additionally, high-dimensional data can suffer from the “curse of dimensionality”, where model performance declines as the number of features increases [13]. For instance, Chaudhuri et al. [8]utilized a multidimensional e-commerce dataset containing over 50,000 unique web sessions to predict user purchase intent. Similarly, Slakey et al. [11] worked with a dataset containing a range between 18,162 to 346,727 dimensions, stemming from high-cardinality categorical columns. As a result, lead scoring models may struggle to handle such complex, high-dimensional, and hierarchical data, limiting their ability to identify the most relevant leads or accurately predict conversions.

To address these challenges, this study explores the use of hyperbolic learning into DL architectures, with the goal of more effectively capturing the underlying structure of complex lead scoring data. While hyperbolic learning has shown promising results in other domains where hierarchy is critical [14], [15], [16], [17], [18], [19], its application to lead scoring remains unexplored. In response to this research gap, the current work introduces HyperKAN, a novel architecture that combines Kolmogorov-Arnold Networks (KAN) with hyperbolic learning. The proposed model leverages the advantage of spline-based univariate transformation capabilities of the KAN model and augments them with the representational power of hyperbolic learning, which inherently supports exponential expansion and hierarchical data relationships. By extending hyperbolic learning into the lead scoring domain and integrating it with the flexible KAN framework, HyperKAN is designed to enhance the model's capacity to learn non-Euclidean structures in lead behavior data.

# Related Works

## Hyperbolic Learning

Hyperbolic learning, or hyperbolic machine learning, refers to the application of hyperbolic geometry in machine learning (ML) algorithms to more effectively capture complex relational structures in data [20]. Unlike traditional Euclidean geometry, which assumes a flat data space, hyperbolic geometry operates in non-Euclidean spaces with constant negative curvature [21], [22]. This curvature enables hyperbolic spaces to be well-suited to represent hierarchical or tree-like structures [23], which are often found in lead scoring scenarios, such as customer segmentation [24], purchasing behavior [25], and customer journey paths [26]. Mathematically, hyperbolic spaces are modeled as Riemannian manifolds with constant curvature -1, providing geometric properties that help preserve latent hierarchies and relational distances in the data [27]. Several analytic models exist for hyperbolic space, each offering distinct geometric formulations useful for embedding and inference tasks. These include the Poincaré Disk, Poincaré Half-Plane, Beltrami-Klein, Minkowski Hyperboloid, and Poincaré Half-Space models [20], [21], [28].

Recent advances in hyperbolic learning have proven highly effective in domains where hierarchical or graph-structured data are prevalent, including natural language processing (NLP), computer vision (CV), and graph learning (GL). In the domain of NLP, Xu et al. [14] proposed a novel framework that incorporates hyperbolic embeddings to represent both words and topics. This approach significantly outperformed traditional Euclidean-embedded topic models in terms of classification accuracy. Similarly, López and Strube [15] introduced a fully hyperbolic model for fine-grained entity classification, achieving an improvement of the F1 score ranging from 5.1% to 16.2%, while significantly reducing the required parameter size by 70% to 91%. In the field of CV, Bdeir et al. [16] demonstrated that the use of a hyperbolic framework resulted in up to 1.5% higher classification accuracy in standard vision tasks, outperforming both Euclidean and hybrid decoder baselines, especially in adversarial and lower dimensional settings. Not only that, Yan et al. [17] applied hyperbolic learning to an unsupervised metric learning method based on hierarchical similarity, achieving recall improvements ranging from 3.6% to 5.0% across several datasets. In the area of GL, Du et al. [18] introduced a federated learning framework in hyperbolic space, specifically designed for graph classification tasks. Their approach showed an accuracy improvement of up to 15.6% compared to the baseline model and its variations. In addition, Yang et al. [19] proposed a hyperbolic-informed embedding model that leverages hierarchical information derived from the hyperbolic distance of nodes to the origin, and achieved a 21.4% improvement over existing baseline models.

