Engineered Science
DOI: https://dx.doi.org/10.30919/es1703

Exploring Autoencoder-based Representations for Tabular Data Classification

Il'murat Tokhtakhunov,[1,2,*] Marat Nurtas,[1,3,*] Alexander Neftissov,[4,5,*] Sharofiddin Pirnaev,[6,*] Ilyas Kazambayev[4,5] and Lalita Kirichenko[4,5]

Abstract

Autoencoders are evaluated as a means of constructing compact and informative vector representations for classification tasks involving high-dimensional tabular data. The methodology addresses the limitations of traditional models that rely on manual feature engineering and task-specific training. Emphasis is placed on building a generalized look-alike model for targeted advertising, using embeddings derived from subscriber-related entities. The approach is assessed on a real-world telecommunications dataset comprising subscriber demographics, devices, tariffs, and network characteristics. Experimental results demonstrate that embeddings produced by autoencoders outperform classical dimensionality reduction methods such as Principal Component Analysis (PCA), both in predictive quality and computational efficiency. Compressed representations enable the identification of nonlinear patterns and semantic similarities, improving classification accuracy across multiple metrics. The study further introduces an integrated vector architecture by concatenating embeddings from heterogeneous entities. Cosine similarity is employed as a metric for identifying similar users, enabling the development of a scalable and automated recommendation service for Business-to-Business (B2B) applications. Performance is benchmarked using traditional quality metrics (precision, recall, Harmonic Mean of Precision and Recall (F1-score), Receiver Operating Characteristic – Area Under the Curve (ROC AUC)) as well as business-specific indicators such as conversion rate and lift. The findings support the applicability of autoencoders in modeling complex tabular structures with minimal information loss. Prospects include the development of domain-specific autoencoder ensembles and the exploration of alternative vector similarity metrics for broader industrial adoption. The suggested solution can be applied for water resource monitoring system as improvement for classification and further prediction.

Keywords: Autoencoder; Embedding; Cosine similarity distance; Look-a-Like model.
Received: 03 July 2025; Revised: 06 August 2025; Accepted: 13 August 2025
Article type: Research article.

1. Introduction

The rapid advancement and widespread integration of machine learning technologies have reached unprecedented levels. Organizations across a broad spectrum of industries are allocating substantial resources toward the development and deployment of machine learning solutions, motivated by tangible business outcomes such as revenue growth and cost optimization. As the Parallel to this, investments in big data infrastructure reflect a paradigm shift toward more sophisticated and data-driven operational frameworks. In recognition of the value derived from analytics and machine learning, businesses increasingly seek to preserve all available data, resulting in growing dataset volumes and a continual expansion in the number of features utilized in modeling tasks.[1]

The high dimensionality of features involved in modern machine learning introduces significant challenges, commonly associated with the curse of dimensionality.[2] This phenomenon results in elevated data sparsity within high-dimensional feature spaces, complicating the task of identifying informative patterns. Such sparsity adversely affects the learning process and often leads to overfitting, where models exhibit poor generalization to unseen data

[1]*Department of Mathematical and Computer Modelling, International Information Technology University, 34/1 Manas street, Almaty, 050000, Kazakhstan*
[2]*School of Digital Technologies, Narxoz University, 55 Zhandosov street, Almaty, 050035, Kazakhstan*
[3]*Faculty of Information technology, Al-Farabi Kazakh National University, 71 Al-Farabi Avenue, Almaty, 050040, Kazakhstan*

despite strong performance on training samples. Traditional feature engineering approaches, including manual selection and construction of variables, can enhance model performance. However, these methods are highly reliant on domain-specific expertise and may fail to account for complex and nonlinear interactions among variables. The increasing dimensionality further exacerbates the impracticality of manual feature creation, particularly when dealing with large-scale tabular datasets.[3]

Dimensionality reduction methods are intended to project data into more compact spaces while preserving core informational content. Techniques such as PCA and t-Distributed Stochastic Neighbor Embedding (t-SNE) have demonstrated utility in various applications but often fall short when faced with nonlinear patterns or intricate feature interactions.[4] While foundational, these techniques possess limitations that become especially pronounced in real-world machine learning problems involving constraints on computational and analytical resources. The challenge of constructing a flexible look-alike model exemplifies such limitations and forms the basis for the current research. The practical problem addressed in this context concerns the development of a generalized model capable of performing look-alike analysis for targeted advertising. The model is required to identify subscribers similar to a reference group, thereby expanding the audience pool for B2B clients. A critical requirement for the proposed solution involves the ability to fully automate the modeling process while maintaining, or exceeding, the predictive quality of existing solutions.

The evaluation metrics selected for performance comparison include standard binary classification indicators such as lift, precision, recall, F1-score, and ROC AUC. Additionally, business-specific measures, particularly the conversion rates observed across advertising campaigns, were incorporated to assess practical effectiveness.[2] The assessment protocol involves executing tests across ten distinct campaigns with varying input segments. Results obtained from these trials are aggregated to facilitate a comprehensive and unbiased comparison between the new and existing approaches. A departure from the conventional method—wherein a dedicated binary classifier is trained for each campaign—was motivated by the need to reduce analyst time and operational complexity.[5] Instead, the goal was to design a generalized look-alike modeling framework that could serve as the foundation for a scalable and automated service. Such a system would accept any given subscriber sample as input and retrieve similar profiles from the broader subscriber base. The implementation of this concept necessitates access to a broad and diverse set of features for analysis. A truly flexible look-alike model must accommodate arbitrary samples, with no predefined assumptions regarding the relevance of individual features to specific tasks.[6] Furthermore, the target service must support concurrent usage by numerous B2B clients, delivering rapid and reliable responses without accumulating significant task backlogs.[7] Consequently, achieving high processing speed emerged as a critical design criterion. However, traditional methods failed to provide adequate performance in terms of speed, accuracy, and feature adaptability.

These challenges prompted the exploration of more advanced modeling paradigms, particularly the application of autoencoders. As a class of neural network architectures, autoencoders offer a means of capturing intrinsic data representations through unsupervised learning. By encoding input data into a latent space of reduced dimensionality, autoencoders are capable of identifying complex structures and nonlinear dependencies with greater efficiency. The resulting embeddings encapsulate essential information in a compact format, addressing many of the deficiencies associated with classical techniques.[8]

In addition to the core exploration of autoencoder architecture, this study introduces a detailed investigation into their capacity to mitigate the curse of dimensionality and enhance the resilience of machine learning models trained on high-dimensional tabular data. Comparative analysis with widespread traditional methods provides further insight into their relative performance and potential advantages in operational settings.[9] A notable aspect of the contribution lies in the integration of several methodological innovations. These include the concatenation of embedding vectors from multiple distinct entities and the application of cosine distance as a similarity metric in the design of a flexible look-alike model. By employing autoencoders to generate dense vector representations, the approach offers a novel strategy for modeling complex data while simultaneously improving speed, accuracy, and feature richness.[10] Beyond the immediate application to targeted advertising, the study also explores the latent structure of the compressed embedding space and its implications for broader analytical tasks. The merging of entity-specific embeddings into a unified subscriber profile represents a distinctive method of capturing interrelated characteristics across domains such as device usage, tariff plans, and geographic behavior. The adoption of cosine similarity for evaluating proximity in the embedding space marks an additional methodological advancement. This metric supports nuanced comparisons of subscriber profiles based on compressed representations, enabling more accurate

[4]*Science Innovation Center Industry 4.0, Astana IT University, Mangilik El C1, Astana, 010000, Kazakhstan*
[5]*Academy of Physical Education and Mass Sports, Mangilik El B2.2, Astana, 010000, Kazakhstan*
[6]*Department of Engineering Technological Machines, Tashkent State Transport University, 1 Temiryolchilar street, Mirabad district, Tashkent, 100167, Uzbekistan*
[*]*Email:* ilmurat.tokhtakhunov@gmail.ru (I. Tokhtakhunov),
maratnurtas@gmail.com (M. Nurtas),
a_neftissov@apems.edu.kz (A. Neftissov),
sharofiddinpirnayev23061983@gmail.com (Sh. Pirnaev)

identification of similar users. Compared to conventional distance metrics, cosine similarity provides enhanced precision and interpretability when applied to normalized latent vectors.[11]

