International Information Technology University

UDC: 004.852(043) On manuscript right

TOKHTAKHUNOV IL'MURAT TURDYMAGAMETOVICH

Deep Learning Models and Methods for Finding “Similar Audience” in Targeted Advertising

8D06105 – Data Science

Thesis for the degree of doctor of Philosophy (PhD)

Scientific consultant
Associate Professor, PhD
International Information Technology University
Nurtas M.
Foreign consultant
Research Scientist, PhD
Meta Platforms, Inc.
Zharmagambetov A.

Republic of Kazakhstan
Almaty, 2026

CONTENT

NORMATIVE REFERENCES

This thesis uses references to the following standards:

Instructions for the preparation of a thesis and an abstract, Higher Attestation Commission of the Ministry of Education and Science of the Republic of Kazakhstan dated September 28, 2004 No. 377-3y.

GOST 7.32-2001. Report on research work. Structure and design rules.

GOST 7.1-2003. Bibliographic record. Bibliographic description. General requirements and rules of compilation.

ST RK 34.005-2002. Information Technology. Basic terms and definitions (first edition).

ST RK. 34.015-2002. Information Technology. Set of standards for automated systems. Terms of reference for creating an IS (first edition).

ST RK 34.027-2006. Information Technologies. Classification of software tools (first edition).

ST RK 34.014-2002. Information Technology. A set of standards for automated systems. Automated systems. Terms and definitions.

DESIGNATIONS AND ABBREVIATIONS

AE	Autoencoder
AUC	Area Under the Curve
B2B	Business-to-Business
DVC	Data Version Control
eSIM	Embedded Subscriber Identity Module
F1	Harmonic Mean of Precision and Recall (F1-score)
GPU	Graphics Processing Unit
HDFS	Hadoop Distributed File System
kNN	k-Nearest Neighbors
L2	L2 Regularization (Ridge / weight decay)
LeakyReLU	Leaky Rectified Linear Unit
LGBM	Light Gradient Boosting Machine
MICE	Multiple Imputation by Chained Equations
MSE	Mean Squared Error
PCA	Principal Component Analysis
PR	Precision-Recall
ReLU	Rectified Linear Unit
RF	Random Forest
ROC	Receiver Operating Characteristic
SAE	Stacked Autoencoder
SIM	Subscriber Identity Module
SVM	Support Vector Machine
t-SNE	t-Distributed Stochastic Neighbor Embedding
UMAP	Uniform Manifold Approximation and Projection

DEFINITIONS

Autoencoder – a neural network architecture trained to compress an input into a lower-dimensional latent representation and reconstruct the original input from that representation, thereby learning informative embeddings in an unsupervised manner.

Batch Normalization – a technique that normalizes the activations of a neural network layer over each mini-batch during training, stabilizing the learning process and accelerating convergence.

Cosine Similarity – a metric that measures the cosine of the angle between two vectors in a multidimensional space, assessing their directional similarity rather than magnitude.

Curse of Dimensionality – a set of phenomena arising when working with data in high-dimensional spaces, including increased sparsity, degraded distance metric meaningfulness, and exponentially growing data requirements.

Embedding – a compact, dense vector representation of an object in a lower-dimensional latent space, learned through a neural network or other representation learning method.

Entity – a thematic group of features describing a distinct behavioral or demographic domain within the user dataset, such as device characteristics, tariff plan parameters, or financial activity.

Latent Space – a lower-dimensional abstract feature space learned by an encoder network, in which semantically similar inputs are positioned close to each other.

Lookalike Audience – a group of users identified by a model as behaviorally similar to a reference seed audience, used to expand target audiences in advertising systems.

Manifold Learning – a class of nonlinear dimensionality reduction methods based on the assumption that high-dimensional data lie on or near a lower-dimensional manifold embedded within the original feature space.

Multi-Entity Embedding – a unified user representation formed by training separate autoencoders on distinct feature domains and concatenating the resulting latent vectors.

Siamese Network – a neural network architecture consisting of two or more identical subnetworks sharing weights, designed to learn a similarity function between pairs of inputs.

t-SNE – t-Distributed Stochastic Neighbor Embedding; a nonlinear dimensionality reduction algorithm that preserves local neighborhood relationships between data points for visualization purposes.

INTRODUCTION

General characteristics of the research

This thesis focuses on the investigation and practical implementation of nonlinear dimensionality reduction and representation learning methods for high-dimensional tabular user data, with a specific focus on their application to look-alike audience detection in targeted advertising systems. The study examines classical linear methods, including Principal Component Analysis (PCA), nonlinear manifold learning approaches such as t-distributed Stochastic Neighbor Embedding (t-SNE), and deep learning–based representation learning using autoencoders. A stacked autoencoder framework is proposed that compresses heterogeneous user feature vectors into compact latent embeddings, enabling efficient user similarity analysis without task-specific model retraining. The framework is evaluated on a large-scale anonymized telecommunications dataset comprising approximately 900,000 subscribers and 948 features organized into six distinct behavioral entity domains.

Relevance of the research

The rapid growth of digital platforms and online services has led to the generation of massive volumes of heterogeneous user data. Modern telecommunications and digital advertising systems routinely collect hundreds of attributes per subscriber, encompassing behavioral signals, demographic characteristics, device specifications, service usage patterns, financial transaction histories, and network activity metrics. The ability to analyze such data effectively and identify behavioral similarities between users has become a central requirement for data-driven business systems, including recommendation engines, customer analytics platforms, and targeted advertising technologies [1, 2].

One of the fundamental challenges in working with high-dimensional datasets is the curse of dimensionality - a phenomenon where the increasing number of features makes it progressively more difficult to identify meaningful relationships between observations. As dimensionality grows, data points become increasingly sparse in the feature space, and traditional distance metrics lose their discriminative power. This negatively affects the ability of machine learning algorithms to identify informative structures and ultimately reduces the quality of similarity-based user analysis [3, 4].

The practical context motivating this dissertation is the development of a generalized look-alike audience detection service for a telecommunications company. Traditional approaches train a dedicated binary classifier for each advertising campaign, which requires substantial analyst effort, cannot be easily automated, and does not scale to serve multiple B2B clients simultaneously. Embedding-based representation learning decouples the user modeling phase from the targeting task, enabling a single trained model to serve diverse campaign objectives without retraining [5, 6].

Previous studies have demonstrated interest in autoencoder-based representation learning for tabular data, but their systematic application to heterogeneous telecommunications user data for look-alike audience modeling, combined with multi-entity embedding strategies and comprehensive evaluation against both classical

methods and business metrics, has not been sufficiently studied. This gap predetermines the topic of the present dissertation and its scientific objectives.

Research aim and objectives

The aim is to develop and validate a nonlinear representation learning framework based on autoencoders for scalable look-alike audience detection in targeted advertising systems, demonstrating its superiority over classical linear and manifold learning techniques.

The objectives are:

– To analyze existing dimensionality reduction techniques PCA and t-SNE and evaluate their applicability and limitations for high-dimensional tabular user data;

– To design and train a stacked autoencoder architecture that generates compact and informative latent embeddings from heterogeneous user feature vectors of 948 dimensions;

– To develop a multi-entity embedding strategy that integrates latent representations from six distinct behavioral domains into a unified user representation;

– To evaluate cosine similarity in the latent space as a mechanism for identifying similar users without task-specific retraining;

– To investigate Siamese networks as a learned adaptive similarity function operating on autoencoder embeddings;

– To design a modular production system architecture supporting scalable deployment and reproducible experimentation;

– To experimentally validate the proposed approach using clustering metrics, kNN classification, and real-world look-alike audience campaign results on an anonymized telecommunications dataset.

Object and subject of research

Object of research: high-dimensional tabular datasets describing user behavior in telecommunications and digital advertising systems.

Subject of research: methods and models for nonlinear dimensionality reduction and representation learning applied to user similarity analysis and look-alike audience detection.

Theoretical and methodological framework

The research employs deep learning representation learning techniques based on stacked autoencoders, classical dimensionality reduction methods (PCA), manifold learning methods (t-SNE), and Siamese network architectures. The implementation uses Python with PyTorch, PyTorch Lightning, scikit-learn, Pandas, NumPy, and Apache Spark libraries. Experimental evaluation follows a rigorous protocol using multiple independent validation datasets with external target variables to ensure unbiased performance estimation and eliminate target leakage.

Information base

The study is based on a large-scale anonymized dataset obtained through a collaboration agreement with a telecommunications operator. The dataset comprises records of approximately 900,000 subscribers, with 948 features derived from 2,814 raw attributes after preprocessing. Features are organized into six entity domains: User Entity (behavioral and demographic characteristics), Web Entity (anonymized internet activity and interests), Finance Entity (anonymized transaction patterns and banking

behavior), Device Entity (mobile device specifications and demographics), Cell Base Station Entity (network quality and geoactivity), and Tariff Plan Entity (service plan parameters and pricing). All data were collected during regular service operation, fully anonymized, and handled in accordance with corporate privacy policies and applicable data protection regulations. Validation was performed on 17 independent advertising campaign datasets with external target variables not present during training.

Scientific novelty

A stacked autoencoder architecture with Batch Normalization and LeakyReLU [7] activations is proposed for generating compact 288-dimensional user embeddings from 948-dimensional heterogeneous tabular feature vectors, demonstrating superior clustering quality and downstream classification performance compared to PCA and t-SNE, as confirmed by Silhouette Score (0.48 vs 0.21), Davies–Bouldin Index (0.79 vs 1.84), and kNN F1 (0.71 vs 0.56).

A unified embedding strategy is introduced, where features from multiple behavioral domains are concatenated into a single representation and processed through a stacked autoencoder [8], producing a comprehensive user profile and yielding a 60% improvement in look-alike detection precision compared to single-entity representations (Lift Top 1: 11.7 vs 7.3).

The application of Siamese networks as an adaptive learned similarity function on autoencoder embeddings is investigated for user similarity measurement in tabular data, demonstrating a 41.6% average improvement over traditional classifiers (SVM, Random Forest [9], LightGBM) across all evaluated look-alike detection metrics.

A modular production-grade system architecture is designed and experimentally validated on a real-world telecommunications dataset of approximately 900,000 subscribers, confirming the practical feasibility of the proposed representation learning framework for industrial deployment.

Scientific results

– A comprehensive comparative analysis of PCA, t-SNE, and autoencoder-based dimensionality reduction was conducted on a large-scale anonymized telecommunications dataset.

– A stacked autoencoder with architecture 948→1000→1000→288→1000→1000→948 was designed, trained, and validated, achieving stable convergence after 400 epochs with validation MSE of 0.61.

– A multi-entity embedding strategy integrating six behavioral domains was developed and experimentally demonstrated to achieve Lift Top 1 = 11.7, ROC AUC = 0.76, and Conversion Rate = 0.31 in look-alike audience detection.

– Siamese networks operating on precomputed autoencoder embeddings achieved Lift Top 1 = 12.9, F1 = 0.75, ROC AUC = 0.79, and Conversion Rate = 0.36, outperforming all baseline methods by an average of 41.6%.

– A modular production system integrating HDFS, DVC, MLflow, and MinIO was designed and validated, supporting both distributed Spark-based and local Pandas-based inference modes.

Key contributions for defense

A stacked autoencoder with the structure 948→1000→1000→288→1000→1000→948 trained with MSE reconstruction loss,

Batch Normalization, and LeakyReLU [7] (α=0.2) activations produces significantly more structured latent representations of high-dimensional tabular user data compared to PCA and t-SNE, confirmed by Silhouette Score (0.48 vs 0.21), Davies–Bouldin Index (0.79 vs 1.84), and kNN F1 (0.71 vs 0.56).

Multi-entity concatenated embedding integrating representations from six behavioral domains improve look-alike audience detection Lift Top 1 from 7.3 (single-entity) to 11.7 (multi-entity), representing a 60% improvement in targeting precision over single-domain representations.

Siamese networks operating on precomputed autoencoder embeddings achieve Lift Top 1 = 12.9, F1 = 0.75, ROC AUC = 0.79, and Conversion Rate = 0.36, outperforming traditional classifiers (SVM, Random Forest [9], LightGBM) by an average of 41.6% across all evaluated metrics.

The proposed modular production architecture, validated on approximately 900,000 anonymized subscriber records with 948 features derived from 2,814 raw attributes, demonstrates computational scalability and deployment feasibility for real-world B2B advertising platforms.

Theoretical and practical significance

This dissertation advances the theoretical understanding of deep representation learning applied to high-dimensional heterogeneous tabular data, establishing a rigorous comparative framework for evaluating dimensionality reduction methods in user similarity analysis. The multi-entity embedding strategy and the integration of Siamese networks with autoencoder representations contribute novel methodological components to the field of representation learning for tabular data.

Practically, the research proposes a fully automated generalized look-alike service that eliminates the need for campaign-specific model training. The proposed system allows B2B clients to submit any reference audience and receive an expanded similar audience within the subscriber base, processed without analyst involvement. This automation reduces human resource costs, improves targeting accuracy (Conversion Rate from 0.13 to 0.36), and enables simultaneous service for multiple clients. The methodology is applicable to any domain involving large-scale heterogeneous tabular user data, including e-commerce, healthcare analytics, and financial services.

Dissemination and publications

The main findings and scientific contributions of this research were presented and discussed at the Department of Mathematical and Computer Modelling at the International Information Technology University (2022–2026) and at the School of Digital Technologies, Narxoz University (2023–2026).

The main results of the research were published in the following works:

1. **I. Tokhtakhunov**, M. Nurtas, A. Altaibek, D. Kozhamzharova, M. Aitimov. The Efficacy of Autoencoders in the Utilization of Tabular Data for Classification Tasks. Procedia Computer Science, vol. 238, pp. 492–502, 2024. https://doi.org/10.1016/j.procs.2024.06.052. [Scopus Q2, percentile: 62, CiteScore(2025): 3.6, available: https://www.scopus.com/sourceid/19700182801]

2. **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, vol. 15, no. 3, pp. 23367–23375, 2025. https://doi.org/10.48084/etasr.10527. [Scopus Q2, percentile: 56, CiteScore: 2.8(2025), available: https://www.scopus.com/sourceid/21101144516]

3. **I. Tokhtakhunov**, M. Nurtas, A. Neftissov, S. Pirnaev, I. Kazambayev, L. Kirichenko. Exploring Autoencoder-Based Representations for Tabular Data Classification. Engineered Science, vol. 37, 1703, 2025. https://doi.org/10.30919/es1703. [Scopus Q1, percentile: 88, CiteScore(2025): 10.2, available: https://www.scopus.com/sourceid/21101039622]

4. **I. Tokhtakhunov**, M. Nurtas. Nonlinear Dimensionality Reduction for Lookalike Audience Detection: From Manifold Learning to Autoencoder-Based Representations. Journal of Problems in Computer Science and Information Technologies, 4(1). https://doi.org/10.26577/jpcsit4120268 [KOKHBO]

Chapter overview

Chapter 1 provides a comprehensive literature review covering the curse of dimensionality, classical and nonlinear dimensionality reduction methods, autoencoder architectures for representation learning, look-alike audience modeling approaches, and Siamese network methods. The chapter identifies the research gap that this dissertation addresses.