Despite the advancements of hyperbolic learning in diverse research areas, its application in lead scoring has yet to be explored. Given the promising results across domains like NLP, CV, and GL, there is a significant opportunity to investigate how hyperbolic embeddings could potentially enhance lead scoring models, particularly in capturing the hierarchical relationships inherent in customer data and behavior patterns. Interestingly, a recent study by Lin et al. [20] proposed a framework that applies hyperbolic learning to tree edit distance for user behavior analysis. In their work, user behavior sequences, derived from historical website actions, are represented as tree structures and analyzed using the Poincaré disk model. While the study has not yet presented quantitative or qualitative results, it highlights the potential of hyperbolic embeddings to analyze user similarity in the e-commerce domain. This approach shares similarities with lead scoring, as it involves analyzing patterns in user interactions, an aspect that could also be relevant for modeling lead behavior and prioritizing leads based on their likelihood to convert. This opens a potential avenue for further research into how hyperbolic embeddings might be leveraged to improve lead scoring models by better capturing the dynamic and hierarchical nature of customer interactions.

## Kolmogorov-Arnold Networks

Kolmogorov-Arnold Networks (KANs), initially proposed by Liu et al. [30], represent a recent advancement in NN architectures based on the Kolmogorov-Arnold representation theorem. By replacing traditional linear weights and fixed activation functions with learnable univariate functions implemented via B-splines, KAN offers an alternative to standard Multi-Layer Perceptrons (MLPs). In the original work of Liu et al. [30], KAN demonstrated promising improvements in predictive accuracy while maintaining a relatively small model size, particularly in science-related tasks such as fitting physical equations and solving partial differential equations. Building on this foundation, the author's previous work [31] explored the application of KAN in the field of lead scoring and found that it exhibits superior results compared to other eight ML algorithms, including Decision Tree (DT), higher-order NN (HONN), K-Nearest Neighbors (KNN), Logistic Regression (LR), Naive Bayes (NB), NN, Random Forest (RF), and Support Vector Machines (SVM). Despite its potential, both the Liu et al.'s and the author’s studies highlighted training inefficiencies as a key practical limitation, especially when KAN is applied to large-scale or time-sensitive tasks.

The present study builds on this foundation by exploring how hyperbolic learning can be integrated and optimized within the architecture of the KAN model. In this context, enhancements to the KAN model are investigated with the aim of improving both predictive effectiveness and computational efficiency compared to the original KAN models.

## Lead Scoring

Despite the growing interest in hyperbolic learning, its application in lead scoring remains unexplored. This study addresses the gap by investigating the current methodologies used in this domain. Existing studies [4], [5], [6], [7], [8], [9], [10] on predictive modeling for lead scoring have focused primarily on implementing NN and DL approaches. The seven reviewed studies reveal the frequent use of NN architectures, including basic NNs, deep NNs, Feedforward Neural Networks (FNNs), recurrent NNs, and MLP. Five of these studies reported that the NN or DL models yielded the best results in their respective experiments. However, not all findings were consistent with this trend. For instance, Espadinha-Cruz et al. [9] found that ensemble learning outperformed NN models, while Nygård and Mezei [6] identified RF as the top performing model. Among the reviewed studies, the highest reported performance was achieved by the MLP model developed by Choudhury and Nur [4], which attained an accuracy of 99.41%, recall of 99.68%, and precision of 98.93%. Notably, Puravankara and Babu [7] employed the same dataset used in the present study. Their FNN model produced a promising accuracy of 94%, providing a direct benchmark for performance comparison. While the reviewed studies demonstrated promising results using traditional Euclidean-based models, none of the studies explored the use of hyperbolic learning, thereby overlooking the potential advantages offered by hyperbolic geometry in capturing complex hierarchical data structures. By integrating hyperbolic learning techniques into existing NN-based models, the present study aims to enhance model performance, ultimately offering a methodological contribution to the lead scoring literature.

# Methodology

This paper extends a prior study [31], which benchmarked the performance of the KAN model against eight ML models, including DT, HONN, KNN, LR, NB, NN, RF, and SVM. The earlier work demonstrated the competitive advantages of the KAN model, which consistently outperformed eight ML models in terms of performance metrics such as accuracy, precision, recall, and F1 score. Building upon this foundation, the present study introduces HyperKAN, an extension of the KAN model that incorporates hyperbolic learning to further enhance performance. Specifically, the study investigates whether incorporating hyperbolic learning can enhance the predictive capabilities of the model, using performance metrics such as accuracy, precision, recall, and F1 score, similar to the evaluation approach in the previous study.