Overall, the findings contribute to the understanding of deep learning applications in the field of tabular data analysis. The proposed methods validate the potential of autoencoder-based representations in real-world settings and support their generalization to diverse application domains. Future development will focus on extending the model through the integration of multiple domain-specific autoencoders and investigating alternative approaches to vector similarity estimation.[12]

2. Abbreviations

Abbreviation	Meaning
PCA	Principal Component Analysis
B2B	Business-to-Business
F1-score	Harmonic Mean of Precision and Recall
ROC AUC	Receiver Operating Characteristic – Area Under the Curve
t-SNE	t-Distributed Stochastic Neighbor Embedding
MICE	Multiple Imputation by Chained Equations
PyTorch	Python Torch
ReLU	Rectified Linear Unit
MSE	Mean Squared Error
Adam	Adaptive Moment Estimation
GPU	Graphics Processing Unit
HDFS	Hadoop Distributed File System
DVC	Data Version Control
L2 Regularization	Ridge regularization (weight decay)
3D	3 Dimensional
SIM	Subscriber Identity Module
eSIM	Embedded Subscriber Identity Module
Wi-Fi	Wireless Fidelity
PR	Precision-Recall
SVM	Support Vector Machine
LGBM	Light Gradient Boosting Machine

3. Training dataset preparation

The proprietary training dataset was provided under a collaboration agreement with a telecommunications company. The task originated from a real-world business request directed to one of the authors, who serves as a data scientist in the operator's big data department. The objective was to construct a flexible look-alike model. For this purpose, data were extracted from internal datamarts responsible for generating user profiles. Multiple data sources were integrated to form a comprehensive and multidimensional dataset. All records were fully anonymized in strict accordance with corporate fairness policies and applicable data protection laws. Due to confidentiality constraints, the dataset is not available for public use. Nevertheless, access was granted within the scope of a research partnership, ensuring that all data handling adhered to stringent privacy and ethical standards.

The dataset consists of detailed records on approximately 900,000 subscribers, encompassing 948 features grouped into thematic domains, referred to as entities. The User Entity contains detailed subscriber profiles, including aggregated indicators of activity, traffic usage, personal interests, and socio-demographic characteristics. The Web Entity captures behavioral data derived from anonymized internet sessions, limited to information collected with explicit user consent. The Finance Entity provides a view of users' financial behavior based on anonymized banking transactions, enabling the construction of consumer profiles and personalized product groupings. The Device Entity contains technical descriptions of mobile devices, including specifications, price tiers, and distribution across demographic segments. The Cell Base Station Entity records information on the main base stations associated with each subscriber, including service quality metrics, signal stability, internet speed, and spatial usage patterns. Lastly, the Tariff Plan Entity comprises detailed information on the active service plans, including pricing, feature sets, and popularity within the user base.

The dataset includes both quantitative and categorical variables. As downstream methods, such as autoencoders and PCA, require numerical input, all categorical features were transformed using one-hot encoding.[13] Preprocessing was essential to prepare the data for subsequent model training and involved multiple operations, including normalization, missing value treatment, multicollinearity resolution, and additional feature transformations. To address multicollinearity, the pairwise correlation coefficient method was employed to detect linear dependencies among features.[14] When two variables exhibited a correlation above a specified threshold, the feature with lower informative value was excluded. The optimal correlation threshold was determined empirically through iterative experimentation, testing values between 0.5 and 1.0. Model performance was evaluated across various configurations, and a threshold of 0.77 was identified as delivering the best balance between precision, recall, and model stability. This threshold was consequently applied throughout the dataset to reduce redundancy and minimize the adverse effects of linear correlation, thereby improving overall model efficiency.[15]

Missing values presented another significant challenge across several entities. A variety of imputation strategies were considered to handle this issue. Although advanced methods such as Multiple Imputation by Chained Equations (MICE) were initially explored, they did not result in substantial performance gains compared to simpler approaches.[16] Furthermore, the computational demands of such methods proved impractical given the dataset's size and the constraints of a production environment where resource optimization is critical.[17] As a result, imputation was conducted using lightweight statistical techniques, including mean, median, and mode replacement, augmented by randomized imputation to enhance robustness. Each method was selected and applied

based on the nature and distribution of missing data within specific fields, ensuring accurate recovery of values while maintaining data consistency for modeling.[18]

Further data refinement was achieved through the application of isolation forests, an anomaly detection method based on the principles of Random Forests. This technique constructs binary trees in a recursive and stochastic manner, isolating data points that deviate significantly from the general distribution. Anomalies are characterized by shorter average path lengths in the trees, reflecting their distinctiveness. This approach proved effective in identifying and excluding outliers from the training dataset, thereby enhancing the stability and reliability of the resulting model.[19]

Another crucial step in the preprocessing pipeline involved feature scaling. Uniformity in feature magnitudes was necessary to avoid unintended biases during model training, especially in the context of autoencoders, which are sensitive to input scales.[20] Without appropriate normalization, features with larger numerical ranges may disproportionately influence model behavior. To address this, min-max normalization was applied to all numerical variables, transforming them into the [0, 1] range. This ensured equal contribution of all features during training. One-hot encoding was applied to categorical attributes, enabling the model to leverage non-numeric relationships effectively. The entire preprocessing procedure including encoding, imputation, scaling, and transformation—was applied consistently to both training and validation datasets to prevent distributional drift. This comprehensive standardization allowed the autoencoder to focus on learning meaningful representations across all feature dimensions, facilitating robust data reconstruction while preserving the underlying structure of the dataset.[21]

4. Autoencoder architecture and training process

At the next stage of the study, various autoencoder architectures were constructed and trained. The autoencoder consists of two primary components: the encoder and the decoder. The encoder processes the input feature vector and compresses it into a lower-dimensional latent representation.[22] This component is designed to identify and retain the most informative attributes of the input data while effectively reducing its dimensionality. As a result, a more compact and abstract representation is produced, preserving the essential structural and semantic characteristics of the original data.

The decoder, in turn, receives the latent representation from the encoder and reconstructs the original input vector.[23] The objective of the decoder is to restore the input data from its compressed form while maintaining the integrity of key patterns and relationships. During training, the decoder learns to reproduce the data with high fidelity, ensuring the retention of important information.[24] Within the scope of the current task, the transformation of sparse input representations into dense, lower-dimensional embeddings was of particular importance. Such embeddings offer multiple advantages that contribute to the overall effectiveness of the model. First, they provide a compact representation of the original data, substantially reducing dimensionality and the number of trainable parameters, which, in turn, accelerates the training process and lowers computational resource requirements. In addition, embedding vectors capture semantic relationships