Chapter 2 describes the dataset and the complete preprocessing pipeline, including the entity-based feature organization, categorical encoding, missing value imputation, outlier detection using isolation forests, multicollinearity reduction through pairwise correlation analysis, and min-max feature normalization.

Chapter 3 presents the mathematical formulation of all dimensionality reduction methods investigated in the study, including PCA, t-SNE, and the proposed autoencoder architecture. The multi-entity embedding strategy and cosine similarity computation are also described.

Chapter 4 describes the Siamese network architecture designed for adaptive similarity learning on autoencoder embeddings, including the pair construction procedure, training protocol, and computational trade-off analysis.

Chapter 5 presents the modular production system architecture, covering data storage and versioning infrastructure, the model training and evaluation pipeline, and the two production inference modes supporting large-scale deployment.

Chapter 6 presents all experimental results, including visual and quantitative comparison of dimensionality reduction methods, clustering quality evaluation, look-alike audience detection performance, Siamese network results, and business impact analysis.

The **Discussion** section analyzes the obtained results in the context of existing literature, explains the sources of the observed performance improvements, discusses practical implications, and identifies limitations and directions for future research.

The **Conclusion** summarizes the main findings of the dissertation and formulates practical recommendations.

Structure and volume of the thesis

The dissertation includes an introduction, six main chapters, a discussion, a conclusion, a list of references, and an appendix. The total length of the main text is 98

pages, excluding appendices. The dissertation contains 17 figures, 5 tables, and 97 bibliographic references.

In all publications related to the dissertation, the author played the leading role, including the development of the research concept, data collection, model design, software implementation, data analysis, interpretation of results, and preparation of the manuscripts.

1 THEORETICAL FOUNDATIONS AND PROBLEM FORMULATION

The problem of identifying similar users in large-scale digital systems arises at the intersection of high-dimensional data analysis, representation learning, and similarity modeling. In modern targeted advertising and customer analytics, user behavior is described by complex heterogeneous tabular data that includes numerical activity metrics, categorical attributes, and derived behavioral features. The effective processing of such data requires methods capable of extracting compact, informative, and generalizable representations.

A fundamental challenge in this context is the high dimensionality of user data, which leads to sparsity, redundancy, and degradation of distance-based similarity measures. These issues necessitate the application of dimensionality reduction and representation learning techniques that transform raw feature spaces into lower-dimensional latent representations suitable for similarity estimation.

At the same time, the practical requirements of look-alike audience modeling impose additional constraints on the choice of methods. Industrial systems must support scalable processing of large user populations, provide consistent representations for new data, and enable efficient similarity search across diverse targeting scenarios. These requirements limit the applicability of many classical and non-parametric approaches and motivate the use of parametric deep learning models.

The methodological framework considered in this dissertation combines three key components. First, dimensionality reduction methods are analyzed as a means of overcoming the curse of dimensionality and improving the structure of the feature space. Second, representation learning techniques based on autoencoder architectures are considered for constructing compact latent embeddings of users. Third, similarity learning approaches, including Siamese neural networks, are analyzed as a mechanism for improving similarity estimation in the embedding space.

The structure of this chapter reflects the logical development of the problem formulation. Section 1.1 introduces the curse of dimensionality as the central challenge in high-dimensional tabular data analysis. Sections 1.2 and 1.3 examine classical linear and nonlinear dimensionality reduction methods and their limitations. Section 1.4 presents autoencoder-based representation learning as a suitable approach for handling heterogeneous tabular data. Section 1.5 discusses the problem of look-alike audience modeling and its practical requirements in targeted advertising systems. Section 1.6 analyzes Siamese neural networks for similarity learning. Finally, Section 1.7 formulates the research gap and summarizes the theoretical basis for the proposed approach.

1.1 Curse of Dimensionality in High-Dimensional Tabular Data

The curse of dimensionality refers to a collection of phenomena that arise when analyzing data in high-dimensional spaces, where the number of features is large relative to the number of observations. The term was first introduced by Bellman (1957) in the context of dynamic programming to describe the exponential growth in computational complexity as the number of dimensions increases [3]. In the context of

machine learning and data analysis, the curse of dimensionality manifests in several interconnected ways that collectively undermine the effectiveness of standard algorithms.

The most fundamental manifestation is the phenomenon of distance concentration. As dimensionality increases, the ratio of the maximum to the minimum pairwise distances between data points tends toward one, effectively rendering nearest-neighbor search meaningless. Beyer et al. (1999) provided a formal theoretical analysis of this effect, demonstrating that in sufficiently high dimensions, the concept of a "nearest neighbor" loses discriminative power [10]. For look-alike audience modeling, which relies fundamentally on measuring distances between user representations, this concentration effect directly undermines the quality of similarity-based retrieval unless the feature space is first reduced to a more discriminative lower-dimensional representation.

The second major manifestation is the exponential growth in data requirements. As the number of dimensions increases, the volume of the feature space grows exponentially, meaning that maintaining a given data density requires an exponentially increasing number of samples. In the context of supervised classification, this implies that the amount of labeled training data required to achieve a given model accuracy grows exponentially with dimensionality - a phenomenon known as the sample complexity explosion [11]. In practice, real-world datasets are always finite, meaning that high-dimensional spaces are inevitably sparsely populated [12].

Multicollinearity represents a third challenge associated with high dimensionality in real-world tabular datasets. When features are derived from overlapping behavioral sources, many of them carry redundant information. In telecommunications user datasets, for example, multiple activity metrics - total call duration, call frequency, and ARPU - all capture related aspects of the same underlying behavior. Including highly correlated variables without filtering distorts distance calculations and reduces the discriminative power of the feature space. Addressing multicollinearity through correlation-based feature selection [13], as implemented in this dissertation, is therefore a critical preprocessing step [1].

The initial raw dataset used in this study contained 2,814 features per subscriber, illustrating the scale of the dimensionality challenge in practical telecommunications applications. After preprocessing - including one-hot encoding of categorical variables, missing value imputation, outlier removal, and multicollinearity reduction through pairwise correlation analysis - the dataset was reduced to 948 features. Even at this scale, the sparsity of the feature space makes direct application of distance-based similarity methods ineffective, motivating the use of dimensionality reduction as a prerequisite for look-alike audience modeling [14].

Overfitting represents a closely related consequence of high dimensionality for supervised machine learning models. When the number of features approaches or exceeds the number of training samples, models tend to memorize the training data rather than learning generalizable patterns. Regularization techniques can partially mitigate this effect, but they do not address the fundamental information-theoretic limitation that a finite number of observations cannot reliably characterize a high-dimensional distribution. This limitation motivates the adoption of unsupervised

representation learning - specifically autoencoders - which can discover compact representations from data without requiring labeled examples [9, 10, 15].

From an industrial perspective, the curse of dimensionality is further compounded by the heterogeneous nature of features in real-world user datasets. A telecommunications subscriber dataset contains numerical activity metrics, categorical device attributes, binary behavioral indicators, ordinal demographic scores, and multi-valued interest vectors derived from diverse data collection systems. These heterogeneous data types interact in complex ways that cannot be captured by simple linear projections, requiring nonlinear representation learning methods capable of handling mixed-type feature spaces [16].

1.2 Classical Linear Dimensionality Reduction

Principal Component Analysis (PCA) is the most widely used linear dimensionality reduction technique in machine learning and data analysis. Developed by Pearson (1901) [17] and later formalized by Hotelling (1933) [18], PCA identifies orthogonal directions - principal components - that maximize variance in the original feature space and projects data onto a lower-dimensional subspace defined by the top-k components [19]. The method provides a computationally efficient solution that scales well to large datasets and produces interpretable results through variance explained metrics.

PCA has been widely applied as a preprocessing step in machine learning pipelines, including for user behavior analysis and recommendation systems. Its strengths include computational efficiency, straightforward implementation, interpretability through explained variance analysis, and the guarantee of optimal linear reconstruction under the Frobenius norm. In the context of tabular data, PCA has been used to reduce collinear features before classification, to visualize high-dimensional datasets in two or three dimensions, and to initialize more complex nonlinear models [20].

However, PCA is fundamentally limited to capturing linear relationships between variables. The method identifies directions of maximum linear variance and projects data onto these directions, which means that nonlinear dependencies between features are entirely ignored. In high-dimensional user behavioral data, the underlying structure is rarely linear: the combination of device preference, usage pattern, and geographic activity interacts in complex nonlinear ways that cannot be adequately represented by any linear projection. As a result, PCA tends to produce sparse and poorly separated embeddings when applied to such data [21].

Related classical linear methods - including Linear Discriminant Analysis (LDA), Factor Analysis, and Independent Component Analysis (ICA) [22] - share similar fundamental limitations. LDA requires labeled class information and optimizes class separation rather than general feature compression, making it unsuitable for the generalized look-alike framework where no predefined target variable is available during embedding learning. Factor analysis assumes a specific latent factor structure that may not correspond to the actual data generating process. ICA decomposes data

into statistically independent components but does not provide a straightforward mechanism for dimensionality reduction in similarity-based retrieval tasks [23, 24].

Regarding computational scalability, standard PCA requires computing the covariance matrix of the full feature space, with time complexity $O(nd^2 + d^3)$ where n is the number of samples and d is the number of features. For datasets with thousands of features, this can be computationally demanding. Randomized PCA algorithms [25] reduce this complexity by approximating the singular value decomposition, enabling efficient computation for large feature spaces. However, the fundamental constraint of linearity remains unaddressed by all these algorithmic variants [20].

The experimental results of this study confirm the limitations of PCA for high-dimensional user behavioral data: PCA achieves a Silhouette Score of 0.21 and kNN F1 of 0.56, compared to 0.48 and 0.71 for the proposed autoencoder-based method. In look-alike audience detection, the best traditional classifier (LightGBM) trained on PCA-reduced features achieves a Lift Top 1 of 6.6, compared to 11.7 for cosine similarity with multi-entity autoencoder embedding. These results confirm the inadequacy of linear projection methods for capturing the complex nonlinear structure of high-dimensional user data [4].

The relationship between PCA and information theory provides a useful perspective on its limitations. PCA maximizes variance in the projected space, which is equivalent to maximizing the mutual information between the projected representation and the original data under a Gaussian distributional assumption. When the data distribution is not Gaussian - as is typically the case for mixed-type tabular user data - PCA's variance maximization objective does not correspond to information maximization, and the resulting projection may discard information that is not captured by the leading principal components but is nonetheless relevant for downstream tasks. The autoencoder's reconstruction objective, by contrast, directly minimizes the information loss during compression without assuming any particular distributional form for the data, making it more appropriate for heterogeneous tabular inputs where Gaussianity is not a reasonable assumption [15, 19].

Sparse PCA and robust PCA variants extend the standard method to handle specific structural assumptions about the data. Sparse PCA constrains the loading vectors to have few non-zero entries, producing principal components that depend on only a small subset of the original features and are therefore more interpretable. Robust PCA decomposes the data matrix into a low-rank component and a sparse error component, handling gross outliers that would otherwise distort standard PCA. While these variants address specific limitations of standard PCA, they retain the fundamental constraint of linear projection and do not provide meaningful improvements for the high-dimensional heterogeneous user profiling task investigated in this dissertation. Their computational overhead relative to standard PCA also makes them less practical for the 948-feature, 900,000-subscriber dataset used here [21, 24].

1.3 Nonlinear Manifold Learning and t-SNE

Manifold learning methods represent a family of nonlinear dimensionality reduction techniques based on the geometric assumption that high-dimensional data

points lie on or near a lower-dimensional manifold embedded within the original feature space [26]. This assumption is motivated by the observation that even very high-dimensional datasets may have a much smaller intrinsic dimensionality if the data distribution is concentrated near a smooth low-dimensional surface. By preserving the geometric structure of this manifold rather than the global linear variance, manifold learning methods can reveal nonlinear patterns invisible to PCA.

The foundational contributions of Roweis and Saul (2000) on Locally Linear Embedding (LLE) and Tenenbaum et al. (2000) on Isomap established the theoretical and practical framework for manifold learning [26, 27, 28]. LLE preserves local linear reconstructions by expressing each data point as a weighted linear combination of its nearest neighbors and then finding a lower-dimensional embedding that preserves these reconstruction weights. Isomap extends classical multidimensional scaling (MDS) to nonlinear manifolds by replacing Euclidean distances with geodesic distances estimated along the manifold surface. Both methods demonstrated the ability to unfold genuinely nonlinear data structures that PCA cannot represent.

t-Distributed Stochastic Neighbor Embedding (t-SNE), introduced by van der Maaten and Hinton (2008), has become the dominant algorithm for high-dimensional data visualization and exploratory analysis [29]. The method converts pairwise distances between observations into conditional probability distributions using Gaussian kernels in the high-dimensional space and Student t-distributions in the embedding space. By minimizing the Kullback–Leibler divergence between these distributions, t-SNE effectively reveals cluster structures and local neighborhood relationships that are informative for understanding the data organization.

t-SNE has been widely applied in various domains, including genomics, natural language processing, image analysis, and user behavior analysis. Its ability to create visually meaningful two-dimensional or three-dimensional projections of complex high-dimensional datasets has made it an invaluable exploratory tool. Several studies have used t-SNE visualizations to validate the quality of learned representations, including confirming the separation of user segments in recommendation systems and verifying the structure of embedding spaces learned by neural networks [30].

However, t-SNE suffers from several fundamental limitations that restrict its applicability in production look-alike systems. The most critical limitation is the absence of a parametric mapping: t-SNE computes embeddings for a fixed dataset through an iterative optimization process, but does not learn a function that can map new data points to the embedding space without rerunning the complete algorithm. This means that every time new subscribers join the system, their embeddings must be computed from scratch using the full dataset, which is computationally prohibitive for systems with millions of users and continuous data updates [30].