To enable direct comparison with the prior study by the author, the present study utilizes a publicly available dataset commonly used in lead scoring research [7], [32], [33], [34]. The dataset originates from a case study developed by UpGrad and IIIT Bangalore, focusing on an educational provider, here referred to as [X Education](https://www.kaggle.com/datasets/amolbhone/lead-score-case-study?select=lead+scoring+case+study.pdf). This company offers personalized online courses for professionals. The dataset contains 9,240 entries with 37 variables, collected through various digital marketing channels such as website forms, online search engines, and referrals. The data capture user information and engagement activity with the primary goal of predicting course enrollment outcomes. The target variable is binary, indicating whether a user enrolled in a course. The class distribution is slightly imbalanced, comprising 5,679 non-enrollment instances (Class 0) and 3,561 enrollment instances (Class 1), which translates to an approximate ratio of 61.5% to 38.5%.

To prepare the dataset for modeling, a structured preprocessing pipeline was employed. First, identifier columns such as “Prospect ID” and “Lead Number” were excluded, as they serve no predictive purpose. Next, the categorical features with only one unique value were removed due to their lack of variance and discriminative power. Entries labeled as “Select” in categorical features were treated as missing value. This label was interpreted as a placeholder resulting from user inaction during form completion, suggesting a data entry issue rather than an informative response. Following this, separate preprocessing pipelines were applied to numerical and categorical features using a ColumnTransformer. For numerical features, missing values were imputed with a constant value of zero, followed by feature scaling via StandardScaler to ensure standardized input distributions with zero mean and unit variance. For categorical features, missing values were imputed with a placeholder value of “Unknown”. Subsequently, one-hot encoding was applied to convert categorical variables into a binary matrix representation.

The baseline NN model was developed using the TensorFlow library [35]. It consisted of an input layer followed by two hidden layers, each with 128 neurons and Rectified Linear Unit (ReLU) activation functions. The output layer comprised a single neuron with a sigmoid activation function to generate binary predictions. The model was compiled using the Riemannian Adam optimizer with a learning rate of 0.001 and binary cross-entropy as the loss function over 50 epochs with a batch size of 8. Other than that, the MLP model was implemented using the MLPClassifier from the Scikit-learn library [36]. This model was trained with default hyperparameter settings, which included a single input and hidden layer with 100 units, ReLU activation, and the Adam optimizer. Further experiments were conducted on both the baseline NN and MLP models. These included applying early stopping with a patience of 3 epochs, testing alternative learning rates of 0.01 and 0.0001, and extending to support hyperbolic learning. For the latter, the models were rebuilt using the HypLL library [37], employing a Poincaré ball model with a learnable curvature parameter. In parallel, the KAN model was implemented using the PyKAN library proposed by Liu et al. [30]. The model was trained with the library's default settings, consisting of an input layer with two neurons, one hidden layer with five neurons, and an output layer with one neuron.

Building on this foundation, the proposed HyperKAN model integrates hyperbolic learning into the KAN framework, aiming to combine their theoretical strengths for improved performance in lead scoring tasks. Specifically, the HyperKAN model extends the original KAN architecture into hyperbolic space by adopting the Poincaré ball manifold with learnable curvature. Figure 1 presents the pseudocode for the proposed HyperKAN model. The architecture consists of stacked hyperbolic linear layers implemented using the HypLL library, each followed by a hyperbolic ReLU activation function. The training processing begins by projecting Euclidean input data to the tangent space of the Poincaré ball at the origin. These projections are then mapped onto the hyperbolic manifold via exponential maps and propagated through two hidden layers with 64 and 32 units, respectively. The final prediction is produced in hyperbolic space using the Riemannian Adam optimizer, which adapts the gradient descent for hyperbolic learning. The hyperbolic implementation enables more efficient representation of hierarchical data patterns compared to its Euclidean counterparts while maintaining theoretical guarantees of the original KAN architecture.

|  |
| --- |
| **FIGURE 1.** Pseudocode of the proposed HyperKAN architecture. The algorithm outlines the initialization procedure and the forward pass, including the construction of hyperbolic linear layers and activation functions to the specified manifold |

The experiments were conducted on a local workstation equipped with a 13th Gen Intel® Core™ i7-13700F processor (16 cores, 24 threads, base clock 2.10 GHz), 64 GB of RAM, and an NVIDIA GeForce RTX 4070 Ti GPU with 12 GB of VRAM. The system runs a 64-bit Windows operating system. Experiments were implemented using PyTorch 2.1 with GPU acceleration enabled via CUDA 12.6 and cuDNN 8.9. To ensure a fair comparison between different models, the experiments were conducted using a fixed data split ratio of 60:20:20 (training, validation, and test sets) and a fixed random state of 42, without any hyperparameter fine-tuning. Besides that, the experiment also optimizes the HyperKAN model by fine-tuning its hyperparameters, including grid size, k (piecewise polynomial order), steps, and λ (lambda). To observe the generalizability of the HyperKAN model, experiments are conducted on different train-test-validation split ratios and random states. Ultimately, the research will conclude by identifying whether the use of hyperbolic learning improves model performance in terms of accuracy, precision, recall, F1 score, and training duration, which are the same performance metrics used in the author's previous work.