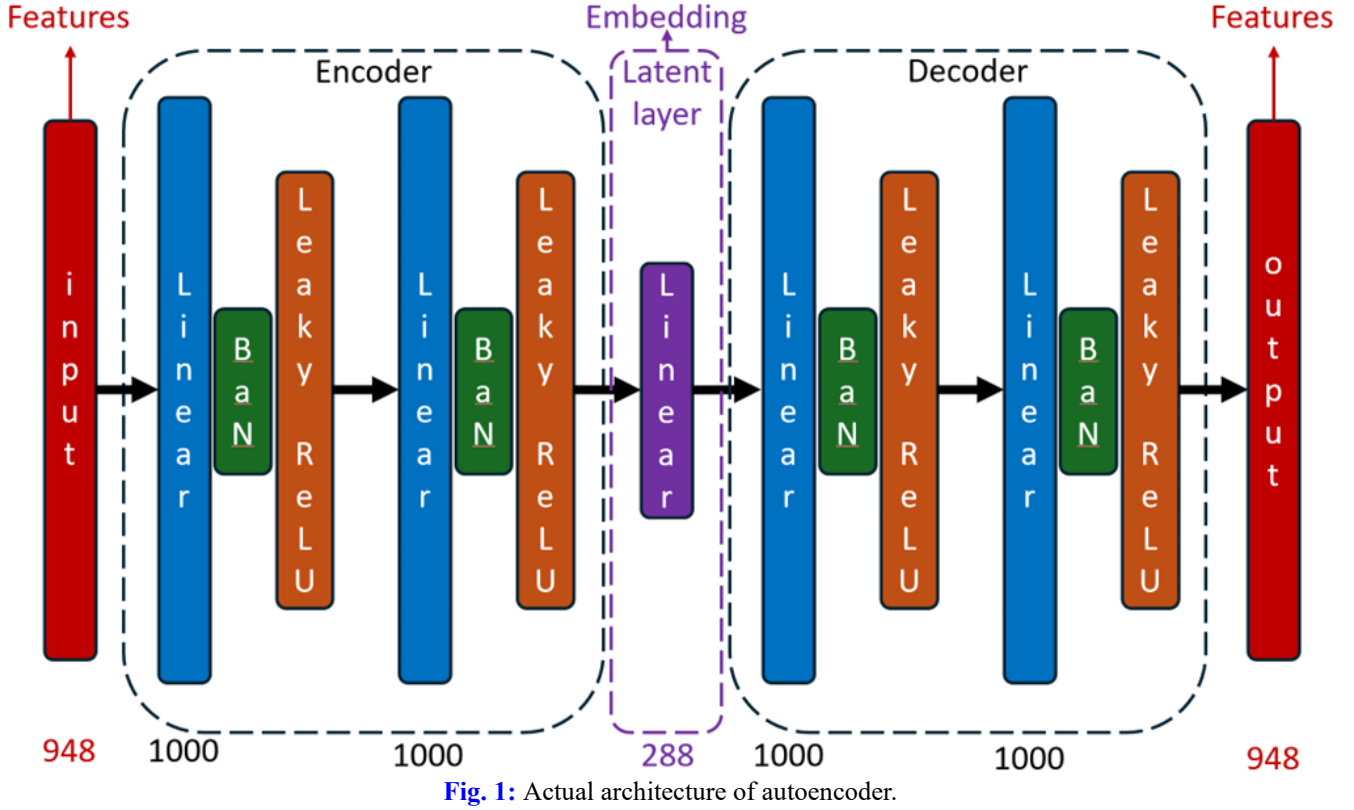

Fig. 1: Actual architecture of autoencoder.

between objects, enabling the identification of similar entities within the latent space.[25] This property is essential for modeling tasks that rely on similarity metrics. Furthermore, the embeddings possess a strong generalization capability, encapsulating abstract feature patterns that can be effectively utilized across various machine learning tasks, such as classification, regression, and clustering.[26]

A combined analytical and empirical approach was used to identify an optimal architecture. Different configurations were evaluated using a validation dataset, with the corresponding results tracked and stored in Metaflow for further analysis and comparison.[27] The final autoencoder architecture was selected following extensive experimentation, and its structure is presented in Fig. 1.

Model training was conducted using the Python Torch (PyTorch) framework, widely recognized for its flexibility and performance in deep learning applications.[28] All layers within the autoencoder utilized the Rectified Linear Unit (ReLU) as the activation function, with the exception of the output layer of the decoder, where the sigmoid activation function was applied. The ReLU activation provides non-linear transformations and helps address vanishing gradient issues, making it a standard choice in modern neural network architectures.[7] The sigmoid function, applied at the output layer, constrains values to the [0, 1] range, which is appropriate for reconstructing normalized input data.

To enhance training efficiency and model stability, batch normalization was incorporated into the architecture.[29] This technique normalizes layer activations within each mini-batch, reducing internal covariate shift and accelerating convergence. The inclusion of batch normalization contributed to improved generalization and a smoother training process.[30] The loss function employed during training was the Mean Squared Error (MSE), which measures the average squared difference between predicted and actual values. This function directly evaluates the reconstruction accuracy of the autoencoder and aligns with the objective of minimizing information loss during compression.[31]

Model optimization was performed using the Adaptive Moment Estimation (Adam) optimizer, which combines momentum and adaptive learning rates. This algorithm provides effective navigation through complex parameter spaces and accelerates convergence relative to traditional gradient descent methods.[32] To prevent overfitting, the Dropout technique was used during training. By randomly deactivating a subset of neurons in each iteration, Dropout introduces regularization, discouraging the model from becoming overly dependent on specific features or pathways and thus improving its generalization ability.[33, 34]

An analysis of the MSE across training epochs revealed that the loss function plateaued after approximately 400 epochs, as illustrated in Fig. 2. Based on this observation, the maximum number of training iterations was fixed at that point. The final model achieved an MSE of 0.61. Training was conducted over a period of roughly three hours using Graphics Processing Unit (GPU) resources, with the final model comprising approximately 140,000 trainable parameters. The training process utilized the designated training subset, while the validation dataset supported hyperparameter tuning. Upon completion of training, the model was evaluated on a separate test set, and a detailed performance analysis was conducted.

Given the complexity of the project, establishing a modular

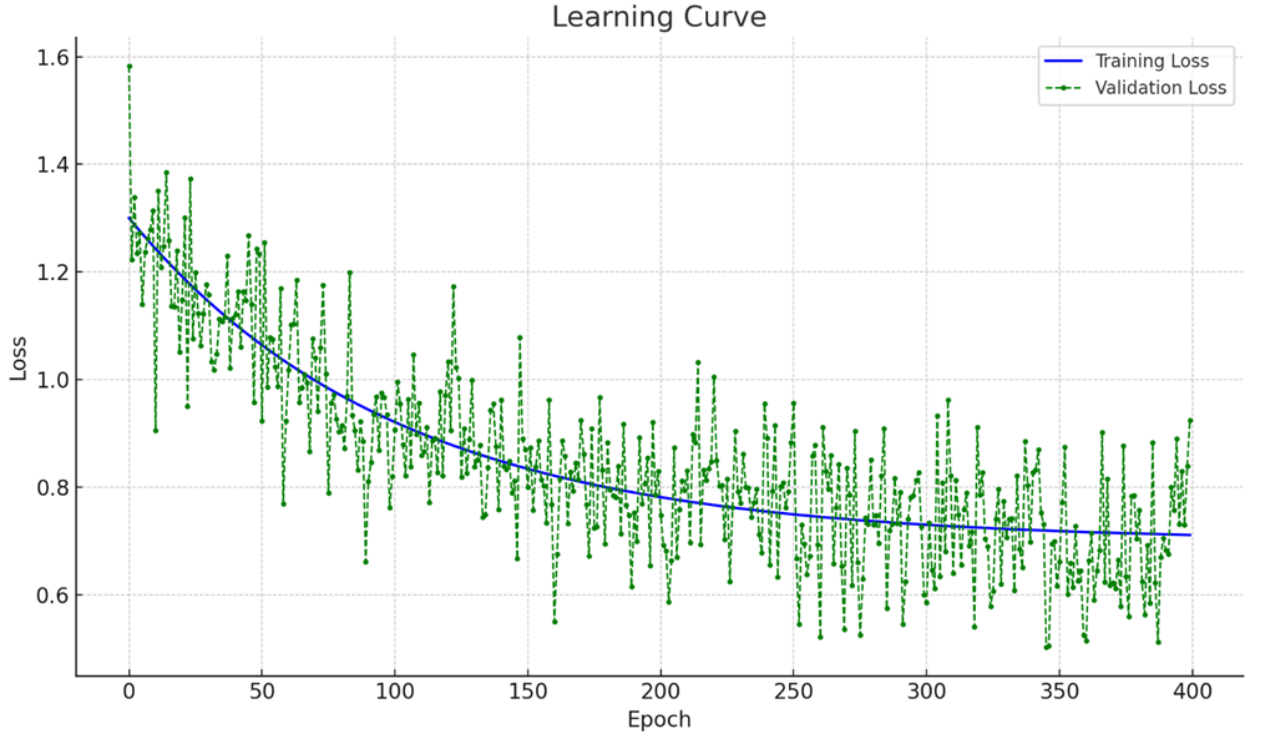

Fig. 2: Training convergence curve showing MSE loss vs epochs.