The computational complexity of standard t-SNE is $O(n^2)$ in time and space, where n is the number of data points. The Barnes-Hut approximation reduces this to $O(n \log n)$ by grouping distant points, but this improvement is insufficient for datasets of the scale encountered in telecommunications systems with hundreds of thousands or millions of subscribers. Furthermore, t-SNE results are sensitive to hyperparameter choices - particularly the perplexity parameter - and different random initializations

can produce qualitatively different embeddings, reducing the reproducibility of analyses [30].

Parametric t-SNE variants that learn explicit encoding functions have been proposed to address the absence of a mapping function. Van der Maaten (2009) [31] introduced a parametric version using a deep neural network as the encoder, but this approach inherits the complexity of neural network training without the systematic reconstruction objective of autoencoders. UMAP (McInnes et al., 2018) offers a parametric extension of manifold learning with better computational efficiency and the ability to embed new points, but is primarily designed for visualization and does not incorporate the reconstruction-based training signal that makes autoencoders suitable for downstream classification tasks [32].

In the experimental evaluation of this study, t-SNE achieves a Silhouette Score of 0.34 and kNN F1 of 0.63, outperforming PCA but substantially falling short of autoencoder-based representations (0.48 and 0.71 respectively). More critically, the absence of a parametric mapping makes t-SNE unsuitable for integration into the production look-alike service, where embeddings must be generated efficiently for new subscribers without access to the full historical dataset.

The Laplacian Eigenmaps algorithm [33] represents another manifold learning approach that constructs a graph connecting nearby data points and then finds embeddings that preserve the graph's spectral structure. Unlike t-SNE, which uses probability distributions to model pairwise similarities, Laplacian Eigenmaps uses the graph Laplacian - a matrix encoding the local connectivity structure of the data - to define the embedding objective. The resulting embeddings minimize the sum of squared distances between connected points, encouraging nearby points in the original space to remain close in the embedded space. While theoretically elegant, Laplacian Eigenmaps shares the limitation of all non-parametric manifold learning methods: the absence of an out-of-sample extension that can map new data points without recomputing the full graph. Belkin and Niyogi (2003) proposed a Nyström-type approximation for out-of-sample extension, but its accuracy degrades for points that are far from the training data, limiting its practical utility for production systems [26].

A practical comparison of t-SNE and UMAP reveals important trade-offs relevant for the look-alike detection application. t-SNE places greater emphasis on preserving local neighborhood structures at the potential cost of global structure, which can result in visually appealing cluster separations that do not accurately represent the global organization of the data. UMAP, by contrast, attempts to preserve both local and global structure through a different mathematical formulation based on fuzzy topological representations. For the user behavioral data analyzed in this dissertation, the global structure of the embedding space is critically important for look-alike search: the system must not only identify compact local clusters of similar users but also correctly rank users across the entire embedding space by their similarity to the reference seed audience. This global ranking requirement further motivates the adoption of autoencoders, whose reconstruction objective explicitly encourages a globally consistent representation [29, 32].

1.4 Autoencoder-Based Representation Learning

Autoencoders are neural network architectures trained to compress input data through a bottleneck layer and reconstruct the original input from the compressed representation [14, 34]. The foundational concept was established by Rumelhart, Hinton, and Williams (1986) [35] in the context of backpropagation, and the potential of deep autoencoders for learning efficient data representations was formally demonstrated by Hinton and Salakhutdinov (2006), who showed that deep autoencoders trained using a greedy layer-wise pretraining strategy could learn more informative representations than PCA [24, 25].

The encoder function $f_\theta: \mathbb{R}^d \rightarrow \mathbb{R}^k$ transforms a high-dimensional input x into a lower-dimensional latent representation $z = f_\theta(x)$, where $k < d$. The decoder function $g_\phi: \mathbb{R}^k \rightarrow \mathbb{R}^d$ then reconstructs the input as $\hat{x} = g_\phi(z)$. The model parameters are optimized by minimizing the reconstruction error, typically measured using Mean Squared Error (MSE) between x and $\hat{x}$ over the training dataset. Because the network must learn to represent the data faithfully through a bottleneck of reduced dimensionality, it is forced to discover the most informative structural properties of the input distribution [15, 36, 37].

The key advantage of autoencoders over manifold learning methods lies in their parametric nature. Once trained, the encoder function f_θ provides an explicit mapping that can be applied to new, previously unseen data points without any additional computation. This property makes autoencoders directly suitable for production deployment in systems that must continuously process new data. Furthermore, the learned representations capture nonlinear patterns through the composition of multiple learnable transformations, enabling the model to represent complex feature interactions that PCA cannot capture [15, 38].

Various autoencoder architectures have been proposed to improve representation quality for different data types and tasks. Denoising autoencoders (Vincent et al., 2008) train the network to reconstruct clean inputs from corrupted versions, forcing the model to learn more robust and abstract features [39]. Sparse autoencoders introduce sparsity constraints on the latent representation, promoting the development of specialized latent units that respond to specific input patterns [40]. Variational autoencoders (Kingma and Welling, 2013) impose a probabilistic prior on the latent space, enabling smooth interpolation between representations and the generation of new samples. Contractive autoencoders penalize the sensitivity of the latent representation to input perturbations, producing representations that are stable under small input variations.

Stacked autoencoders (SAE) build deep hierarchical representations through multiple layers of encoding and decoding [41]. By learning progressively more abstract representations at each layer, SAEs can capture complex multilevel feature hierarchies that shallow architectures cannot represent. The stacked architecture used in this dissertation - with two hidden layers of 1,000 neurons each in both the encoder and decoder - allows the model to learn such hierarchical representations while maintaining computational tractability for the 948-dimensional input space.

The application of autoencoders to tabular data has received growing research attention. Borisov et al. (2024) provided a comprehensive survey of deep neural networks for tabular data, confirming that representation learning approaches can achieve competitive or superior performance compared to gradient boosting methods on many classification and regression benchmarks [41]. Abrar and Samad (2022) demonstrated that perturbation of autoencoder weights enables effective model compression while maintaining classification accuracy on tabular datasets [42]. Delong and Kozak (2023) investigated the use of autoencoders for training neural networks with mixed categorical and numerical features, directly addressing the challenge encountered in telecommunications user data [11].

For heterogeneous tabular user data in telecommunications, the preprocessing challenge must be carefully addressed. Categorical features must be one-hot encoded, numerical features must be normalized, and multi-valued categorical attributes - such as lists of web interests or application categories - must be encoded through vocabulary filtering and binary expansion. The preprocessing pipeline described in Chapter 2 of this dissertation addresses these challenges systematically, producing a homogeneous numerical input suitable for autoencoder training while preserving the semantic information contained in each feature domain [1, 4].

Recent research has explored ensemble approaches combining multiple autoencoders trained on different feature subsets. Berahmand et al. (2024) provided a survey of autoencoder applications covering ensemble methods, anomaly detection, and multi-modal representation learning, confirming the potential of domain-specific autoencoder training for heterogeneous data sources [43]. The multi-entity embedding strategy proposed in this dissertation directly instantiates this research direction for telecommunications user data, where features from multiple behavioral entity domains are concatenated into a single representation and processed through a stacked autoencoder [8].

The question of optimal latent space dimensionality for autoencoder-based user representations has been studied in several applied contexts. In natural language processing, word embedding dimensionality is typically set to 300 (Word2Vec, GloVe) or 768 (BERT), reflecting a practical trade-off between representational capacity and computational cost. For tabular user behavioral data, the optimal dimensionality is less well-established and depends heavily on the complexity and heterogeneity of the input feature space. Hu et al. (2024) [44] analyzed autoencoder architectures for large-scale multi-objective optimization, finding that latent space dimensionality significantly affects the quality of learned representations for complex multi-dimensional problems [45]. For the 948-dimensional user feature space investigated in this dissertation, the ablation study reported in Chapter 3 identifies 288 as the optimal dimensionality, representing a compression ratio of approximately 3.3:1 that balances reconstruction quality (MSE = 0.61) with downstream representation utility (Silhouette Score = 0.48).

The relationship between autoencoder-based representation learning and more recent self-supervised learning paradigms is worth examining. Self-supervised learning methods - including masked autoencoders (He et al., 2022) [45, 46], contrastive predictive coding (van den Oord et al., 2018), and BYOL (Grill et al., 2020) - have achieved remarkable results in visual and language domains by training encoders to

predict masked or transformed portions of the input. These methods can be viewed as generalized autoencoders where the reconstruction target is not the full input but a partially masked or augmented version. The masked autoencoder variant for tabular data (SCARF, Bahri et al., 2022) [47] has shown promise for pre-training representations that transfer well to downstream classification tasks. While such advanced self-supervised approaches were not the focus of this dissertation, they represent a natural extension of the reconstruction-based autoencoder framework that could further improve representation quality for the look-alike detection application [14, 36].

1.5 Look-Alike Audience Modeling in Targeted Advertising

Look-alike audience modeling refers to the identification of users whose behavioral and demographic characteristics are similar to those of a known seed audience - a reference group of users who have previously engaged with a specific product, service, or advertisement [5]. The objective is to expand the addressable audience for advertising campaigns by discovering users in the broader population who share relevant characteristics with known converters. This task is fundamental to targeted digital advertising and has become a core capability of modern advertising platforms.

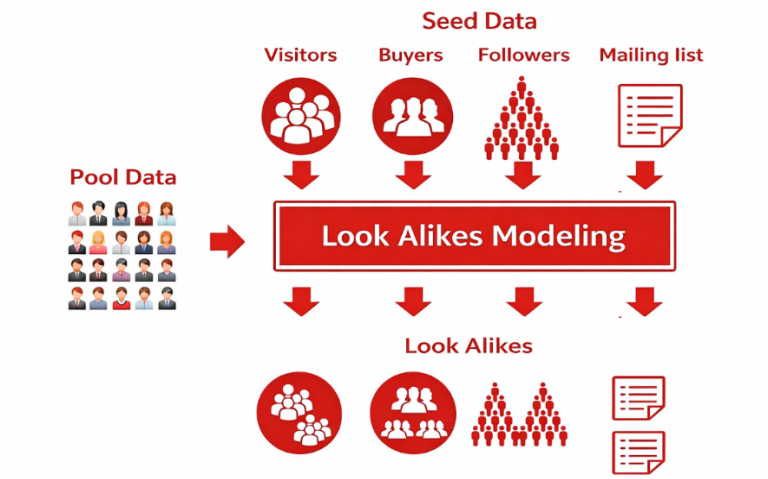

Figure 1 – Conceptual design of a look-alike audience modeling service

Figure 1 illustrates the conceptual design of a look-alike modeling service. A pool of candidate users is scored against multiple types of seed audiences - visitors, buyers, followers, and mailing list subscribers - each representing a distinct behavioral reference group. The look-alike modeling component processes both the pool data and seed definitions simultaneously, producing separate expanded audiences that mirror the behavioral profile of each seed type. This parallel multi-seed architecture directly motivates the generalized embedding-based approach proposed in this dissertation: rather than training a dedicated classifier per seed type, a single universal user representation enables efficient similarity retrieval across all seed definitions without retraining.

The historical development of look-alike modeling tracks the broader evolution of machine learning in advertising technology. Early approaches relied on simple demographic matching or collaborative filtering: finding users who shared demographic categories (age group, geographic region, device type) with the seed audience. While computationally simple, these methods are limited by their inability to capture complex, nonlinear relationships between behavioral signals and the target behavior of interest [48, 49].

The adoption of binary classification models represented the first major advance. A binary classifier - typically logistic regression, a gradient boosting model, or a support vector machine - is trained on a dataset of positive examples (seed users) and negative examples (non-seed users), and then applied to score all subscribers by their estimated probability of belonging to the seed audience. Ma et al. (2016) proposed a sub-linear massive-scale look-alike audience extension system based on logistic regression with hashed features, demonstrating computational scalability for industrial deployment [48]. Bagherjeiran et al. (2015) described a patent for adaptive look-alike targeting that adjusts the feature representation based on the specific campaign objectives [5].

Binary classification models require campaign-specific training: a separate model must be trained for each new seed audience definition, which requires significant analyst effort for feature selection, model validation, and deployment. This operational overhead prevents the approach from scaling to serve a large number of B2B clients simultaneously and limits the speed with which new campaigns can be launched. Furthermore, binary classifiers require explicit negative examples, which may be difficult to define correctly for diverse campaign objectives [6].

Embedding-based look-alike systems emerged as a more flexible alternative, decoupling the user representation learning phase from the look-alike search phase. In these systems, each user is first mapped to a dense vector in a learned embedding space, and look-alike search is then implemented as nearest-neighbor retrieval in this space. This architecture enables a single trained model to serve diverse campaigns without retraining: any seed audience can be represented as a set of embedding vectors, and similar users can be identified by computing cosine or Euclidean distances to other vectors in the database [50, 51].

A critical requirement for industrial look-alike systems is the ability to operate as generalized services that handle diverse targeting requests simultaneously. Different B2B clients may submit seed audiences defined by entirely different behavioral criteria - purchasing patterns, geographic activity, application usage, demographic profiles - and the underlying representation must capture general behavioral patterns relevant across all these dimensions. This requirement motivates the unsupervised learning approach adopted in this dissertation, where autoencoders are trained purely on reconstruction objectives without access to any campaign-specific labels, ensuring that the learned representations capture general behavioral structure rather than task-specific signals [6, 37].

The evaluation methodology for look-alike systems presents specific challenges. If the same target variable is used both for training the representation model and for evaluating look-alike performance, the evaluation metrics will be artificially inflated

through target leakage. This dissertation addresses this challenge by evaluating embeddings on 17 independent validation datasets with external target variables that were entirely absent during autoencoder training, providing unbiased estimates of the generalization ability of the learned representations across diverse targeting scenarios [52].

The choice of evaluation methodology for look-alike systems significantly influences the conclusions drawn about model performance. Rainio et al. (2024) highlighted the importance of selecting appropriate evaluation metrics for machine learning systems that serve specific operational objectives [52]. For look-alike audience modeling, the Lift Top 1 metric is particularly informative because it directly measures the model's ability to identify the most valuable users within the target audience - a capability that is central to the economic justification of the system. A Lift Top 1 of 12.9, as achieved by the proposed Siamese network system, means that the subscribers ranked highest by the model are 12.9 times more likely to convert than a randomly selected subscriber, enabling highly efficient allocation of advertising budget.

The scalability requirements of industrial look-alike systems impose additional constraints beyond predictive accuracy. Al-Otaibi (2024) discussed the computational requirements of deep learning frameworks for customer behavior prediction in e-commerce, noting that inference latency is a critical constraint when systems must process user similarity queries in near real-time [51]. For the proposed framework, the use of precomputed embeddings stored in HDFS enables look-alike search to be implemented as a fast vector retrieval [53] operation, decoupling the computationally expensive embedding generation (performed monthly in batch mode) from the similarity search phase (performed on-demand in response to B2B client requests).