# Results and Discussions

Table 1 presents a comparative evaluation of ten NN-based models on the X Education dataset assessed across accuracy, precision, recall, F1 score, and training duration in minutes. The proposed HyperKAN achieved a respectable performance, with an accuracy of 92.50%, precision of 92.57%, recall of 92.50%, and F1 score of 92.42%. While it did not outperform the standalone KAN model in accuracy or precision, the HyperKAN model achieved a more balanced recall and F1 score, suggesting that incorporating hyperbolic learning may improve the model’s ability to identify lead cases effectively. Compared to the highest performance in each metrics, the HyperKAN model exhibited a drop of approximately 1.80% in accuracy, 1.49% in precision, 1.55% in recall, and 1.60% in F1 score. Although these differences are relatively modest, they may be attributed to the additional architectural complexity introduced by integrating hyperbolic learning into the KAN framework. This suggests that the current implementation of the HyperKAN model may not yet fully take advantage of its theoretical benefits, highlight a need for improved hybrid design strategies. The standalone KAN model outperformed all other models with the highest accuracy of 94.30%, probably due to its novel use of learnable activation functions applied to edges instead of fixed activations in neurons.

| **TABLE 1.** Comparison results across nine neural network models on the X Education dataset. Unless otherwise specified, models use a default learning rate of 0.001. Results for accuracy, precision, recall, F1 score, and training duration are rounded to two decimal places. | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1 Score (%)** | **Duration (min)** |
| NN | 92.46 | 92.54 | 92.46 | 92.48 | 0.85 |
| NN - ES[[1]](#footnote-1) | 93.29 | 93.29 | 93.29 | 93.29 | 0.17 |
| NN - H[[2]](#footnote-2) | 93.18 | 93.17 | 93.18 | 93.17 | 32.00 |
| NN - H& ES | 91.38 | 91.36 | 91.38 | 91.36 | 6.79 |
| NN - H& ES& LRate[[3]](#footnote-3) of 0.01 | 91.81 | 92.05 | 91.81 | 91.86 | 6.78 |
| NN - H& ES& LRateof 0.0001 | 94.05 | **94.05** | **94.05** | **94.02** | 29.57 |
| MLP | 93.72 | 93.72 | 93.72 | 93.72 | **0.09** |
| MLP - H | 92.93 | 93.07 | 92.93 | 92.84 | 6.57 |
| KAN | **94.30** | 93.12 | 91.70 | 92.40 | 33.46 |
| HyperKAN (proposed model) | 92.50 | 92.57 | 92.50 | 92.42 | 20.95 |

However, the superior accuracy performance came at the cost of the longest training duration of 33.46 minutes among all models. The prolonged training time can be attributed to the modeling of spline-based univariate functions, which is computationally intensive. Interestingly, the model variation that combined hyperbolic learning, early stopping, and a learning rate of 0.0001 achieved impressive results of second highest accuracy, along with the highest precision, recall, and F1 score across all models. The results suggest that hyperbolic learning may help capture underlying patterns and relationships in user behavior under certain configurations. As for the MLP model, it was by far the fastest to train, in just 0.09 minutes, but this came at a slight cost to its predictive accuracy. In general, models that incorporated hyperbolic learning, whether in NN, MLP, or KAN variants, tended to take much longer to train. This trend highlights that while model performance improves, it comes with higher computational costs. This raises practical concerns about whether hyperbolic learning is truly efficient for real-time or large-scale use.