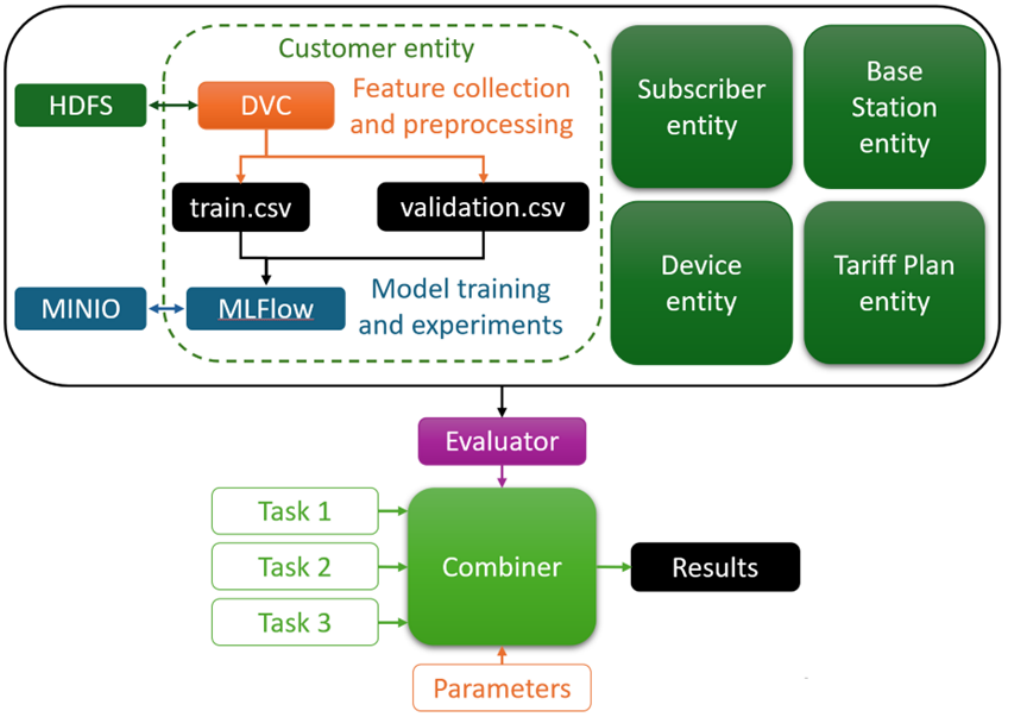

Fig. 3: Modular system design for embedding-based look-alike model.

architecture for model components was a key consideration A comprehensive interaction scheme was developed to organize feature extraction, model training, and evaluation. Fig. 3 illustrates this architecture, which is centered on independent feature gathering per entity. The Hadoop Distributed File System (HDFS), a core element of the Hadoop ecosystem, served as the central repository, ensuring distributed and fault-tolerant data storage.[35] Feature extraction was implemented via Python scripts utilizing PySpark, enabling efficient large-scale data processing across the cluster.

To ensure reproducibility and version control, a Git repository integrated with Data Version Control (DVC) was employed. DVC tracked dataset versions and training artifacts generated by Python pipelines.[36] Training parameters and results were logged using MLFlow in combination with MinIO, providing detailed experiment management and traceability.[37] To support look-alike modeling, a dedicated module was created to concatenate embedding vectors from all entities. This module generated a unified representation of each subscriber by integrating embeddings from multiple domains - such as user behavior, device characteristics, network usage, and tariff data. The resulting composite vector served as the basis for computing pairwise similarities between subscribers.[38]

An evaluation component was also developed to identify similar subscribers based on cosine similarity between these unified embeddings. This approach enabled the detection of users with comparable characteristics in the overall subscriber database, facilitating the construction of robust look-alike models for targeted applications. [39]

5. Mathematical model

To ensure formal clarity and reproducibility, this section presents the mathematical formulation of the autoencoder architecture and the similarity evaluation process used in proposed decision.

5.1 Autoencoder structure

An autoencoder consists of two components: an encoder and a decoder.

Consider the following notation**:**

- $x \in \mathbb{R}^{n}$ – input feature vector
- $z \in \mathbb{R}^{k}$ – latent (compressed) representation
- $\hat{x} \in \mathbb{R}^{n}$ – reconstructed output

The encoder maps the input to a latent space:

$$z = \sigma(W_e x + b_e) \tag{1}$$

This Eq. (1) defines how the encoder transforms the input x into a latent representation z, using weights W_e, bias b_e, and a non-linear activation function σ.

The decoder reconstructs the original input from the latent representation Eq. (2):

$$\hat{x} = \sigma(W_d z + b_d) \tag{2}$$

here:

- W_e, W_d – weight matrices,
- b_e, b_d – bias vectors,
- σ – activation function (Leaky ReLU in hidden layers).

The decoder maps the latent vector z back to the original feature space to approximate the input x, thus enabling reconstruction learning.

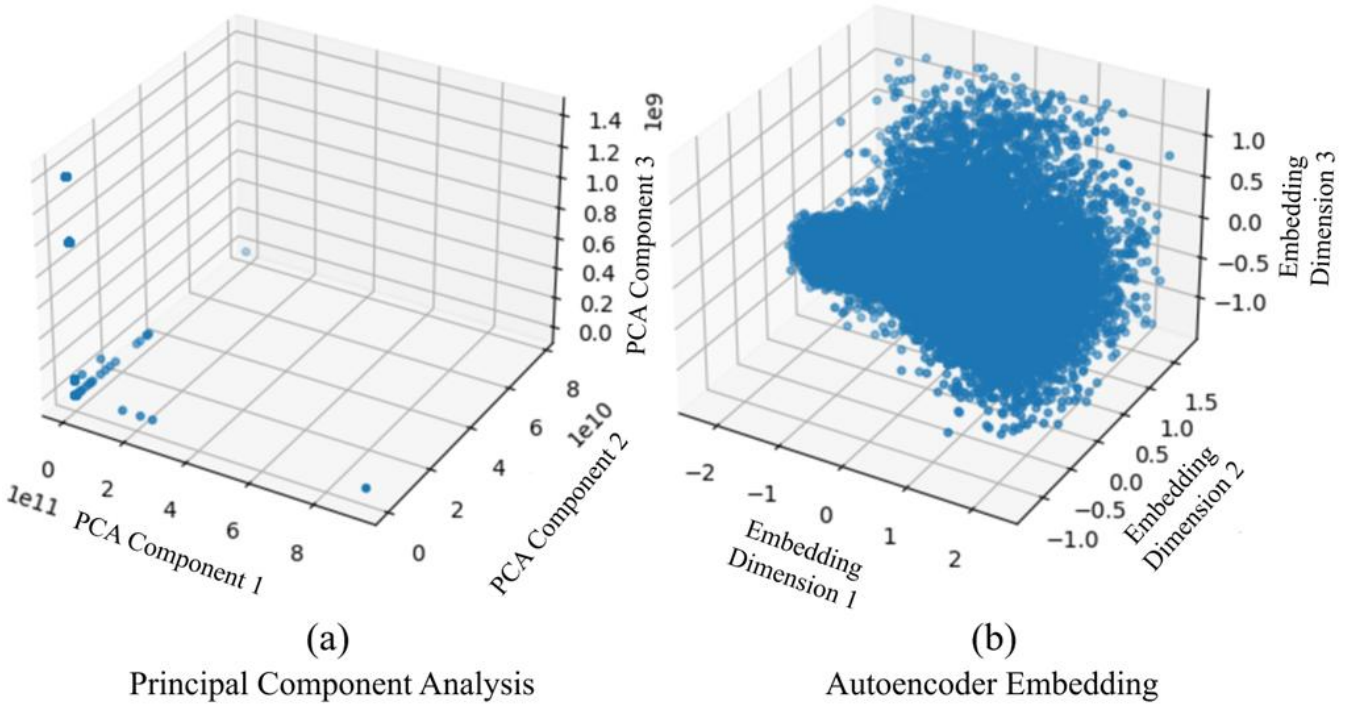

Fig. 4: Comparison of dimensionality reduction techniques applied to the dataset: (a) PCA projection onto the first three principal components, showing sparse variance concentration; (b) 3 Dimensional (3D) embedding obtained using an autoencoder, capturing dense and structured data distribution.