Privacy considerations represent an additional dimension that distinguishes look-alike modeling in regulated industries such as telecommunications from research settings. Lom et al. (2024) analyzed privacy concerns in mobile advertising contexts, emphasizing that user data used for audience modeling must be handled in accordance with applicable data protection regulations and that transparency about data usage is essential for maintaining user trust [54]. The dataset used in this dissertation was fully anonymized before analysis, representing aggregated statistical indicators rather than raw individual-level behavioral events. This anonymization strategy enables the development of effective behavioral representations while complying with privacy regulations and eliminating the risk of re-identification from the processed data.

The business economics of look-alike audience modeling systems are worth examining to understand the performance thresholds that determine commercial viability. For a telecommunications operator with millions of subscribers, a 1-percentage-point improvement in Conversion Rate across a portfolio of advertising campaigns translates directly into significant additional revenue from B2B clients who pay per conversion or per click. The improvement from CR = 0.19 (LightGBM baseline) to CR = 0.31 (multi-entity cosine similarity) represents a 63% relative improvement that, when applied to campaigns at the scale of this study (90,000 subscribers per campaign, dozens of campaigns per month), generates revenue improvements that comfortably justify the infrastructure investment in the

autoencoder-based system. This economic argument is consistent with published industry reports on the ROI of advanced machine learning for advertising: Gustriansyah et al. (2024) noted that machine learning–based audience segmentation consistently delivers measurable improvements in campaign efficiency, with the magnitude of improvement correlated with the sophistication of the behavioral representation [49].

The concept of audience quality - distinct from audience size - is central to evaluating look-alike systems in practice. A look-alike system that selects 90,000 subscribers with a Conversion Rate of 0.36 delivers 32,400 conversions, while a random selection of 90,000 subscribers at the baseline conversion rate of approximately 2.8% (the organic conversion rate in the target market) would deliver only 2,520 conversions. This 12.9× improvement - represented by the Lift Top 1 metric - is the key value proposition of the proposed system for B2B advertising clients. Understanding that Lift Top 1 measures improvement over random baseline, rather than over a competing model, helps practitioners correctly interpret the reported metrics and communicate the system's value to non-technical stakeholders [5, 50].

1.6 Siamese Networks for Similarity Learning

Siamese networks are a specialized neural network architecture designed for similarity measurement and comparison tasks. First proposed by Bromley et al. (1993) for handwritten signature verification, the architecture consists of two or more identical subnetworks - twin branches - that share weights and parameters [55]. Each branch processes one input in a pair and produces a fixed-length feature representation. The representations from the two branches are then combined using a similarity function - typically Euclidean distance or cosine similarity - to produce a scalar similarity score.

The weight-sharing constraint enforces that both branches apply the same transformation to their respective inputs, ensuring that the similarity function is symmetric and that representations from the same input space are directly comparable. This design allows the network to learn a discriminative similarity function directly from labeled pairs of similar and dissimilar examples, rather than relying on geometric distance measures computed in the original feature space [56, 57].

Siamese networks gained widespread adoption in computer vision applications, including face verification (Koch et al., 2015 [58]; FaceNet, Schroff et al., 2015 [59]), person re-identification, and one-shot learning. In these domains, the network learns to map visually similar inputs close together and dissimilar inputs far apart in the embedding space, enabling recognition of new classes from only one or a few examples [40]. The key insight is that the network learns a task-specific metric - a similarity function that captures what it means for two inputs to be "similar" in the context of the training task - rather than using a universal geometric distance.

In the context of tabular user data and look-alike modeling, Siamese networks provide a mechanism for learning a discriminative similarity function that adapts to the specific distributional properties of the behavioral embedding space. Unlike cosine similarity, which treats all dimensions of the embedding space equally, the Siamese network can learn to emphasize dimensions that are most informative for

distinguishing similar from dissimilar users in the training data. This adaptive weighting is particularly valuable in the concatenated multi-entity embedding space, where different dimensional subspaces correspond to different behavioral domains with varying relevance for specific targeting tasks [45, 60].

The training procedure for Siamese networks requires the construction of labeled pairs of similar and dissimilar examples. For look-alike modeling, positive pairs consist of users who both belong to the reference seed audience - they are similar by definition - while negative pairs pair seed users with randomly sampled non-seed subscribers. The network is trained using contrastive loss [61] or triplet loss to minimize the distance between positive pairs and maximize the distance between negative pairs in the learned similarity space [54].

Serrano and Bellogín (2023) investigated Siamese networks for collaborative filtering in recommendation systems, demonstrating their effectiveness for capturing complex user similarity patterns in sparse behavioral data [56]. Zhang et al. (2023) showed that similarity-based pair construction strategies during Siamese network training significantly affect model performance, motivating the careful design of the pair sampling procedure [57]. Fedele et al. (2024) analyzed the explainability of Siamese networks in few-shot learning, highlighting the interpretability challenges that arise with learned similarity functions compared to geometric distance measures [45]. These studies collectively confirm the potential of Siamese networks for user similarity learning while highlighting the importance of training procedure design.

The broader ecosystem of representation learning methods for tabular data has expanded considerably in recent years. Transformer-based architectures originally developed for natural language processing have been adapted for tabular settings. TabTransformer (Huang et al., 2020) [61, 62] applies self-attention to categorical embeddings, allowing the model to learn context-dependent representations of categorical features based on the values of other features in the same row. FT-Transformer (Gorishniy et al., 2021) [63] extends this approach to numerical features through a feature tokenization mechanism, producing fully attention-based representations of tabular inputs. While these methods demonstrate promising results on classification benchmarks, their application to large-scale heterogeneous user profiling tasks - where the number of features is in the hundreds and the number of samples is in the hundreds of thousands - introduces challenges related to computational efficiency and attention matrix scalability that autoencoders do not face [12, 41].

Contrastive learning methods, which train encoders by maximizing agreement between different augmented views of the same input, represent another promising direction for tabular representation learning. The SimCLR framework (Chen et al., 2020) [64] and its variants have produced state-of-the-art results for visual representation learning, and adaptations for tabular data - using feature masking and noise injection as augmentation strategies - have shown competitive performance on classification benchmarks. However, contrastive methods require careful design of augmentation strategies that preserve the semantic content of behavioral profiles, which is domain-specific knowledge not readily available without expert input. The reconstruction-based autoencoder objective, while not directly optimized for

downstream similarity tasks, provides a more straightforward training signal that does not require domain-specific augmentation design [14, 65].

Graph neural network approaches have also been explored for user representation learning in scenarios where relational information between users is available - for example, social connections, co-purchase patterns, or shared interest categories. These methods learn representations that capture not only individual user features but also the structural position of each user within the behavioral graph. For the look-alike application described in this dissertation, explicit relational information between subscribers is not available in the training data, making graph-based approaches inapplicable in their standard form. However, the implicit neighborhood relationships captured by cosine similarity in the autoencoder embedding space serve a functionally similar role, identifying users whose behavioral profiles position them near specific reference points in the latent space.

1.7 Summary

The literature review reveals a clear progression from classical linear dimensionality reduction methods through nonlinear manifold learning to deep representation learning frameworks for tabular data analysis. PCA provides a computationally efficient baseline but is fundamentally limited to linear relationships. t-SNE improves visualization and local structure preservation but lacks a parametric mapping function, preventing its use in scalable production systems. Autoencoders address both limitations by learning nonlinear parametric transformations applicable to new data.

For look-alike audience modeling, the existing literature demonstrates a clear advantage of embedding-based approaches over traditional binary classification methods, but systematic investigation of autoencoder-based multi-entity embedding applied to large-scale heterogeneous telecommunications user data with rigorous multi-campaign evaluation has not been conducted. The integration of Siamese networks with autoencoder embeddings for tabular user similarity learning also represents an underexplored combination that this dissertation investigates.

The research gap addressed is therefore the systematic development and evaluation of an autoencoder-based representation learning framework for high-dimensional heterogeneous tabular user data, incorporating a multi-entity embedding strategy, Siamese network similarity learning, and validation through multiple independent look-alike campaign tasks with external target variables. The following chapters describe the proposed framework and its experimental validation.

2 DATASET AND PREPROCESSING

This section describes the dataset used in the experimental evaluation and the complete preprocessing pipeline applied to prepare the data for dimensionality reduction and representation learning. The quality of learned representations depends critically on the quality and consistency of the input data, making preprocessing one of the most important stages of the methodology.

The dataset used in this dissertation represents one of the largest and most heterogeneous user behavioral datasets applied to look-alike audience modeling in the published literature. The combination of telecommunications activity, web behavior, financial transaction, device, network, and tariff plan data into a unified six-entity representation provides a far richer behavioral portrait of each subscriber than is typically available in datasets used for academic machine learning research, where single-source behavioral datasets of modest size are the norm. This richness is both an opportunity - enabling more accurate behavioral similarity assessment - and a challenge, as the heterogeneity of the data types and the varying completeness of each entity domain require careful preprocessing to produce a model-ready representation. The preprocessing pipeline described in this chapter was developed iteratively over multiple months of production system development, with each step validated through its impact on downstream model performance rather than through abstract data quality metrics alone [1, 12].

The ethical and legal framework governing the use of this dataset is important context for interpreting the research. The telecommunications operator obtained informed subscriber consent for the use of behavioral data for advertising personalization through its standard service agreement terms, which comply with the requirements of Kazakhstan's personal data protection legislation (Law on Personal Data and Its Protection, 2013) [66]. The anonymization protocol applied to the dataset removes all direct identifiers (subscriber phone numbers, names, and contract identifiers) and replaces them with internal system IDs that cannot be reverse-mapped without access to the operator's internal identity resolution systems. The use of aggregated statistical indicators rather than raw event logs provides a further layer of privacy protection, ensuring that no individual's specific activities can be inferred from the processed dataset. These protections are consistent with the requirements of data protection best practices for machine learning research and ensure that the findings of this dissertation do not contribute to any privacy risk for the subscribers whose data was used.

2.1 Dataset Description and Entity Structure

The experiments were conducted using a large-scale anonymized dataset collected from a telecommunications and digital services platform during regular service operation. The dataset was obtained through a collaboration agreement with the operator. 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, to develop a flexible look-alike audience detection model [1, 12].

To ensure compliance with privacy and data protection regulations, all records used in the study were fully anonymized and did not contain any personally identifiable information. The dataset represents aggregated statistical indicators rather than raw user-level events, which further ensures the protection of sensitive information. Due to confidentiality constraints, the dataset is not publicly available; however, access was granted within the scope of a research partnership, ensuring that all data handling adhered to stringent privacy and ethical standards.

The dataset covers approximately 900,000 subscribers. Initially, the raw dataset contained 2,814 features describing different aspects of user behavior and system interaction, derived from multiple heterogeneous data collection systems operating over a three-month observation period. After applying the preprocessing pipeline described in subsequent sections, the final dataset used in experiments consisted of 948 features organized into six thematic domains referred to as entities.

The User Entity forms the foundation of the subscriber profile. It contains aggregated indicators of telecommunications activity including call volumes and durations (voice traffic metrics for local, national, and international calls, segmented by direction and day period), internet data consumption (2G/3G/4G traffic volumes and roaming usage), financial activity (ARPU, bundle charges, topup behavior), and socio-demographic attributes including modeled age group, gender, income level, household size, and geographic activity patterns derived from home and work location analysis. The User Entity is the most feature-rich domain, with metrics computed as aggregates over a 3-month sliding window to capture behavioral trends rather than single-period snapshots.

The Web Entity captures behavioral data derived from anonymized internet sessions, including web interest categories derived from URL classification across more than 60 topic categories (finance, sports, entertainment, automotive, health, etc.) and mobile application usage categories collected from the operator's application analytics service. All data in this entity were collected with explicit user consent and processed through anonymization procedures that prevent identification of specific websites visited or applications used. The Finance Entity provides a view of users' financial behavior based on anonymized payment service transaction data, enabling the construction of consumer profiles and merchant category preferences.

The Device Entity contains technical descriptions of mobile devices used by subscribers, including hardware specifications (screen size, camera presence, USB type), modeled price tier, operating system type (Android, iOS, proprietary), device model classification, and device type (Smartphone, Feature+, Voice-centered). The Cell Base Station Entity records information on the primary network stations associated with each subscriber, including service quality metrics, signal stability indicators, and network generation usage patterns. The Tariff Plan Entity comprises detailed parameters of each subscriber's active service plan, covering pricing structure, package contents (included minutes, data, SMS), tariff type, and plan age.

Table 1 – Dataset entity structure summary

Entity	Key Feature Groups	Feature Count	Update Frequency
User	Activity, ARPU, demographics, geography	~280	Monthly aggregates
Web	Web interests, app categories	~190	Monthly aggregates
Finance	Payment service transactions	~80	Monthly aggregates
Device	Hardware specs, OS, price tier	~120	Device change detection
Base Station	Network quality, geoactivity	~180	Monthly aggregates
Tariff Plan	Plan parameters, pricing, age	~98	Plan change detection

For experimental evaluation, the dataset was divided into training (70%) and validation (30%) subsets. The training data were used to learn the dimensionality reduction models and latent representations, while the validation subset was used for assessing embedding quality through clustering metrics and for evaluating look-alike detection performance. Critically, the target variables used in look-alike campaign validation were obtained from 17 independent advertising campaigns with external target variables completely absent from the training data, ensuring unbiased performance estimation [52].

The volume of the dataset presents both opportunities and challenges for representation learning. With approximately 900,000 training examples, the autoencoder has access to sufficient data to learn generalizable representations without relying on data augmentation techniques that are common in lower-data regimes. Ghori et al. (2023) demonstrated that machine learning classifiers benefit substantially from large training sets in high-dimensional feature spaces, and this principle applies equally to representation learning methods [67, 68]. At the same time, the large dataset size requires careful attention to computational efficiency during both training and inference, motivating the architectural choices - batch size of 512, gradient accumulation, and distributed processing - described in subsequent chapters.

The temporal structure of the dataset has important implications for the design of the preprocessing pipeline and the evaluation methodology. Features were computed as aggregates over a 3-month sliding window, capturing medium-term behavioral trends rather than transient single-day activities. This choice reflects the business requirement for stable user representations that are robust to daily fluctuations in behavior: a subscriber who made a large purchase in a single day should not be represented as fundamentally different from their typical behavioral profile. Torabi et al. (2023) [69] discussed similar considerations in the context of autoencoder-based

anomaly detection, noting that the temporal granularity of feature aggregation significantly affects the stability and interpretability of learned representations [45].