To further assess the robustness of the HyperKAN model, experiments were also conducted across different train-test-validation splits and random states without applying fine-tuning, as shown in Table 2. The model achieved its best performance under a 60:20:20 split and a random state of 7, reaching an accuracy of 92.71%, precision of 93.00%, recall of 92.71%, and F1 score of 92.61%. However, this configuration also incurred the longest training duration at 28.21 minutes, indicating a trade-off between model performance and computational efficiency. On the other hand, the same random state with a 70:15:15 split yielded the lowest scores across all performance metrics, with an accuracy of 88.06%, precision of 89.64%, recall of 88.06%, and F1 score of 87.58%. Even though there was a drop in performance, the training time was much shorter, with only 3.79 minutes. In other words, allocating less data for validation and testing can speed up training, but it may come at the expense of the model’s ability to generalize to new data. Overall, the results show that HyperKAN’s performance varies depending on how the data is split and the random seed used. Such sensitivity suggests that careful selection of data splits and seeds is critical when deploying the model in real-world applications.

| **TABLE 2.** Comparison results of the HyperKAN model across different train-test-validation splits and random states without fine-tuning. Results for accuracy, precision, recall, F1 score, and training duration are rounded to two decimal places. The best result in each column is highlighted in bold. | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Train-test-validation split** | **Random State** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1 Score (%)** | **Duration (min)** |
| 60:20:20 | 7 | **92.71** | **93.00** | **92.71** | **92.61** | 28.21 |
| 42 | 92.50 | 92.57 | 92.50 | 92.42 | 20.95 |
| 70:15:15 | 7 | 88.06 | 89.64 | 88.06 | 87.58 | **3.79** |
| 42 | 92.61 | 92.85 | 92.61 | 92.51 | 25.25 |

To explore the potential benefits of optimization, the HyperKAN model was also evaluated under the same data splits and random states combinations, but this time with fine-tuning applied. The results are summarized in Table 3 and show performance improvements under all configurations. Overall, the configuration with a 60:20:20 split and a random state of 7 achieved the best overall results, with an accuracy of 93.94%, precision of 93.93%, recall of 93.94%, and an F1 score of 93.93%, outperforming all configurations regardless of with and without fine-tuning. This improvement came at a significant cost, with 47.77 minutes of training, nearly double the duration required without fine-tuning, highlighting a clear trade-off in computational cost. Interestingly, the fastest training time of 15.78 minutes was still observed under the 70:15:15 split with a random state of 7, showing that certain configurations may provide a practical trade-off between performance and efficiency. Overall, these results emphasize that while fine-tuning increases computational efficiency, it plays a crucial role in boosting accuracy and performance stability. Notably, a configuration with a 70:15:15 split and a random state of 42 was the only one to exhibit a slight decline in performance after fine-tuning compared to its untuned counterpart. This degradation is likely due to the limitations of the fine-tuning process itself, which was capped at 200 hyperparameter search trials due to computational constraints. Such a cap may have prevented the optimizer from identifying a more optimal configuration for this specific combination of data split and random state.

| **TABLE 3.** Comparison results of the HyperKAN model across different train-test-validation splits and random states with fine-tuning. Results for accuracy, precision, recall, F1 score, and training duration are rounded to two decimal places. The best result in each column is highlighted in bold. | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Train-test-validation split** | **Random State** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1 Score (%)** | **Duration (min)** |
| 60:20:20 | 7 | **93.94** | **93.93** | **93.94** | **93.93** | 47.77 |
| 42 | 93.61 | 93.60 | 93.61 | 93.59 | 47.19 |
| 70:15:15 | 7 | 92.92 | 92.92 | 92.92 | 92.88 | **15.78** |
| 42 | 92.58 | 92.65 | 92.58 | 92.60 | 35.59 |

# Conclusion

This paper explores HyperKAN, a new method that combines hyperbolic learning with the KAN model in an effort to improve lead scoring. In the early test, before the fine-tuning process, the HyperKAN model did not outperform the standalone KAN model in terms of accuracy, precision and training duration. However, it did show better performance in both recall and F1 score, which suggests it was more reliable in identifying potential leads and useful in recall-sensitive scenarios where missing a lead is more costly than flagging a false positive. Once fine-tuning was applied, the HyperKAN model showed moderate improvements: accuracy increase by 1.56%, precision by 1.47%, recall by 1.56%, and F1 score by 1.63%. Even with these gains, the HyperKAN model did not surpass the top-performing models in the study, which was achieved by an NN variant that used hyperbolic learning along with early stopping and a learning rate of 0.0001. This model delivered the second-highest accuracy overall, and topped the charts for precision, recall, and F1 score. While the HyperKAN model shows potential in certain use cases, it is not yet the best option for this particular lead scoring dataset. One key takeaway from the study is that there is a clear trade-off between model performance and training duration. When hyperbolic learning was implemented to KAN, as well as to the NN and MLP models, training took noticeably longer, reflecting the computational cost that comes with increased model complexity. The performance of the HyperKAN also shifted depending on the data split and random state, which highlights how important it is to tune these settings carefully when deploying the model.