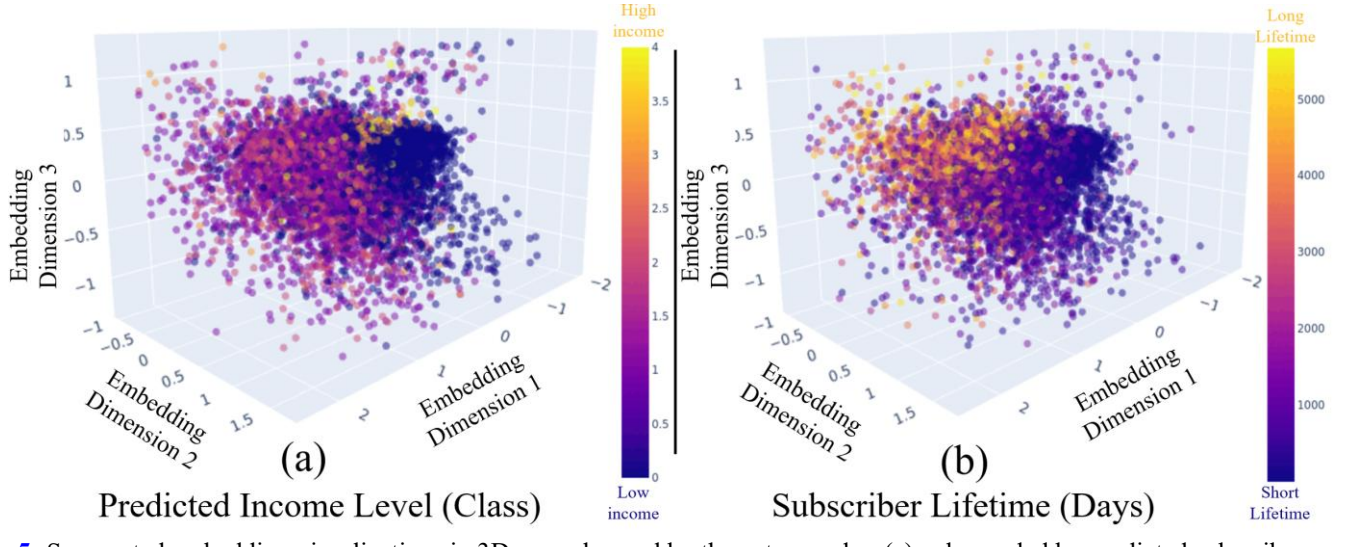

Fig. 5: Segmented embedding visualizations in 3D space learned by the autoencoder: (a) color-coded by predicted subscriber income level class, and (b) by actual subscriber lifetime in days.

5.2 Loss function

The autoencoder is trained to minimize reconstruction loss using MSE Eq. (3):

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left\|x^{(i)} - \hat{x}^{(i)}\right\|^2 \tag{3}$$

This loss function penalizes deviations between the original input and the reconstructed output, encouraging the autoencoder to learn compact and informative representations. This ensures that the model accurately reconstructs the input from its compressed representation.

5.3 Regularization and optimization

To improve generalization and prevent overfitting, several regularization techniques are applied Eq. (4):

Dropout randomly disables a fraction p of neurons during training:

$$\tilde{h}_i = d_i \cdot h_i, \quad d_i \sim Bernoulli(1-p) \tag{4}$$

Dropout improves robustness by preventing the network from relying on specific neurons.

Batch Normalization stabilizes and accelerates training Eq. (5):

$$\mathrm{BN}(x_i) = \gamma \cdot \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \quad (5)$$

This normalizes the input to each layer, reducing internal covariate shift and improving training dynamics.

Ridge regularization (L2 Regularization) is applied to avoid large weights Eq. (6):

$$\mathcal{L}_{total} = \mathcal{L}_{MSE} + \lambda(\|W_e\|_2^2 + \|W_d\|_2^2) \quad (6)$$

This term adds a penalty proportional to the squared magnitude of the model weights, helping to reduce overfitting.

Optimization is performed using the Adam algorithm Eq. (7):

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon} \quad (7)$$

Where $\widehat{m_t}, \widehat{v_t}$ are moment estimates, and α is the learning rate. Adam is an adaptive optimizer that uses moving averages of gradients and squared gradients for efficient parameter updates.

5.4 Similarity measure

After training, embeddings z are used to compare users. Similarity is measured using cosine similarity Eq. (8):

$$\mathrm{cosine\_sim}(z_1, z_2) = \frac{z_1 \cdot z_2}{\|z_1\|\|z_2\|} \quad (8)$$

This metric quantifies the angular similarity between two users in the latent space, which is robust to scale differences.

A similarity threshold θ is chosen using the Precision-Recall curve Eq. (9). If:

$$\mathrm{cosine\_sim}(z_{target}, z_{candidate}) \geq \theta \quad (9)$$

then the candidate is considered similar to the target. This rule defines how the system determines whether a candidate user shares similar behavior or characteristics with a known target group.

5.5 Multi-entity embedding

Final embeddings are constructed by concatenating compressed representations from multiple domains Eq. (10):

$$z_{\mathrm{concat}} = [z_{\mathrm{sub}}, z_{\mathrm{dev}}, z_{\mathrm{bs}}, z_{\mathrm{tar}}] \quad (10)$$

where:

- z_{sub} – subscriber embedding
- z_{dev} – device embedding
- z_{bs} – base station embedding
- z_{tar} – tariff plan embedding

Each component encodes a different entity: subscriber behavior (z_{sub}), device information (z_{dev}), base station context (z_{bs}), and tariff plan features (z_{tar}). The concatenated vector represents a holistic user profile for downstream tasks. This unified vector forms a full subscriber profile and enables efficient search for look-alike users.

6. Results and discussion

To preliminarily assess the capability of autoencoders to generate compact and informative embedding vectors, a comparative analysis was conducted against PCA, a well-established linear dimensionality reduction technique, as shown in Fig. 4. While PCA effectively compresses data by projecting it onto orthogonal components, it lacks the capacity to capture nonlinear dependencies inherent in complex datasets.[40] Upon reducing the data dimensionality to a three-dimensional space and visualizing the resulting representations, it was observed that autoencoders produced a denser and more structured embedding space, forming stable and distinguishable clusters. In contrast, the embeddings generated by PCA appeared more dispersed and exhibited less coherent structure.

To further evaluate the hypothesis and explore its applicability in real-world scenarios, an additional analysis was conducted to determine whether the positioning of data points within the n-dimensional compressed embedding space reflects meaningful physical properties.[41] The objective was to examine whether the embeddings facilitate natural clustering, whereby similar observations are located in proximity, and dissimilar ones are spatially separated.[42] For this purpose, the data were projected into a three-dimensional space and visualized based on specific segmentation criteria: subscriber lifetime (*i.e.*, the number of days a user has been present in the system) and predicted income level. Fig. 5 illustrates these visualizations within a dense, well-structured embedding space. The left-hand plot demonstrates that subscribers with longer lifetimes tend to be concentrated in the yellow-colored regions, while newer users are predominantly represented in purple. The right-hand plot shows a similar separation based on profitability, where high-income subscribers are visualized in yellow and low-income ones in purple, indicating that the learned embeddings successfully capture meaningful behavioral and economic distinctions among users.

An additional visualization was performed based on the Subscriber Identity Module (SIM) card configuration of subscriber devices. As shown on the left side of Fig. 6, distinct clustering patterns emerge depending on the SIM setup, which includes the following categories: devices without SIM cards (*e.g.*, Wireless Fidelity (Wi-Fi)–only devices), devices with two Embedded Subscriber Identity Modules (eSIMs), devices with one physical SIM and one eSIM, devices with a single physical SIM, and devices with two physical SIMs. These groupings suggest meaningful behavioral or device-type patterns associated with different connectivity configurations. The right-hand side of the figure illustrates the positioning of Apple devices within the multidimensional embedding space. The Android group exhibits a wider spread across the embedding space, which can be attributed to the greater variability in device manufacturers, hardware specifications, and price segments. Notably, devices that closely resemble Apple products are located in close proximity to one another.