2.2 Categorical Encoding

All categorical variables in the dataset were transformed into numerical representations using one-hot encoding prior to model training. This transformation is necessary because autoencoders, like all neural network architectures, operate exclusively on numerical inputs. One-hot encoding converts each categorical variable with c unique values into c binary features, where exactly one feature is active (set to 1) for any given observation.

For standard categorical variables - such as device operating system type, device category, tariff plan charge period, and demographic group classifications - one-hot encoding was applied directly after standardizing category labels to lowercase and replacing irregular values (marked as -99 or "null" in the source data) with an "unknown" category. For multi-valued categorical attributes - such as the lists of web interest categories detected for a subscriber or the categories of mobile applications installed - the encoding was applied after filtering each list against a predefined vocabulary of known category values, and computing binary presence/absence indicators for each vocabulary entry [1, 2].

The predefined category vocabularies for multi-valued attributes were derived from the business knowledge of the data collection systems and validated against the empirical distribution of category values in the training data. Categories appearing in fewer than 0.1% of subscribers were grouped into an "other" category to prevent the creation of near-zero features that contribute minimal information while increasing dimensionality. This vocabulary filtering also ensures a fixed and consistent feature space across all subscribers, regardless of the specific categories encountered for each individual.

The choice of one-hot encoding over other categorical encoding strategies - such as ordinal encoding, target encoding, or learned embeddings - was deliberate. Ordinal encoding assigns integer values to categories based on an assumed ordering, which is inappropriate for nominal categories such as device operating system type where no natural ordering exists. Target encoding uses the mean target value for each category as its numerical representation, which would introduce target leakage into the preprocessing pipeline and undermine the unsupervised training objective of the autoencoder. Learned embedding layers, while powerful in natural language processing contexts, require significantly more training data per category to produce reliable representations, which is not guaranteed for low-frequency categories in the device and tariff plan domains [1].

An important consideration in one-hot encoding for high-dimensional datasets is the risk of creating feature vectors that are dominated by zeros. For a subscriber with no recorded web interests, the corresponding 60+ binary features will all be zero, which could cause the autoencoder to ignore these dimensions during training. To mitigate this effect, the preprocessing pipeline includes a strategy of assigning a special "unknown/missing" indicator feature for subscribers with no recorded values in multi-

valued categorical domains, distinguishing genuine absence of interest data from subscribers who simply have no interests in any category. This distinction provides the autoencoder with an additional signal that can help differentiate between subscribers with missing data and those with genuinely empty behavioral profiles.

2.3 Missing Value Imputation

Missing values presented a significant challenge across several feature domains, particularly for attributes that were not applicable to all subscribers. Device features were absent for subscribers using SIM-only connections without an associated device identifier. Financial transaction features were unavailable for subscribers without registered accounts in the payment service system. Web interest features were absent for subscribers who had not granted consent for internet activity analytics.

Figure 2 summarises the missing value rates observed across the six entity domains. The pattern reveals a clear stratification driven by data collection coverage. The User entity is nearly complete, with fewer than 2% missing values for most features, reflecting the universal nature of core telecommunications service tracking - every active subscriber generates call, data, and ARPU records by definition. The Finance entity occupies the opposite extreme, with missingness rates of 15–35% for transaction-based features, as these are only available for subscribers who have registered accounts in the payment service platform. The Web and Device entities occupy an intermediate position (5–15% and 5–12% respectively), reflecting partial consent rates for internet analytics and the proportion of subscribers using unregistered devices. These observed patterns directly informed the choice of imputation strategy for each domain: mean imputation for the nearly complete User entity features, median imputation for the skewed Finance and Base Station distributions, mode imputation for the categorical Device and Tariff Plan attributes, and zero-filling for absent Web interest categories.

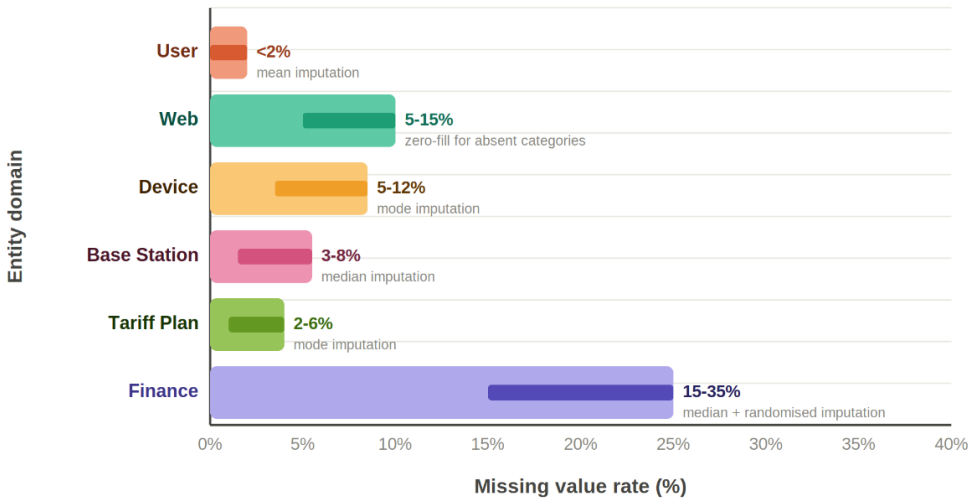

Figure 2 - Missing value rates across six behavioral entity domains

A range of imputation strategies was evaluated. Advanced methods such as Multiple Imputation by Chained Equations (MICE) [70] were initially explored but did not yield significant performance improvements compared to simpler statistical approaches, while being substantially more computationally demanding. Given the large scale of the dataset and the constraints of the production computing environment, imputation was performed using lightweight statistical techniques: mean imputation for approximately normally distributed numerical features, median imputation for skewed distributions (such as ARPU and data volume metrics, which typically exhibit heavy right tails in telecommunications datasets), and mode imputation for categorical features with low cardinality [21, 23].

For multi-valued categorical attributes where a subscriber had no recorded values (e.g., no detected web interests), the feature vector was filled with zeros, indicating absence of all categories. This zero-filling strategy preserves the binary semantics of the encoded features and is consistent with the interpretation that the subscriber does not exhibit the corresponding behavioral patterns. Randomized imputation was applied to fields where a single statistical summary was insufficient to preserve the original distribution without introducing systematic bias [1].

The proportion of missing values varied substantially across feature domains and individual features. The User entity had relatively complete coverage (less than 2% missing values for most features) due to the high penetration of core telecommunications service tracking. The Finance entity had higher missingness rates (15–35% for transaction-based features) due to the partial adoption of the payment service platform among subscribers. The Device entity had intermediate missingness rates (5–12%) due to subscribers who use devices not registered in the device database. Understanding these patterns guided the selection of appropriate imputation strategies for each domain: mean imputation is appropriate for features with low missingness and approximately normal distributions, while median or mode imputation is more appropriate for highly skewed or discrete distributions with high missingness rates [21, 23].

Cha et al. (2023) demonstrated that data preprocessing quality - including missing value handling and feature scaling - has a more significant impact on machine learning model performance than is commonly acknowledged, with models trained on poorly preprocessed data performing substantially worse than the same models trained on carefully cleaned data. This finding motivated the thorough approach to preprocessing adopted in this dissertation, where each preprocessing step was validated by measuring its impact on downstream autoencoder reconstruction quality and embedding structure before being incorporated into the final pipeline [28].

2.4 Outlier Detection and Removal

Outlier detection was performed using isolation forests [71] - an anomaly detection method based on random tree ensembles [29]. The algorithm constructs binary trees by randomly selecting features and split points. Anomalous observations are identified by their shorter average path lengths in the trees, reflecting their

distinctiveness from the majority of the data: outliers require fewer random splits to be isolated because they lie far from the dense regions of the data distribution.

Isolation forests are particularly well-suited for high-dimensional tabular data because they avoid the explicit density estimation or distance computation required by other anomaly detection methods, which become unreliable in high-dimensional spaces due to the curse of dimensionality. The algorithm runs in O(n log n) time and scales efficiently to datasets of the size encountered in this study [29].

Identified outliers were removed from the training dataset to prevent their adverse influence on autoencoder training. Outliers in tabular user data typically correspond to inactive test accounts with unusual usage patterns, data collection errors in sensor systems, or extreme behavioral patterns associated with very recently activated or about-to-churn subscribers. The proportion of removed records was approximately 3–5% of the training set, which did not affect the statistical representativeness of the training data while meaningfully improving the quality of the learned representations.

2.5 Multicollinearity Reduction

Redundant features were identified and removed through pairwise Pearson correlation analysis. The correlation coefficient between all pairs of numerical features was computed, and when two variables exhibited a coefficient above a predefined threshold, the feature with lower marginal informativeness - assessed through variance and mutual information with proxy behavioral signals available in the training data - was removed.

The correlation threshold of 0.77 was determined empirically through iterative experimentation, testing values in the range [0.5, 1.0] and evaluating model performance at each threshold. At thresholds below 0.6, excessive features are removed, losing informative behavioral signals. At thresholds above 0.9, many redundant features are retained, increasing computational cost without improving representation quality. The value of 0.77 delivered the best balance between precision, recall, and model stability for this specific dataset and was therefore selected as the final threshold [1, 72].

This step reduced the dimensionality of the feature space from 2,814 raw attributes to 948 features used in all subsequent experiments, eliminating redundant information while preserving the most discriminative characteristics of each entity domain. The reduction factor of approximately 3x demonstrates the high degree of collinearity present in the raw feature set, which is typical for telecommunications datasets where many metrics are derived from the same underlying behavioral signals.

An important characteristic of multicollinearity in telecommunications datasets is that it often reflects genuine causal relationships between behavioral metrics rather than redundant measurement of the same underlying quantity. For example, the correlation between data consumption volume and internet session duration reflects a genuine behavioral tendency rather than a measurement artifact. Removing one of these features does not eliminate the underlying behavioral signal - it is still captured by the retained feature - but reduces the redundancy in the input representation,

allowing the autoencoder to allocate its capacity more efficiently. The pairwise correlation analysis was therefore conducted iteratively, removing one feature at a time and recomputing correlations among the remaining features to avoid cascading effects where the removal of one feature changes the correlation structure of others [72].

2.6 Feature Scaling

All numerical variables were normalized using min-max scaling, which transforms feature values into the range [0, 1] according to:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where x_{min} and x_{max} are the minimum and maximum values observed in the training data for the respective feature. The scaling parameters were computed on the training data and applied consistently to both training and validation datasets to prevent any information about the validation distribution from influencing the learned model [30].

This normalization ensures that all features contribute proportionally during model training and prevents variables with larger numerical ranges - such as data volume measured in megabytes - from disproportionately influencing the autoencoder's MSE reconstruction loss relative to binary or low-range categorical features. Without normalization, the reconstruction loss would be dominated by high-magnitude features, causing the autoencoder to allocate most of its capacity to reconstructing these features while ignoring the fine-grained behavioral signals encoded in binary and categorical attributes.

The complete preprocessing pipeline was implemented in Python using a combination of the Pandas library for data manipulation and the scikit-learn library for statistical transformations [73]. The implementation follows a scikit-learn [74] Pipeline architecture, where all transformation steps are encapsulated in a single serializable object that applies the same sequence of operations to any input dataset. This design ensures that the exact same preprocessing operations - with parameters fitted on the training data - are applied to validation and production data, eliminating the risk of preprocessing inconsistencies between training and deployment environments. The Pipeline object is serialized to disk using the joblib library and loaded at inference time, providing a complete and reproducible preprocessing specification [30].

The choice of Apache Spark [75] for large-scale preprocessing deserves additional comment. PySpark enables the preprocessing pipeline to operate on datasets that exceed the memory capacity of a single machine by distributing the computation across multiple nodes in the Hadoop cluster. For the 900,000-subscriber dataset with 948 features, the full preprocessing run requires approximately 25 minutes on a 16-node cluster, which is well within the overnight batch processing window available before the monthly embedding update. The use of columnar Parquet file format for intermediate storage provides efficient compression (the 948-feature dataset occupies approximately 8GB in Parquet format vs 45GB in raw CSV) and enables predicate

pushdown - the ability to load only the feature columns required for a specific downstream task without reading the full dataset [8].

Feature importance analysis was conducted as a diagnostic step after preprocessing to understand which feature categories contribute most to the variance in the preprocessed dataset. Using variance-based ranking, the telco activity features (voice traffic metrics, data volumes) accounted for approximately 35% of total dataset variance, followed by demographic one-hot features (22%), device features (18%), web interest binary features (12%), tariff plan features (8%), and financial transaction features (5%). This ranking reflects the relative density and magnitude of different feature categories but does not necessarily reflect their importance for the look-alike detection task - low-variance features such as device type indicators can still be highly informative for identifying specific behavioral subgroups even if they contribute little to overall dataset variance.

2.7 Summary

This chapter described the large-scale anonymized telecommunications dataset comprising 900,000 subscribers and 948 features organized into six behavioral entity domains, and the five-stage preprocessing pipeline applied to prepare the data for representation learning. The preprocessing pipeline - categorical encoding, missing value imputation, isolation forest outlier detection, pairwise correlation-based multicollinearity reduction (threshold 0.77), and min-max scaling - was applied consistently to both training and validation datasets. These steps reduced the raw feature space from 2,814 attributes to 948 features while preserving the most informative behavioral signals. The resulting standardized dataset provides the input for the dimensionality reduction methods described in the following chapter.

3 DIMENSIONALITY REDUCTION METHODS

This chapter presents the mathematical formulations and implementation details of the three dimensionality reduction approaches evaluated in this study: Principal Component Analysis as a linear baseline, t-distributed Stochastic Neighbor Embedding as a nonlinear manifold learning method, and stacked autoencoders as the proposed deep learning–based representation learning framework. The multi-entity embedding strategy and cosine similarity computation are also described.

Dimensionality reduction methods can be broadly categorized along two orthogonal axes: whether the transformation is linear or nonlinear, and whether it is parametric or non-parametric. Linear methods assume that the relevant structure in the data can be captured through linear combinations of the original features, while nonlinear methods relax this assumption and are capable of representing curved manifolds and complex interaction effects. Parametric methods learn an explicit function mapping inputs to embeddings, enabling out-of-sample extension - the ability to embed new data points without recomputing the entire model. Non-parametric methods compute embeddings for a fixed dataset but do not generalize to new points without additional computation. This two-axis taxonomy places PCA as linear and parametric, t-SNE as nonlinear and non-parametric, and autoencoders as nonlinear and parametric. For the production look-alike service described in this dissertation, the parametric property is a hard requirement, which immediately eliminates standard t-SNE from consideration for the production inference pipeline regardless of its embedding quality [14, 19, 29].