Other than that, there are several limitations of this study worth noting. The experiments were conducted on a single dataset, and the fine-tuning process was capped at 200 hyperparameter trials due to constraints in computing resources. Moreover, the extended training time of the HyperKAN model also raises concerns about the model’s practicality in real-time or resource-constrained environments.

Ultimately, this study takes a step toward understanding how hyperbolic learning could work and be adapted for practical applications in lead scoring. Future research could explore more efficient fine-tuning approaches to cut down on training time without compromising performance. In addition, further development of the HyperKAN's architecture or integration of hyperbolic learning with other model types could also better harness the advantages of hyperbolic learning. Furthermore, it might be useful to evaluate the generalizability of the HyperKAN model across different datasets and domains to determine whether it is scalable for real-time environments where both speed and prediction performance matter.

# Acknowledgments

This research is supported by Tunku Abdul Rahman University of Management and Technology (TAR UMT) through the research grant of UC/I/G2022-00098.

# References

1. J.P. Monat, “Industrial sales lead conversion modeling,” Marketing Intelligence & Planning **29**(2), 178–194 (2011).
2. M. Wu, P. Andreev, and M. Benyoucef, “The state of lead scoring models and their impact on sales performance,” Inf Technol Manag **25**(1), 69–98 (2024).
3. J.Y. Lee, C.W. Tan, C.F. Ho, and N.A. Husaini, “Predictive Lead Scoring Models: A Contemporary Review of Identified Gaps and Future Research Directions (unpublished),” (2024).
4. A.M. Choudhury, and K. Nur, “A Machine Learning Approach to Identify Potential Customer Based on Purchase Behavior,” in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, (IEEE, Dhaka, Bangladesh, 2019), pp. 242–247.
5. G. Giorcelli, “Variable-sized input, character-level recurrent neural networks in lead generation: predicting close rates from raw user inputs,” (2019).
6. R. Nygård, and J. Mezei, *Automating Lead Scoring with Machine Learning: An Experimental Study* (2020).
7. R. Puravankara, and C. Narendra Babu, “Lead Forecasting using LSTM based Deep Learning Architecture for Sentiment Analysis,” in *2020 3rd International Conference on Information and Communications Technology (ICOIACT)*, (IEEE, Yogyakarta, Indonesia, 2020), pp. 159–164.
8. N. Chaudhuri, G. Gupta, V. Vamsi, and I. Bose, “On the platform but will they buy? Predicting customers’ purchase behavior using deep learning,” Decision Support Systems **149**, 113622 (2021).
9. P. Espadinha-Cruz, A. Fernandes, and A. Grilo, “Lead management optimization using data mining: A case in the telecommunications sector,” Computers & Industrial Engineering **154**, 107122 (2021).
10. S. Binte Ayaz, “Lead Scoring with Machine Learning,” (2023).
11. A. Slakey, D. Salas, and Y. Schamroth, “Encoding Categorical Variables with Conjugate Bayesian Models for WeWork Lead Scoring Engine,” (2019).
12. P. Kontkanen, J. Lahtinen, P. Myllymäki, T. Silander, and H. Tirri, “Supervised model-based visualization of high-dimensional data,” IDA **4**(3–4), 213–227 (2000).
13. N. Altman, and M. Krzywinski, “The curse(s) of dimensionality,” Nat Methods **15**(6), 399–400 (2018).
14. Y. shi Xu, D. Wang, B. Chen, R. Lu, Z. Duan, and M. Zhou, “HyperMiner: Topic Taxonomy Mining with Hyperbolic Embedding,” Advances in Neural Information Processing Systems **35**, 31557–31570 (2022).
15. F. López, and M. Strube, “A Fully Hyperbolic Neural Model for Hierarchical Multi-Class Classification,” (2020).