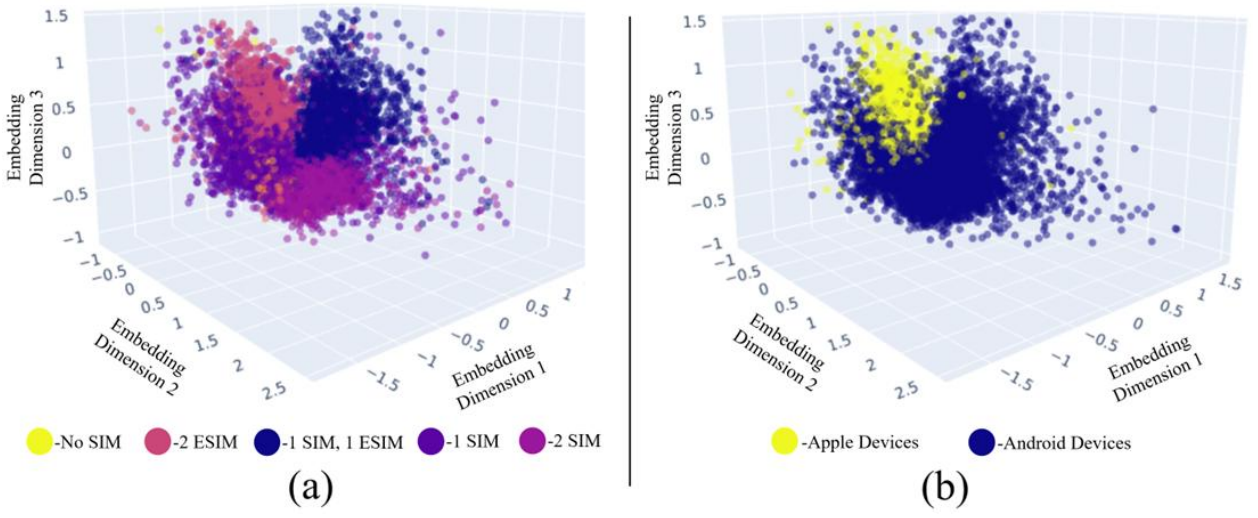

Fig. 6: Segmented embedding visualizations in 3D space: (a) colored by SIM configuration category (nosim, 2esim, 1sim1esim, 1sim, 2sim), and (b) colored by Apple device indicator (yellow = Apple, blue = other).

Table 1: Model performance comparison.

Model	CR	ROC AUC	Lift top 1	Precision	Recall
SVM no entities embeddings[1]	0.13	0.64	4.9	0.54	0.55
Random Forest no entities embeddings[1]	0.15	0.66	5.4	0.60	0.54
LGBM no entities embeddings[1]	0.19	0.69	6.6	0.64	0.56
Cosine similarity with subscriber entity embedding	0.21	0.70	7.3	0.67	0.61
Cosine similarity with all concatenated entities embeddings	0.31	0.76	11.7	0.73	0.70

These observations suggest that the embedding vectors encode meaningful physical properties that reflect similarities in device characteristics.

Following model training on the designated training dataset, evaluation was conducted using a separate test set. As previously noted, cosine distance served as the similarity metric between vector representations. To determine whether a given subscriber is similar to users who performed the target action, a decision threshold was introduced. If the cosine distance between the subscriber's vector and the reference vector corresponding to the target action falls below this threshold, the subscriber is classified as similar to the target group. [43, 44] To comprehensively evaluate model performance, several metrics were employed, including lift, precision, recall, and F1-score. Of particular importance was the Lift Top 1 metric, which quantifies the model's effectiveness in identifying high-probability cases relative to a baseline model, focusing on the top 1% of predictions ranked by score. This metric provides insights into the model's discriminative power in prioritizing the most relevant instances.[45]

In addition to these technical metrics, a business-oriented evaluation was performed using the conversion rate, which measures the extent to which the model's predictions result in desired outcomes—such as user engagement or successful response to targeted actions.[46]

The precision-recall (PR) curve was also utilized to determine the optimal threshold for cosine similarity, enabling a balanced trade-off between precision and recall and enhancing the practical utility of the model.[47] While the embedding-based similarity approach (using cosine distance) differs fundamentally from supervised classification models such as Support Vector Machine (SVM), Random Forest, and Light Gradient Boosting Machine (LGBM), these approaches are compared in Table 1 to benchmark effectiveness across common paradigms used in user targeting. This comparison aims to evaluate whether learned representations and unsupervised similarity-based retrieval can outperform classical supervised learning in practical business applications.

A baseline cosine similarity using raw, non-encoded feature vectors was not included, as such input lacks a meaningful vector geometry suitable for cosine distance computations. Raw features in the original space are often heterogeneous and unnormalized, making cosine similarity ineffective without prior representation learning. The autoencoder, therefore, plays a critical role in transforming the input data into a structured latent space where cosine-based comparisons become meaningful and robust. According to the summarized probing results, the use of autoencoders in this task led to a notable increase in computational efficiency while also enabling the extraction of latent nonlinear patterns in the data. This, in turn, contributed to performance improvements across all evaluated metrics.[48]

7. Discussion

The experimental results demonstrate the effectiveness of autoencoders in capturing complex, nonlinear structures in the data that are not discernible through linear techniques such as PCA. The visualizations of the learned embeddings across

different segmentation criteria—such as subscriber lifetime, income levels, SIM configurations, and device types—provide strong evidence that autoencoders can extract meaningful latent representations that align with real-world behavioral and technical attributes. Compared to traditional classification models (*e.g.*, SVM, Random Forest, and LGBM), the unsupervised embedding-based approach using cosine similarity shows a clear advantage in terms of Lift Top 1 and conversion rate. These improvements highlight the utility of representation learning when labeled data are sparse or when the definition of a target class is ambiguous, as is often the case in look-alike modeling tasks.

However, there are some limitations that warrant consideration. First, the performance of the model is highly dependent on the quality and representativeness of the input data. Incomplete or noisy features may negatively affect the quality of the learned embeddings. Second, although the use of autoencoders improves computational efficiency compared to high-dimensional raw data, training neural architectures still involves significant overhead and tuning, which can be a constraint in real-time production environments. Additionally, the method's generalizability to domains beyond telecommunications has yet to be fully tested and may require domain-specific modifications to input features and model architecture. From a practical perspective, the findings suggest that the proposed approach can be beneficial in a variety of user-targeting and behavioral prediction scenarios. Nevertheless, future research should consider evaluating alternative similarity metrics beyond cosine distance, exploring hybrid approaches that combine supervised and unsupervised learning, and incorporating more comprehensive sets of features—potentially including temporal patterns or graph-based relationships.

Finally, extending the architecture to jointly encode all entities in a unified latent space, rather than using multiple separate autoencoders, may yield richer and more globally coherent representations. Such improvements could further enhance the performance and interpretability of the embedding-based retrieval and prediction systems.

8. Potential applications in infrastructure and resource monitoring

In future research, the methodology presented in this paper may be extended to the domain of industrial process optimization and material performance prediction. Specifically, autoencoder-based representations can be applied to predict equipment wear, identify failure patterns, and optimize composite materials used for hydrotechnical structures in raw conditions such as rivers with high-speed flow. Several recent studies[49-53] have demonstrated the growing relevance of machine learning in the development of wear-resistant materials, surface treatments, and efficiency assessment of construction equipment. By transforming high-dimensional sensor and operational data into compressed embeddings, the proposed approach could enable predictive diagnostics and real-time control in manufacturing and construction industries.

Moreover, the efficient use of water resources for agrotechnical needs remains a critical concern in many countries and regions. Challenges arise not only in water discharge control but also in the monitoring process itself. Traditional measurement methods demonstrate low efficiency and require installation in specific conditions, such as laminar flow zones. Furthermore, conventional instruments are often incapable of tracking actual water consumption by end users. These limitations highlight the need for machine learning methods, which can offer advanced evaluation and monitoring capabilities. Such tools enable the analysis of water usage through aerospace imagery or structured tabular datasets, as discussed in previous studies.[54]

9. Conclusion

In this study, autoencoders were utilized to address the challenge of insufficient negative examples in datasets used for constructing look-alike models within the telecommunications sector. The main objective was to predict subscriber interest in advertisements by leveraging historical data and behavioral features.

Embedding vectors were generated for distinct entities, including subscribers, mobile devices, base stations, and tariff plans. Autoencoders were trained on these entity-specific datasets, and cosine similarity was used to measure the distance between embedding vectors. To assess whether a subscriber resembled those who had previously engaged in the target behavior, a similarity threshold was defined. The resulting model demonstrated satisfactory performance across standard quality metrics.

The results indicate the broad applicability of the proposed approach, supporting its use across multiple domains. In healthcare, for instance, this methodology could facilitate the compression of extensive diagnostic data into structured embeddings for predictive modeling. In the financial sector, it may be applied to detect transaction patterns, while in environmental science, it could support the analysis of climate-related tabular data. The ability to identify latent nonlinear dependencies with minimal information loss underscores the effectiveness of this technique in handling complex and high-dimensional tabular datasets. This adaptability enhances its potential value across domains that rely on structured data analysis.