A complementary consideration is whether the dimensionality reduction method optimizes a reconstruction objective or a distribution-preserving objective. Reconstruction-based methods - including PCA and autoencoders - minimize the average error between the original input and its reconstruction from the lower-dimensional representation. Distribution-preserving methods - including t-SNE and UMAP - minimize divergences between similarity distributions in the original and embedded spaces. The reconstruction objective has the advantage of being a well-defined global loss that encourages the entire embedding space to be informationally consistent, while distribution-preserving objectives can produce visually striking local cluster separations but may not preserve global structure accurately. For the look-alike application, where the system must correctly rank subscribers across the entire embedding space by their global similarity to a reference audience, the reconstruction objective of the autoencoder is better aligned with the operational requirements than the local distribution-preserving objective of t-SNE [15, 29].

3.1 Principal Component Analysis

PCA identifies the directions of maximum variance in the data by computing the eigenvalue decomposition of the sample covariance matrix. Given a zero-mean data matrix $X \in \mathbb{R}^{(n \times d)}$, the sample covariance matrix is:

$$C = \frac{X^\top X}{n-1} \tag{2}$$

where eigenvalue decomposition $C = V\Lambda V^T$ yields eigenvectors $V \in \mathbb{R}^{(d\times d)}$ and corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$. The k-dimensional PCA projection is obtained by selecting the top-k eigenvectors $V_k \in \mathbb{R}^{(\mathrm{d}\times\mathrm{k})}$ and computing:

$$Z = XV_k \tag{3}$$

where $Z \in \mathbb{R}^{(n\times k)}$ is the reduced representation. The proportion of variance explained by the *j*-th principal component is $\lambda_j/\Sigma_i\lambda_i$, and the cumulative explained variance for k components provides a measure of information retention. PCA guarantees optimal linear reconstruction of the data in the least-squares sense: among all *k*-dimensional linear projections, PCA minimizes the average squared reconstruction error.

For the dataset used in this study, PCA was applied after the full preprocessing pipeline, computing the decomposition on the training data and projecting both training and validation sets. Three-dimensional projections were used for visualization, while higher-dimensional projections (k = 50, 100, 200) were evaluated for downstream kNN classification tasks. The explained variance curve was analyzed to understand the effective intrinsic dimensionality of the user behavioral data.

3.2 t-Distributed Stochastic Neighbor Embedding

t-SNE converts pairwise distances between observations into conditional probability distributions modeling the similarity between data points. In the high-dimensional space, the similarity between observations x_i and x_j is defined using a Gaussian distribution centered at x_i:

$$p_{ij} = \frac{\exp\left(-\left\|x_i - x_j\right\|^2/2\sigma_i^2\right)}{\sum_{k\neq l}\exp(-\|x_k - x_l\|^2/2\sigma_k^2)} \tag{4}$$

where σ_i is the bandwidth of the Gaussian kernel around point x_i, set adaptively using a binary search to achieve a target perplexity (a user-specified parameter that approximately corresponds to the number of effective nearest neighbors). In the low-dimensional embedding space, similarities between points y_i and y_j are modeled using a Student *t*-distribution with one degree of freedom:

$$q_{ij} = \frac{\left(1 + \left\|y_i - y_j\right\|^2\right)^{-1}}{\sum_{k\neq l}(1 + \|y_k - y_l\|^2)^{-1}} \tag{5}$$

The use of a heavy-tailed *t*-distribution in the embedding space rather than a Gaussian addresses the "crowding problem": in low-dimensional spaces, there is

insufficient volume to accommodate the moderate-distance neighbors of high-dimensional data. The heavier tails of the t-distribution allow these points to be placed further apart in the embedding space, creating clearer visual separation between cluster structures. The t-SNE objective minimizes the Kullback–Leibler divergence between P and Q:

$$KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{6}$$

This objective is optimized using gradient descent, requiring $O(n^2)$ gradient computations per iteration. The Barnes-Hut approximation reduces this to $O(nlogn)$ by approximating the repulsive forces from distant points using a quad-tree or octree spatial data structure. Despite this improvement, t-SNE remains computationally intensive for large-scale datasets and does not provide a parametric mapping function applicable to new data points.

The gradient of the t-SNE KL divergence with respect to the embedding coordinates y_i has a simple and intuitive form that clarifies the optimization dynamics. The gradient consists of two terms: an attractive force pulling y_i toward similar points (those with high p_{ij}), and a repulsive force pushing y_i away from all other points (weighted by q_{ij}). The attractive forces ensure that local neighborhoods are preserved, while the repulsive forces prevent all points from collapsing to a single location. The use of the *t*-distribution for q_{ij} rather than a Gaussian amplifies the repulsive forces for distant pairs - the heavier tails of the t-distribution assign higher q_{ij} values to distant point pairs than a Gaussian would, increasing the gradient magnitude pushing them apart and creating clearer visual separation in the embedded space. This mechanical understanding of the t-SNE optimization clarifies why the algorithm tends to produce well-separated clusters even when the high-dimensional data does not have sharply defined cluster boundaries - the repulsive forces push cluster margins apart during optimization, exaggerating separations that may be subtle in the original space [29, 30].

3.3 Autoencoder Architecture and Training

This section presents a rigorous mathematical description of the stacked autoencoder architecture used in this study, following the notational conventions established in applied mathematics for deep neural networks [37]. The network is described layer by layer; the cost function is defined to be minimized and derived the backpropagation equations that enable efficient gradient computation.

3.3.1 Basic Building Blocks

The fundamental operation at each layer is the affine (linear plus bias) transformation of an input vector $a \in \mathbb{R}^n$:

$$z = Wa + b \tag{7}$$

where $W \in \mathbb{R}^{(m \times n)}$ is the weight matrix, $b \in \mathbb{R}^m$ is the bias vector, and $z \in \mathbb{R}^m$ is the weighted input (pre-activation). The number of rows m equals the number of neurons at the current layer; the number of columns n equals the number of neurons at the previous layer. The element w_{jk} specifies the connection strength from neuron *k* at the previous layer to neuron *j* at the current layer.

Batch Normalization

Batch Normalization (BN) is applied after each linear transformation and before the activation function. For a mini-batch $B = h^{(1)}, \ldots, h^{(m)}$ of pre-activation vectors at a given layer, BN normalizes each feature dimension j independently. The batch mean and variance are:

$$\mu_{Bj} = \frac{1}{m}\sum_{i=1}^{m} h_{ij} \tag{8}$$

$$\sigma_{Bj}^2 = \frac{1}{m}\sum_{i=1}^{m}\left(h_{ij} - \mu_{Bj}\right)^2 \tag{9}$$

The normalized value is then scaled and shifted by learnable parameters:

$$\widehat{h_j} = \frac{h_j - \mu_{Bj}}{\sqrt{\sigma_{Bj}^2 + \varepsilon}} \tag{10}$$

$$BN\left(h_j\right) = \gamma_j \widehat{h_j} + \beta_j \tag{11}$$

where γ_j and β_j are learnable scale and shift parameters, and $\varepsilon = 10^{-5}$ is a numerical stability constant. In vector notation, BN: $\mathbb{R}^n \rightarrow \mathbb{R}^n$ acts as $BN(h) = \gamma \odot \hat{h} + \beta$, where $\odot$ denotes the Hadamard (element-wise) product, and $\gamma, \beta \in \mathbb{R}^n$ are vectors of learnable parameters. The full parameter set of the autoencoder is therefore

$$\Theta = \{W^{[l]}, b^{[l]}, \gamma^{[l]}, \beta^{[l]}\}_{l=2}^{7} \tag{12}$$

Batch Normalization stabilizes and accelerates training by reducing internal covariate shift. During inference (embedding generation), BN uses running statistics μ_{run} and σ_{run}^2 accumulated during training rather than batch statistics, ensuring deterministic embeddings for any single subscriber.

Leaky ReLU activation

All hidden layers use the Leaky Rectified Linear Unit (LeakyReLU) as activation function, defined component-wise as:

$$\varphi(z_j) = \begin{cases} z_j & \text{if } z_j > 0 \\ \alpha z_j & \text{if } z_j \leq 0 \end{cases} \quad \alpha = 0.2 \tag{13}$$

Its derivative, required for backpropagation, is:

$$\varphi'(z_j) = \begin{cases} 1 & \text{if } z_j > 0 \\ \alpha & \text{if } z_j \leq 0 \end{cases} \tag{14}$$

Unlike standard ReLU (where $\varphi'(z_j) = 0$ for $z_j \leq 0$), LeakyReLU maintains a non-zero gradient for negative inputs. This prevents the dying-neuron phenomenon and ensures that all network parameters continue to receive gradient signals throughout training [26, 29].

Complete computational block

Combining the linear transformation (7), Batch Normalization (8-11), and LeakyReLU (13), the complete computational block at hidden layer *l* is:

$$a^{[l]} = \varphi\left(BN\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)\right) \tag{15}$$

This block receives the activation vector $a^{[l-1]}$ from the previous layer, applies an affine transformation, normalizes by the batch statistics, applies learnable scale and shift, and finally applies the LeakyReLU nonlinearity. The result $a^{[l]}$ is passed to the next layer.

3.3.2 General Network Notation

The notation follows Higham and Higham (2018) [76]. Suppose the network has *L* layers, with layer 1 the input layer and layer *L* the output layer. Let n_l denote the number of neurons at layer *l*. The notation $W^{[l]} \in \mathbb{R}^{n_l \times n_{l-1}}$ denotes the weight matrix at layer *l*, and $b^{[l]} \in \mathbb{R}^{n_l}$ to denote the bias vector. More precisely, $w_{jk}^{[l]}$ is the weight that neuron *j* at layer *l* applies to the output from neuron *k* at layer *l* - 1.

Given an input $x \in \mathbb{R}^{n_1}$, the activation $a^{[l]} \in \mathbb{R}^{n_l}$ at each layer is defined by the following recursion:

$$a^{[1]} = x \text{ (input layer)} \tag{16}$$

$$\mathrm{a}^{[\mathrm{l}]} = \varphi\left(BN\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)\right), \; l \in \{2,3,5,6\}, \text{ (hidden layers)} \tag{17}$$

$$a^{[4]} = W^{[4]}a^{[3]} + b^{[4]} \in \mathbb{R}^{288} \text{ (latent layer, no activation, no BN)} \tag{18}$$

$$a^{[7]} = W^{[7]}a^{[6]} + b^{[7]} \in \mathbb{R}^{948} \text{ (output layer, no activation, no BN)} \tag{19}$$

The weighted input (pre-activation) at each layer, which is essential for the backpropagation derivation, is defined as follows:

$$z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]} \in \mathbb{R}^{\mathbb{m}_{\mathbb{l}}}, \quad l = 2,3,\dots,7 \tag{20}$$

The layer widths are: n_1 = 948, n_2 = 1000, n_3 = 1000, n_4 = 288, n_5 = 1000, n_6 = 1000, n_7 = 948.

3.3.3 Stacked Autoencoder Architecture (Stacked AE)

The Stacked AE has L = 7 layers. The input is a subscriber feature vector $x \in \mathbb{R}^{948}$. The full forward pass is defined as a composition of three mappings:

$$F = f_{dec} \circ f_{lat} \circ f_{enc} \tag{21}$$

where $f_{\text{enc}}: \mathbb{R}^{948} \to \mathbb{R}^{1000}$ is the encoder, $f_{\text{lat}}: \mathbb{R}^{1000} \to \mathbb{R}^{288}$ is the latent layer, and $f_{\text{dec}}: \mathbb{R}^{288} \to \mathbb{R}^{948}$ is the decoder. The symbol $\circ$ denotes sequential function composition.

Encoder f_{enc}. The encoder applies two computational blocks of the form (15):

$$f_1^{(e)}: \mathbb{R}^{948} \to \mathbb{R}^{1000}, \quad x \mapsto h_1 = \varphi\left(BN\left(W_1^{(e)}x + b_1^{(e)}\right)\right), \quad W_1^{(e)} \in \mathbb{R}^{1000\times 948} \tag{22}$$

$$f_2^{(e)}: \mathbb{R}^{1000} \to \mathbb{R}^{1000}, \quad h_1 \mapsto h_2 = \varphi\left(BN\left(W_2^{(e)}h_1 + b_2^{(e)}\right)\right), \quad W_2^{(e)} \in \mathbb{R}^{1000\times 1000} \tag{23}$$

Thus $f_{\text{enc}}(x) = h_2 \in \mathbb{R}^{1000}$.

Latent layer f_{lat}. The bottleneck layer projects the hidden representation to the latent space without activation or Batch Normalization [77]:

$$f_{lat}: \mathbb{R}^{1000} \to \mathbb{R}^{288}, \quad h_2 \mapsto z = W^{(lat)}h_2 + b^{(lat)}, \quad W^{(lat)} \in \mathbb{R}^{288\times 1000} \tag{24}$$

The vector $z \in \mathbb{R}^{288}$ is the learned latent representation (embedding) of the subscriber. The absence of nonlinearity in this layer allows the embedding coordinates to take arbitrary real values, which is important for the metric properties required by cosine similarity computation downstream.

Decoder f_{dec}. The decoder mirrors the encoder structure symmetrically. The first two blocks apply the full computational block (15), while the final layer is a linear projection without activation:

$$g_1: \mathbb{R}^{288} \to \mathbb{R}^{1000}, \quad z \mapsto g_1 = \varphi\left(BN\left(W_1^{(d)}z + b_1^{(d)}\right)\right), \quad W_1^{(d)} \in \mathbb{R}^{1000\times 288} \tag{25}$$

$$g_2: \mathbb{R}^{1000} \to \mathbb{R}^{1000}, \quad g_1 \mapsto g_2 = \varphi\left(BN\left(W_2^{(d)} g_1 + b_2^{(d)}\right)\right), \quad W_2^{(d)} \in \mathbb{R}^{1000\times1000} \tag{26}$$

$$g_{out}: \mathbb{R}^{1000} \to \mathbb{R}^{948}, \quad g_2 \mapsto \hat{x} = W_3^{(d)} g_2 + b_3^{(d)}, \quad W_3^{(d)} \in \mathbb{R}^{948\times1000} \tag{27}$$

Thus $f_{\text{dec}}(z) = \hat{x} \in \mathbb{R}^{948}$.