16. A. Bdeir, K. Schwethelm, and N. Landwehr, “Fully Hyperbolic Convolutional Neural Networks for Computer Vision,” (2024).
17. J. Yan, L. Luo, C. Deng, and H. Huang, “Unsupervised Hyperbolic Metric Learning,” (2021), pp. 12465–12474.
18. H. Du, C. Liu, H. Liu, X. Ding, and H. Huo, “An efficient federated learning framework for graph learning in hyperbolic space,” Knowledge-Based Systems **289**, 111438 (2024).
19. M. Yang, M. Zhou, R. Ying, Y. Chen, and I. King, “Hyperbolic Representation Learning: Revisiting and Advancing,” in *Proceedings of the 40th International Conference on Machine Learning*, (PMLR, 2023), pp. 39639–39659.
20. T.-Y. Lin, Y.-H. Liang, H. Uno, and T.H. Lee, “Hyperbolic-Based Tree Edit Distance as Similarity of User Behavior,” in *2024 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, (IEEE, Danang, Vietnam, 2024), pp. 1–4.
21. J.W. Cannon, W.J. Floyd, R. Kenyon, and W.R. Parry, “Hyperbolic geometry,” Flavors of Geometry **31**(59–115), 2 (1997).
22. Z.A.M. Makari, and F.D. Ali, “Geometric Transformations and Their Applications in Non-Euclidean Spaces,” African Journal of Advanced Pure and Applied Sciences (AJAPAS), 184–192 (2024).
23. S. Ghosh, and S. Das, “Consistent Spectral Clustering in Hyperbolic Spaces,” (2024).
24. H.-H. Zhao, X.-C. Luo, R. Ma, and X. Lu, “An Extended Regularized K-Means Clustering Approach for High-Dimensional Customer Segmentation With Correlated Variables,” IEEE Access **9**, 48405–48412 (2021).
25. F. Xia, and R. Chatterjee, “Multicategory choice modeling with sparse and high dimensional data: A Bayesian deep learning approach,” Decision Support Systems **157**, 113766 (2022).
26. R.A. Moral, Z. Chen, S. Zhang, S. McClean, G.R. Palma, B. Allan, and I. Kegel, “Profiling Television Watching Behavior Using Bayesian Hierarchical Joint Models for Time-to-Event and Count Data,” IEEE Access **10**, 113018–113027 (2022).
27. M. Bucataru, and D. Manea, “Discrete Laplacians on the hyperbolic space -- a compared study,” arXiv.Org, (2024).
28. D. Celinska-Kopczynska, and E. Kopczynski, “Numerical Aspects of Hyperbolic Geometry,” (2024).
29. T. Ji, Y. Hou, and D. Zhang, “A Comprehensive Survey on Kolmogorov Arnold Networks (KAN),” (2025).
30. Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T.Y. Hou, and M. Tegmark, “KAN: Kolmogorov-Arnold Networks,” (2025).
31. J.Y. Lee, C.W. Tan, C.F. Ho, and N.A. Husaini, “A Comparative Analysis of Machine Learning Algorithms for Lead Scoring Model (unpublished),” (2025).
32. W.Y. Yim, K.W. Khaw, S.T. Lim, and X. Chew, “Enhancing Conversions and Lead Scoring in Online Professional Education: DOI: https://doi.org/10.33093/ijomfa.2024.5.1.2,” International Journal of Management, Finance and Accounting **5**(1), 15–63 (2024).
33. M. Sharma, Identifying Factors Contributing to Lead Conversion Using Machine Learning to Gain Business Insights, masters, Dublin, National College of Ireland, 2023.
34. A. Jadli, M. Hamim, M. Hain, and A. Hasbaoui, “TOWARD A SMART LEAD SCORING SYSTEM USING MACHINE LEARNING,” INDJCSE **13**(2), 433–443 (2022).
35. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, “{TensorFlow}: A System for {Large-Scale} Machine Learning,” (2016), pp. 265–283.
36. L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. Vanderplas, A. Joly, B. Holt, and G. Varoquaux, “API design for machine learning software: experiences from the scikit-learn project,” arXiv.Org, (2013).
37. O. Ganea, G. Becigneul, and T. Hofmann, “Hyperbolic Neural Networks,” in *Advances in Neural Information Processing Systems*, (Curran Associates, Inc., 2018).

1. ES stands for Early Stopping [↑](#footnote-ref-1)
2. H stands for Hyperbolic [↑](#footnote-ref-2)
3. LRate stands for Learning Rate [↑](#footnote-ref-3)