Beyond classification tasks, the approach can also be employed for anomaly detection, such as identifying irregularities in subscriber behavior, which may contribute to faster resolution of service-related issues and improved service quality. Moreover, the method can serve as a foundation for recommendation systems built on behavioral embeddings, thereby improving personalization and user targeting.

Future research will focus on exploring alternative similarity metrics for comparing embeddings produced by

autoencoders. Additionally, the set of input features will be expanded through the integration of new entities representing additional dimensions of user behavior and context. A comparative analysis will also be conducted between two modeling strategies: the use of multiple independent autoencoders trained on separate entities versus a unified autoencoder architecture that ingests all features jointly. Moreover, the suggested method could be practically applied in classifications of rivers and water reservoirs for different categories of drought indicators. In addition, the solution can be used to improve the existing water resource monitoring modules and also eliminate the disadvantages by applying them for tabular data processing.

Acknowledgments
This research was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan, grant number BR24993128 "Information-analytical system development for the transboundary water resources effective use in the Zhambyl region agricultural sector."

Conflict of Interest
There is no conflict of interest.

Supporting Information
Not applicable.

CRediT Statement
Il'murat Tokhtakhunov: Conceptualization, Data curation, Methodology, Software, Formal analysis, Validation, Visualization, Resources, Writing – Original draft. **Marat Nurtas:** Supervision, Project administration, Validation, Funding acquisition, Resources, Writing – Review & Editing. **Alexander Neftissov, Sharofiddin Pirnaev, Ilyas Kazambayev and Lalita Kirichenko:** Funding acquisition, Resources, Writing – Review & Editing.

References
[1] A. Altaibek, I. Tokhtakhunov, M. Nurtas, D. Kozhamzharova, M. Aitimov, The efficacy of autoencoders in the utilization of tabular data for classification tasks, *Procedia Computer Science*, 2024, **238**, 492-502, doi: 10.1016/j.procs.2024.06.052.
[2] A. Sannigrahi, R. Walambe, K. Kotecha, Multi-head variational graph autoencoder framework for link prediction on citation graphs, *Engineered Science*, 2025, **34**, doi: 10.30919/es1406.
[3] K. Saranya, A. Valarmathi, A multilayer deep autoencoder approach for cross layer IoT attack detection using deep learning algorithms, *Scientific Reports*, 2025, **15**, 10246, doi: 10.1038/s41598-025-93473-9.
[4] A. M. Al-Hinawi, R. A. Alelaimat, E. Alhenawi, M. I. AlBiajawi, Hybrid deep learning approach for accurate prediction of flowability in ultra-high-performance concrete, *Engineered Science*, 2024, **30**, doi: 10.30919/es1182.
[5] W. Zhang, S. E. Barykin, T. V. Kirillova, I. V. Kapustina, N. S. Lukashevich, A. Zaytsev, An innovative resource management framework using logit-boosted machine learning algorithms for vehicular ad hoc networks (VANETs), *Engineered Science*, 2023, **26**, doi: 10.30919/es980.
[6] D.-K. Kim, D. Ryu, Y. Lee, D.-H. Choi, Generative models for tabular data: a review, *Journal of Mechanical Science and Technology*, 2024, **38**, 4989-5005, doi: 10.1007/s12206-024-0835-0.
[7] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, G. Kasneci, Deep neural networks and tabular data: a survey, *IEEE Transactions on Neural Networks and Learning Systems*, 2024, **35**, 7499-7519.
[8] P. Geetha, C. Naikodi, L. Suresh, Optimized deep learning for enhanced trade-off in differentially private learning, *Engineering, Technology & Applied Science Research*, 2021, **11**, 6745-6751, doi: 10.48084/etasr.4017.
[9] A. M. Karim, H. Kaya, M. S. Güzel, M. R. Tolun, F. V. Çelebi, A. Mishra, A novel framework using deep auto-encoders based linear model for data classification, *Sensors*, 2020, **20**, 6378, doi: 10.3390/s20216378.
[10] A. Paricio-Garcia, M. A. Lopez-Carmona, S. Sierra-Arquero, P. Manglano-Redondo, Analysis and evaluation of autoencoder-driven dimensionality reduction for face recognition pipelines, *Applied Soft Computing*, 2025, **172**, 112877, doi: 10.1016/j.asoc.2025.112877.
[11] M. Nurtas, Z. Zhantaev, A. Altaibek, S. Nurakynov, N. Mekebayev, K. Shiyapov, B. Iskakov, A. Ydyrys, Predicting the likelihood of an earthquake by leveraging volumetric statistical data through machine learning techniques, *Engineered Science*, 2023, **26**, doi: 10.30919/es1031.
[12] S. Abrar, M. D. Samad, Perturbation of deep autoencoder weights for model compression and classification of tabular data, *Neural Networks*, 2022, **156**, 160-169, doi: 10.1016/j.neunet.2022.09.020.
[13] T. O. Hodson, Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not, *Geoscientific Model Development*, 2022, **15**, 5481-5487, doi: 10.5194/gmd-15-5481-2022.
[14] M. Nurtas, F. Tokmukhamedova, A. Ydyrys, Z. Zhantaev, S. Nurakynov, B. Iskakov, A. Altaibek, B. Matkerim, Application of finite element method for solving seismoacoustic modeling problems in poroelastic composite media, *Engineered Science*, 2023, **26**, doi: 10.30919/es1030.
[15] H. Torabi, S. L. Mirtaheri, S. Greco, Practical autoencoder based anomaly detection by using vector reconstruction error, *Cybersecurity*, 2023, **6**, 1, doi: 10.1186/s42400-022-00134-9.
[16] A. I. Abueid, Big data and cloud computing opportunities and application areas, *Engineering, Technology & Applied Science Research*, 2024, **14**, 14509-14516, doi: 10.48084/etasr.7339.
[17] H. S. Lom, A. C. Thoo, W. M. Lim, K. Y. Koay, Advertising value and privacy concerns in mobile advertising: the case of SMS advertising in banking, *Journal of Financial Services Marketing*, 2024, **29**, 1135-1153, doi: 10.1057/s41264-023-