Full forward pass. Combining all components, the complete autoencoder function is:

$$\hat{x} = F(x) = (f_{dec} \circ f_{lat} \circ f_{enc})(x) \in \mathbb{R}^{948} \tag{28}$$

The dimension sequence through the network is 948 → 1000 → 1000 → 288 → 1000 → 1000 → 948. The weight matrices have the following dimensions: $W_1^{(e)} \in \mathbb{R}^{1000\times948}$, $W_2^{(e)} \in \mathbb{R}^{1000\times1000}$, $W^{(\text{lat})} \in \mathbb{R}^{288\times1000}$, $W_1^{(d)} \in \mathbb{R}^{1000\times288}$, $W_2^{(d)} \in \mathbb{R}^{1000\times1000}$, $W_3^{(d)} \in \mathbb{R}^{948\times1000}$. The total number of trainable parameters (weights, biases, and BN parameters) is approximately 5.3×10^6.

The latent representation $z^{[4]} = W^{[4]}a^{[3]} + b^{[4]} \in \mathbb{R}^{288}$ constitutes the embedding of a subscriber *x*. This 288-dimensional vector is used for all downstream similarity computation and look-alike detection, Figure 3 illustrates the layer structure.

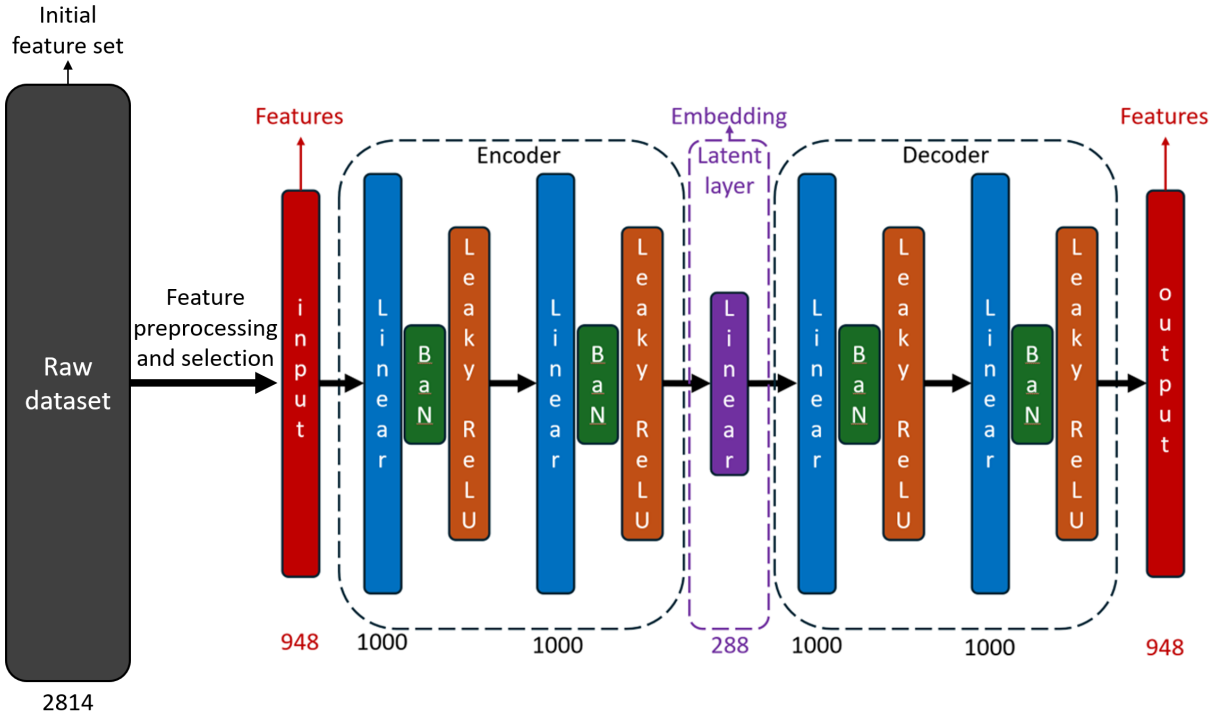

Figure 3 - Architecture of the stacked autoencoder: encoder (layers 1–4) mapping *x* to latent embedding $z^{[4]}$, and decoder (layers 4–7) reconstructing $\hat{x}$

3.3.4 Target Vector and Cost Function

In classification tasks, the target vector $y\left(x^{(i)}\right)$ is provided by external labels. In the autoencoder, no external labeling is required: the training target coincides with the input, so

$$y\left(x^{(i)}\right) = x^{(i)} \in \mathbb{R}^{948}, \quad i = 1,2,\dots,N \tag{29}$$

This means that training is self-supervised: the network learns to preserve subscriber feature information through compression and reconstruction without any manual labeling.

For a single observation $x^{(i)}$, the per-sample cost is the squared reconstruction error:

$$C_i = \frac{1}{2}\left\|x^{(i)} - F\left(x^{(i)}\right)\right\|^2 = \frac{1}{2}\sum_{j=1}^{948}\left(x_j^{(i)} - \widehat{x_j^{(i)}}\right)^2 \tag{30}$$

The factor 1/2 is included for computational convenience: it cancels with the factor of 2 that arises when differentiating the square. Following the quadratic cost function of Higham and Higham (2018) [76], the total cost over all N training subscribers is:

$$\text{Cost}(\Theta) = \frac{1}{N}\sum_{i=1}^{N} C_i = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}||x^{(i)} - F\left(x^{(i)};\Theta\right)||^2 \tag{31}$$

where $\Theta = \{W^{[l]}, b^{[l]}, \gamma^{[l]}, \beta^{[l]}\}_{l=2}^{7}$ is the full set of trainable parameters. Training consists of finding Θ^* that minimizes Cost(Θ):

$$\Theta^* = \arg\min_{\Theta} \text{Cost}(\Theta) \tag{32}$$

Because Cost(Θ) is a sum over individual training points, its gradient decomposes as:

$$\nabla_{\Theta}\,\text{Cost} = \frac{1}{N}\sum_{i=1}^{N}\nabla_{\Theta} C_i \tag{33}$$

This separability motivates the mini-batch stochastic gradient approach: at each step, the mean over all N samples is approximated by the mean over a randomly chosen mini-batch Ω_{batch} of size m = 512:

$$\nabla_{\Theta}\,\text{Cost} \approx \frac{1}{m}\sum_{i\in\Omega_{\text{batch}}}\nabla_{\Theta} C_i \tag{34}$$

This mini-batch approximation introduces noise into the gradient estimate, which is beneficial in practice: the stochastic noise helps escape local minima and saddle points in the non-convex loss landscape. The use of mini-batches is also essential for Batch Normalization, which requires a batch of sufficient size to compute meaningful statistical estimates.

3.3.5 Backpropagation (Lemma)

To compute the gradients $\nabla_\Theta C_i$ required by the mini-batch update (34), the backpropagation algorithm is applied. For a fixed training point $x^{(i)}$ the superscript is dropped and whole equation transformed using $C = (1/2)\left\|x - a^{[7]}\right\|^2$. The error (sensitivity) of neuron *j* at layer *l* is defined as:

$$\delta_j^{[l]} = \frac{\partial C}{\partial z_j^{[l]}}, \quad 1 \le j \le n_l, \quad 2 \le l \le 7 \tag{35}$$

The quantity $\delta_j^{[l]}$ measures how much the cost function changes when the weighted input of neuron *j* at layer *l* is perturbed. In vector form: $\delta_j^{[l]} \in \mathbb{R}^{n_l}$.

Lemma (Backpropagation for the StackedAE)

With the notation above, and using the Hadamard (element-wise) product ∘, the following results hold:

(i) Output layer error (l = 7, linear layer -- no activation):

$$\delta^{[7]} = a^{[7]} - x \tag{36}$$

Proof. Since the output layer is linear, $a^{[7]} = z^{[7]}$, so $da_j^{[7]}/dz_j^{[7]} = 1$. From the cost function:

$$\frac{\partial C}{\partial a_j^{[7]}} = a_j^{[7]} - x_j \tag{37}$$

Applying the chain rule:

$$\delta_j^{[7]} = \frac{\partial C}{\partial z_j^{[7]}} = \frac{\partial C}{\partial a_j^{[7]}} \cdot \frac{\partial a_j^{[7]}}{\partial z_j^{[7]}} = \left(a_j^{[7]} - x_j\right) \cdot 1 = a_j^{[7]} - x_j \tag{38}$$

In vector form: $\delta^{[7]} = a^{[7]} - x$. ■

(ii) Hidden layer errors with activation ($l \in \{6, 5, 4, 2\}$):

$$\delta^{[l]} = \varphi'\left(z^{[l]}\right) \odot \left(W^{[l+1]}\right)^T \delta^{[l+1]} \tag{39}$$

Proof. By the chain rule, passing from layer l to layer l+1:

$$\delta_j^{[l]} = \frac{\partial C}{\partial z_j^{[l]}} = \sum_k \frac{\partial C}{\partial z_k^{[l+1]}} \cdot \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} = \sum_k \delta_k^{[l+1]} \cdot \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} \tag{40}$$

From the definition of the weighted input $z_k^{[l+1]} = \sum_s w_{ks}^{[l+1]} a_s^{[l]} + b_k^{[l+1]}$ and the relation $a_j^{[l]} = \varphi\left(\mathrm{BN}\left(z_j^{[l]}\right)\right)$:

$$\frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} = w_{kj}^{[l+1]} \cdot \varphi'\left(z_j^{[l]}\right) \tag{41}$$

Substituting (41) into (40):

$$\delta_j^{[l]} = \varphi'\left(z_j^{[l]}\right) \cdot \sum_k w_{kj}^{[l+1]}\ \delta_k^{[l+1]} = \varphi'\left(z_j^{[l]}\right) \cdot \left[\left(W^{[l+1]}\right)^{\mathrm{T}} \delta^{[l+1]}\right]_j \tag{42}$$

In vector form: $\delta^{[l]} = \varphi'\left(z^{[l]}\right) \circ \left(W^{[l+1]}\right)^T \delta^{[l+1]}$. ■

(iii) Latent layer error (l = 4, linear layer - no activation):

$$\delta^{[4]} = \left(W^{[5]}\right)^T \delta^{[5]} \tag{43}$$

Proof. The latent layer is linear: $a^{[4]} = z^{[4]}$, so $da_j^{[4]}/dz_j^{[4]} = 1$. By the same chain rule argument as above, with the identity replacing φ':

$$\delta_j^{[4]} = \sum_k \delta_k^{[5]} \cdot w_{kj}^{[5]} \cdot 1 = \left[\left(W^{[5]}\right)^{\mathrm{T}} \delta^{[5]}\right]_j \tag{44}$$

In vector form: $\delta^{[4]} = \left(W^{[5]}\right)^T \delta^{[5]}$. ■

(iv) Gradients with respect to biases and weights:

$$\frac{\partial C}{\partial b_j^{[l]}} = \delta_j^{[l]}, \quad 2 \le l \le 7 \tag{45}$$

$$\frac{\partial C}{\partial w_{jk}^{[l]}} = \delta_j^{[l]} \cdot a_k^{[l-1]}, \quad 2 \le l \le 7 \tag{46}$$

Proof of (45). From the definition $z_j^{[l]} = \sum_k w_{jk}^{[l]} a_k^{[l-1]} + b_j^{[l]}$, it follows that $\frac{\partial z_j^{[l]}}{\partial b_j^{[l]}} = 1$. Therefore:

$$\frac{\partial C}{\partial b_j^{[l]}} = \frac{\partial C}{\partial z_j^{[l]}} \cdot \frac{\partial z_j^{[l]}}{\partial b_j^{[l]}} = \delta_j^{[l]} \cdot 1 = \delta_j^{[l]} \ \blacksquare \tag{47}$$

Proof of (46). Similarly, $dz_j^{[l]}/dw_{jk}^{[l]} = a_k^{[l-1]}$. Therefore:

$$\frac{\partial C}{\partial w_{jk}^{[l]}} = \sum_s \frac{\partial C}{\partial z_s^{[l]}} \cdot \frac{\partial z_s^{[l]}}{\partial w_{jk}^{[l]}} = \delta_j^{[l]} \cdot a_k^{[l-1]} \ \blacksquare \tag{48}$$

In matrix form: $\frac{\partial C}{\partial W^{[l]}} = \delta^{[l]} \left(a^{[l-1]}\right)^T \in \mathbb{R}^{n_l \times n_{l-1}}$.

The Lemma shows that all required partial derivatives can be computed in two passes. The forward pass (equations 16-20) computes the activations $a^{[l]}$ and weighted inputs $z^{[l]}$ in order l = 1, ..., 7. The backward pass (equations 36-44) computes the errors $\delta^{[l]}$ in reverse order l = 7, ..., 2. Equations (45-46) then give the complete gradient at the cost of one forward pass and one backward pass per training point.

The explicit gradient formulas for the output and latent layers are:

$$\frac{\partial C}{\partial W^{[7]}} = \delta^{[7]} \left(a^{[6]}\right)^\top = \left(a^{[7]} - x\right) \left(a^{[6]}\right)^\top \in \mathbb{R}^{948 \times 1000} \tag{49}$$

$$\frac{\partial C}{\partial b^{[7]}} = \delta^{[7]} = a^{[7]} - x \in \mathbb{R}^{948} \tag{50}$$

For the latent layer:

$$\frac{\partial C}{\partial W^{(\text{lat})}} = \delta^{[4]} \left(a^{[3]}\right)^\top \in \mathbb{R}^{288 \times 1000} \tag{51}$$

$$\frac{\partial C}{\partial b^{(\text{lat})}} = \delta^{[4]} \in \mathbb{R}^{288} \tag{52}$$

These formulas make explicit that updating any weight requires only the error signal $\delta^{[l]}$ of the current layer and the activation $a^{[l-1]}$ of the previous layer.

3.3.6 Adam Optimizer and Training Protocol

The mini-batch gradient (34) is used by the Adam optimizer to update Θ [78]. Adam maintains first-moment vector m_t and second-moment vector v_t estimates of the gradient and updates parameters at step t as:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\Theta \mathrm{Cost}_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\Theta \mathrm{Cost}_t)^2 \tag{53}$$

$$\Theta_{t+1} = \Theta_t - \eta \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \varepsilon}, \qquad \widehat{m_t} = \frac{m_t}{1-\beta_1^t}, \qquad \widehat{v_t} = \frac{v_t}{1-\beta_2^t} \tag{54}$$

with $\beta_1 = 0.9, \beta_2 = 0.999, \eta = 0.001, \varepsilon = 10^{-8}$. The bias-corrected estimates $\widehat{m_t}$ and $\widehat{v_t}$ compensate for the initialization bias in the running averages during early training steps. An L2 regularization term $\lambda\|\Theta\|^2$ with $\lambda = 10^{-4}$ is added to the cost function to penalize large weights and reduce overfitting [79]. Training runs for 400 epochs; as illustrated in Figure 4, the validation loss stabilizes after approximately 350-400 epochs, reaching a final MSE of 0.61.