00263-3.
[18] G.-W. Cha, W.-H. Hong, Y.-C. Kim, Performance improvement of machine learning model using autoencoder to predict demolition waste generation rate, *Sustainability*, 2023, **15**, 3691, doi: 10.3390/su15043691.
[19] A. Kairanbayeva, Z. Zhantayev, S. Nurakynov, M. Nurtas, G. Nurpeissova, D. Panyukova, C. A. Kazakhstan, A. Mitkov, D. Talgarbayeva, M. Kudaibergenov, Predictive system for road condition monitoring based on open climate and remote sensing data–a case study with mountain roads, *Engineered Science*, 2024, **28**, doi: 10.30919/es1081.
[20] M. Garouani, A. Ahmad, M. Bouneffa, M. Hamlich, Autoencoder-kNN meta-model based data characterization approach for an automated selection of AI algorithms, *Journal of Big Data*, 2023, **10**, 14, doi: 10.1186/s40537-023-00687-7.
[21] N. Capuano, M. Meyer, F. D. Nota, Analyzing the impact of conversation structure on predicting persuasive comments online, *Journal of Ambient Intelligence and Humanized Computing*, 2024, **15**, 3719-3732, doi: 10.1007/s12652-024-04841-8.
[22] T. P. Rinjeni, A. Indriawan, N. A. Rakhmawati, Matching scientific article titles using cosine similarity and jaccard similarity algorithm, *Procedia Computer Science*, 2024, **234**, 553-560, doi: 10.1016/j.procs.2024.03.039.
[23] A. J. Chemmanam, B. Jose, A. Moopan, Improved multi object tracking with locality sensitive hashing, *Pattern Analysis and Applications*, 2024, **27**, 136, doi: 10.1007/s10044-024-01353-1.
[24] M. Nurtas, Z. Zhantaev, A. Altaibek, Earthquake time-series forecast in Kazakhstan territory: forecasting accuracy with SARIMAX, *Procedia Computer Science*, 2024, **231**, 353-358, doi: 10.1016/j.procs.2023.12.216.
[25] M. T. Islam, K. M. Hasib, M. M. Rahman, A. N. Tusher, M. S. Alam, M. R. Islam, Convolutional auto-encoder and independent component analysis based automatic place recognition for moving robot in invariant season condition, *Human-Centric Intelligent Systems*, 2023, **3**, 13-24, doi: 10.1007/s44230-022-00013-z.
[26] S. Bunian, M. A. Al-Ebrahim, A. A. Nour, Role and applications of artificial intelligence and machine learning in manufacturing engineering: a review, *Engineered Science*, 2024, **29**, doi: 10.30919/es1088.
[27] P. Lavanya, I. V. Subba Reddy, V. Selvakumar, Long range radio technology implementation on Internet of Things to detect particulate matter at the community level and prediction using machine learning based approach, *Engineered Science*, 2024, **29**, doi: 10.30919/es1119.
[28] O. Rainio, J. Teuho, R. Klén, Evaluation metrics and statistical tests for machine learning, *Scientific Reports*, 2024, **14**, 6086, doi: 10.1038/s41598-024-56706-x.
[29] K. Berahmand, F. Daneshfar, E. S. Salehi, Y. Li, Y. Xu, Autoencoders and their applications in machine learning: a survey, *Artificial Intelligence Review*, 2024, **57**, 28, doi: 10.1007/s10462-023-10662-6.
[30] D. V. Dong, Application of advanced deep convolutional neural networks for the recognition of road surface anomalies, *Engineering, Technology & Applied Science Research*, 2023, **13**, 10765-10768, doi: 10.48084/etasr.5890.
[31] S. Pooja, L. K. Raju, U. Chhapekar, B. Chandrakala C, Face detection using deep learning to ensure a coercion resistant blockchain-based electronic voting, *Engineered Science*, 2021, **16**, doi: 10.30919/es8d585.
[32] Z. Liu, Y. Liu, Z. Yu, Z. Yang, Q. Fu, Y. Guo, Q. Liu, G. Wang, PT-VAE: Variational autoencoder with prior concept transformation, *Neurocomputing*, 2025, **638**, 130129, doi: 10.1016/j.neucom.2025.130129.
[33] F. Yang, Q. Xu, F. Chen, Active learning deep autoencoder model with importance sampling for reliability analysis, *Applied Soft Computing*, 2025, **180**, 113428, doi: 10.1016/j.asoc.2025.113428.
[34] Y. D. Al-Otaibi, Enhancing e-commerce strategies: a deep learning framework for customer behavior prediction, *Engineering, Technology & Applied Science Research*, 2024, **14**, 15656-15664, doi: 10.48084/etasr.7945.
[35] Y. Zhang, J. Menke, J. He, E. Nittinger, C. Tyrchan, O. Koch, H. Zhao, Similarity-based pairing improves efficiency of Siamese neural networks for regression tasks and uncertainty quantification, *Journal of Cheminformatics*, 2023, **15**, 75, doi: 10.1186/s13321-023-00744-6.
[36] A. Fedele, R. Guidotti, D. Pedreschi, Explaining Siamese networks in few-shot learning, *Machine Learning*, 2024, **113**, 7723-7760, doi: 10.1007/s10994-024-06529-8.
[37] R. Gustriansyah, J. Alie, N. Suhandi, A hybrid machine learning model for market clustering, *Engineering, Technology & Applied Science Research*, 2024, **14**, 18824-18828, doi: 10.48084/etasr.9259.
[38] Q. Yu, Q. Chang, T. Ouyang, T. Kurita, R. Dong, Direction-aware convolutional autoencoder based on positional encoding for one-dimensional anomaly detection, *Information Sciences*, 2025, **716**, 122227, doi: 10.1016/j.ins.2025.122227.
[39] I. Tokhtakhunov, A. Altaibek, M. Nurtas, Optimizing similar audience search in targeted advertising: effectiveness of Siamese networks for autoencoder-based user embeddings, *Engineering, Technology & Applied Science Research*, 2025, **15**, 23367-23375, doi: 10.48084/etasr.10527.
[40] Z. Hosseini, S. Mohammadi, H. Safari, An assessment of the impact of information technology on marketing and advertising, *Engineering, Technology & Applied Science Research*, 2018, **8**, 2526-2531, doi: 10.48084/etasr.1620.
[41] N. Serrano, A. Bellogín, Siamese neural networks in recommendation, *Neural Computing and Applications*, 2023, **35**, 13941-13953, doi: 10.1007/s00521-023-08610-0.
[42] P. Ren, Y. Wang, Z. Wang, D. Peng, C. Liu, T. Han, Denoising autoencoder multilayer perceptron spiking neural network for isonicotinic acid yield prediction on real industrial dataset, *Advanced Engineering Informatics*, 2025, **65**, 103273, doi: 10.1016/j.aei.2025.103273.
[43] Z. Hu, Z. Xiao, H. Sun, H. Yang, Autoencoder evolutionary algorithm for large-scale multi-objective optimization problem, *International Journal of Machine Learning and Cybernetics*, 2024, **15**, 5159-5172, doi: 10.1007/s13042-024-02221-4.

[44] Q.-T. Tran, An application of neural network-based sliding mode control for multilevel inverters, *Engineering, Technology & Applied Science Research*, 2024, **14**, 12530-12535, doi: 10.48084/etasr.6516.
[45] Ł. Delong, A. Kozak, The use of autoencoders for training neural networks with mixed categorical and numerical features, *ASTIN Bulletin*, 2023, **53**, 213-232, doi: 10.1017/asb.2023.15.
[46] K. M. Ghori, M. Imran, A. Nawaz, R. A. Abbasi, A. Ullah, L. Szathmary, Performance analysis of machine learning classifiers for non-technical loss detection, *Journal of Ambient Intelligence and Humanized Computing*, 2023, **14**, 15327-15342, doi: 10.1007/s12652-019-01649-9.
[47] W. Lee, S. Lee, H. Kim, J. Lee, Sliced Wasserstein adversarial training for improving adversarial robustness, *Journal of Ambient Intelligence and Humanized Computing*, 2024, **15**, 3229-3242, doi: 10.1007/s12652-024-04791-1.
[48] S. Merugu, R. Yadav, V. Pathi, H. R. Perianayagam, Identification and improvement of image similarity using autoencoder, *Engineering, Technology & Applied Science Research*, 2024, **14**, 15541-15546, doi: 10.48084/etasr.7548.
[49] T. Askarxodjayev, S. Pirnaev, F. Dzhumabaeva, G. Yangiboev, A. Idiev, Development of wear-resistant material for strengthening, *Asia-Pacific Conference on Applied Mathematics and Statistics*, Chiangmai, Thailand, AIP Publishing, 2022, 030098, doi: 10.1063/5.0091543.
[50] S. Pirnaev, R. Sindarov, F. Dzhumabeva, S. Saidova, Technique for experimental studies of asphalt concrete milling process, *E3S Web of Conferences*, 2021, **264**, 02016, doi: 10.1051/e3sconf/202126402016.
[51] S. Pirnayev, A. Mukhitdinov, D. Saydaliyeva, G. Tulaboyeva, Economic effectiveness of choosing and based on the method of spraying the plasma coating to road milling cutters, *Problems in the Textile and Light Industry in the Context of Integration of Science and Industry and Ways to Solve Them: PTLICISIWS-2*, Namangan, Uzbekistan, AIP Conference Proceedings, 2024, **3045**(1), 060034, doi: 10.1063/5.0197334.
[52] O. Rabat, S. Pirnaev, D. Saydaliyeva, S. Tulaboyeva, K. Rustamov, Strengthening the Protective Layer of Road Milling Machines Using a Combined Method, *15th International Conference on Thermal Engineering: Theory and Applications*, Tashkent, Uzbekistan, May 28 - June 01, 2024, **1**(1).
[53] O. Rabat, S. Pirnaev, K. Rustamov, I. Usmanov, S. Shermatov, K. Magdiyev, Development of corrosion-resistant material for asphalt concrete cutting part, *E3S Web of Conferences*, 2024, **587**, 03012, doi: 10.1051/e3sconf/202458703012.
[54] A. Mosavi, P. Ozturk, K.-W. Chau, Flood prediction using machine learning models: literature review, *Water*, 2018, **10**(11), 1536, doi: 10.3390/w10111536.