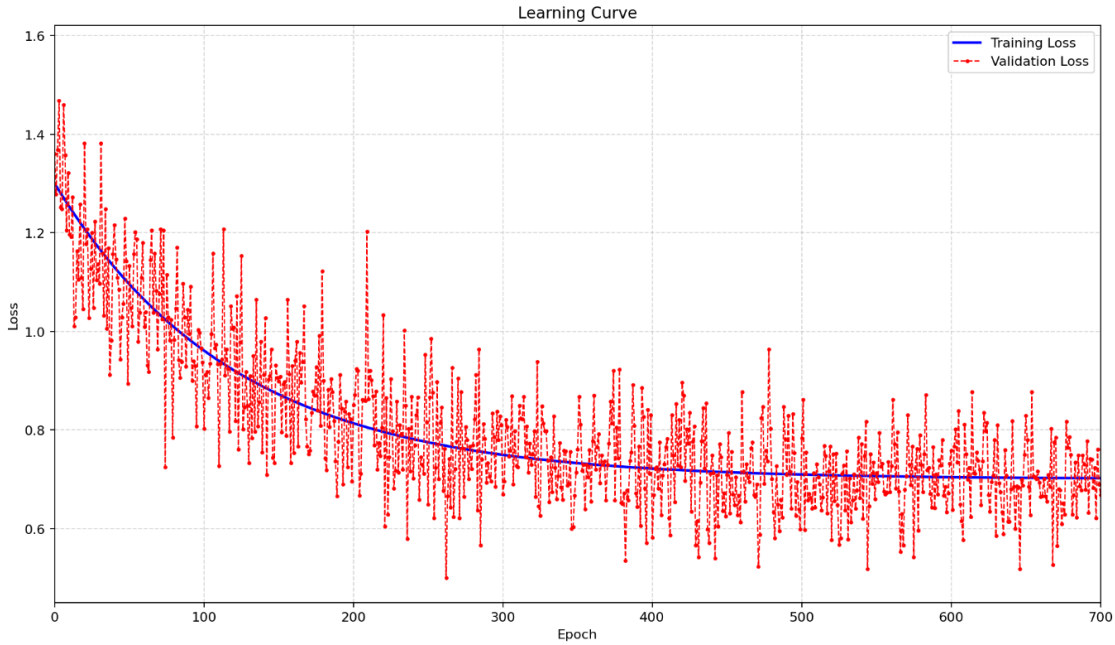

Figure 4 – Learning curves of the autoencoder: training and validation MSE vs. number of epochs

Table 2 summarizes the complete hyperparameter configuration.

Table 2 – Stacked autoencoder hyperparameter configuration

Hyperparameter	Value
Number of layers L	7
Layer widths [n_1, ..., n_7]	[948, 1000, 1000, 288, 1000, 1000, 948]

Continuation of table 2

Latent dimensionality k = n_4	288
Activation σ( ) (hidden layers)	LeakyReLU (α = 0.2)
Activation (latent + output layers)	Identity
Batch Normalization	After each linear layer, ε = 10^{-5}
Cost function	Mean Squared Error (MSE)
Optimizer	Adam, $\beta_1 = 0.9$, $\beta_2 = 0.999$
Learning rate η	0.001 (ReduceLROnPlateau schedule)
L2 regularization λ	0.0001 (weight decay)
Mini-batch size m	512
Training epochs	400
Total parameters |Θ|	$\approx 5.3 \times 10^6$
Final validation MSE	0.61

3.4 Multi-Entity Embedding Strategy

The dataset contains six heterogeneous groups of features describing distinct aspects of user behavior. Instead of training separate models for each domain, all 948 features are concatenated into a single unified input representation. A shared autoencoder is then trained on this combined feature space, enabling the model to capture cross-domain interactions and latent dependencies across different aspects of user behavior within a single representation.

The proposed embedding strategy is illustrated in Figure 5. The six behavioral domains - User, Web, Finance, Device, Base Station, and Tariff Plan - are first combined into a single feature vector, which is then processed by a unified autoencoder. The resulting latent vector z_{user} of 288 dimensions is used directly for cosine similarity-based look-alike retrieval without any task-specific retraining. This unified modeling approach ensures that both intra-domain patterns and inter-domain relationships are jointly learned, providing a more holistic representation of user behavior.

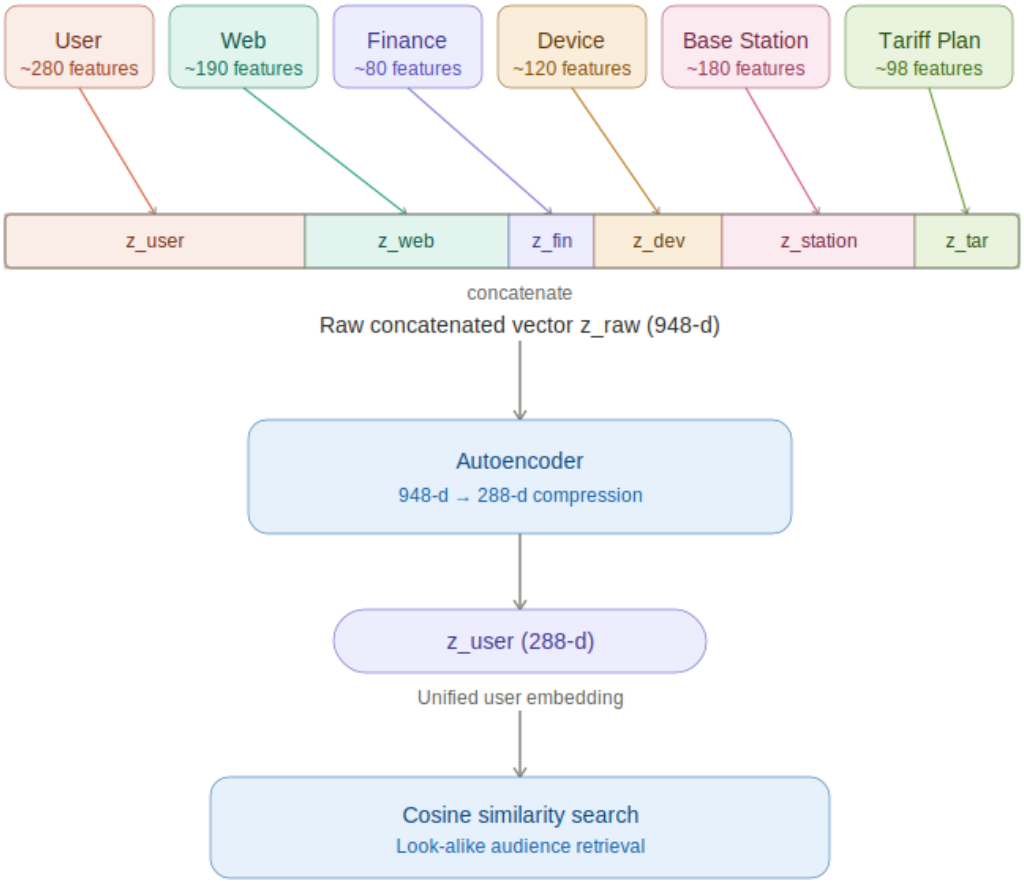

Figure 5 - Multi-entity embedding strategy

For the combined feature space {User, Web, Finance, Device, BaseStation, TariffPlan}, a single autoencoder is trained on the unified input vector $x \in \mathbb{R}^{948}$, obtained by concatenating features from all behavioral domains. The encoder produces a latent representation $z_{user} = a^{[4]}(x) \in \mathbb{R}^{288}$ via the encoder (equations 22-24). which serves as the unified subscriber embedding for downstream similarity-based tasks.

$$z_{\text{user}} = \left[z^{(\text{User})};\ z^{(\text{Web})};\ z^{(\text{Fin})};\ z^{(\text{Dev})};\ z^{(\text{BS})};\ z^{(\text{Tariff})}\right] \tag{55}$$

where [;] denotes vector concatenation. The multi-entity strategy offers several important practical advantages: independence of update (entity-specific data can be reconsidered independently when new data becomes available), modular deployment, and interpretable structure (the concatenated vector has a natural partition into entity-specific subspaces). The experimental validation in Chapter 6 demonstrates the importance of this strategy: single-entity embeddings achieve a Lift Top 1 of 7.3, while multi-entity concatenated embedding improve this to 11.7 - a 60% improvement - confirming that different behavioral domains provide genuinely complementary information about user similarity.

3.5 Cosine Similarity for User Matching

After training, the encoder and latent layer are used to generate the embedding for each subscriber. The decoder is discarded. For subscriber *i*, the embedding is:

$$z^{(i)} = f_{\text{lat}}\left(f_{\text{enc}}\left(x^{(i)}\right)\right) \in \mathbb{R}^{288} \tag{56}$$

Look-alike detection is performed by computing the cosine similarity between the seed audience and all subscribers in the database. Cosine similarity measures the angular distance between two vectors:

$$\text{sim}(u, v) = \frac{\langle u, v \rangle}{||u||_2 \cdot ||v||_2} \in [-1,\ 1] \tag{57}$$

where $<u, v>$ denotes the dot product and $||\cdot||_2$ the Euclidean norm. Cosine similarity is invariant to the scale of vectors - a property that is particularly important for telecommunications subscriber data, where high-activity and low-activity subscribers may have similar behavioral patterns but different total usage volumes.

Let S denote the set of positive seed subscribers and $U = \{1,...,N\} \setminus S$ the remaining subscriber base. For each candidate j in U, the aggregated similarity score with the seed is:

$$\text{Score}(j) = \max_{s \in S} \text{sim}\left(z^{(s)}, z^{(j)}\right), \quad j \in U \tag{58}$$

Subscribers are ranked by decreasing Score(j). The final look-alike audience T is formed as the K subscribers with the highest scores:

$$T = \{j \in U : \text{Score}(j) \text{ is in the top-}K\} \tag{59}$$

Preliminary experiments comparing cosine similarity with Euclidean distance showed that cosine similarity consistently produced higher Lift Top 1 scores (11.7 vs 10.4) on the validation campaigns, confirming that the magnitude-invariance property of cosine similarity provides a genuine advantage for this task.

Subscribers are ranked by decreasing cosine similarity to the seed audience, and a threshold is applied to select the final look-alike audience. The optimal threshold is determined for each campaign using the Precision-Recall curve on a held-out portion of the validation data. In practice, B2B clients typically specify a desired audience size, and the threshold is set automatically to achieve the requested count.

3.6 Summary

This chapter presented the complete mathematical formulations of the three dimensionality reduction approaches evaluated in this study. For the proposed stacked autoencoder, the description covered: (i) the basic building blocks - linear

transformation (7), Batch Normalization (8-12), and Leaky ReLU activation (13-14); (ii) the full layer-by-layer architecture with the composition-of-mappings notation f_{enc}, f_{lat}, f_{dec} and the recursive notation $a^{[l]}$ (equations 16-20); (iii) the self-supervised training target y(x) = x (29) and the MSE cost function (31); (iv) the mini-batch stochastic gradient formulation (34); and (v) the complete backpropagation Lemma (equations 36-48) including step-by-step proofs for the output layer, hidden layers, latent layer, and parameter gradients. The multi-entity embedding strategy (55) and cosine similarity-based look-alike detection procedure (57-59) were also described. The following chapters describe the Siamese network extension and the production system architecture.

4 SIAMESE NETWORK FOR ADAPTIVE SIMILARITY LEARNING

While cosine similarity in the latent space provides an effective and computationally efficient mechanism for look-alike audience detection, it applies a uniform geometric distance to all dimensions of the embedding space. A Siamese network extends this framework by learning an adaptive similarity function that can weight different embedding dimensions based on their relevance for distinguishing similar from dissimilar users in the training data. This chapter describes the architecture, training procedure, and computational properties of the Siamese network component of the proposed framework.

Similarity learning - the problem of learning a function that measures the similarity between pairs of objects - occupies a central position in modern machine learning. The field has evolved from simple distance metric learning approaches, where a linear transformation of the feature space is optimized to bring similar pairs closer and push dissimilar pairs apart (Mahalanobis metric learning, LMNN), to deep metric learning approaches that apply neural networks to learn nonlinear similarity functions from labeled pairs or triplets. Siamese networks represent one of the most widely adopted deep metric learning architectures due to their conceptual simplicity - the same network is applied to both inputs, ensuring symmetric similarity assessment - and their strong empirical performance across diverse similarity learning benchmarks [55, 56].

The integration of Siamese networks with pre-trained autoencoder embeddings is a two-stage representation learning approach. In the first stage, the autoencoder learns a general-purpose behavioral representation through reconstruction-based training, capturing the essential structure of user behavior without any task-specific objective. In the second stage, the Siamese network fine-tunes the similarity assessment within the autoencoder embedding space using task-specific labeled pairs from real advertising campaigns. This two-stage approach offers an important advantage: the autoencoder can be trained once on the full subscriber database using only the self-supervised reconstruction objective, and the Siamese network can be trained separately using labeled data from individual campaigns without modifying the underlying behavioral representations. This decoupling means that new Siamese networks can be trained for new campaign types or product categories without retraining the computationally expensive autoencoder, making the system highly efficient in terms of total training cost [2, 45].

4.1 Architecture and Design

The Siamese network architecture used in this study consists of two identical subnetworks - twin branches - that share all weights and parameters. Each branch receives a 288-dimensional user embedding produced by the autoencoder and processes it through a sequence of fully connected layers to produce a fixed-length output embedding. The outputs of the two branches are then compared using a distance function to produce a similarity score.

Each twin branch consists of five fully connected layers with widths 256, 512, 224, 160, and 128, each followed by Batch Normalization and ReLU activation. A

dropout layer with dropout probability p = 0.3 is applied before the final dense layer to provide regularization and prevent overfitting. The output of each branch is a 128-dimensional vector representing the processed embedding of the corresponding input user. The similarity between two input users is then computed as the Euclidean distance between their processed embeddings:

$$d(u_1, u_2) = ||f_S(z_{u_1}) - f_S(z_{u_2})||_2 \tag{60}$$

where f_S denotes the shared twin network function and z_u denotes the autoencoder embedding of user u. Small values of d indicate that the two users are similar according to the learned similarity function; large values indicate dissimilarity. Figure 6 presents the complete Siamese network architecture.

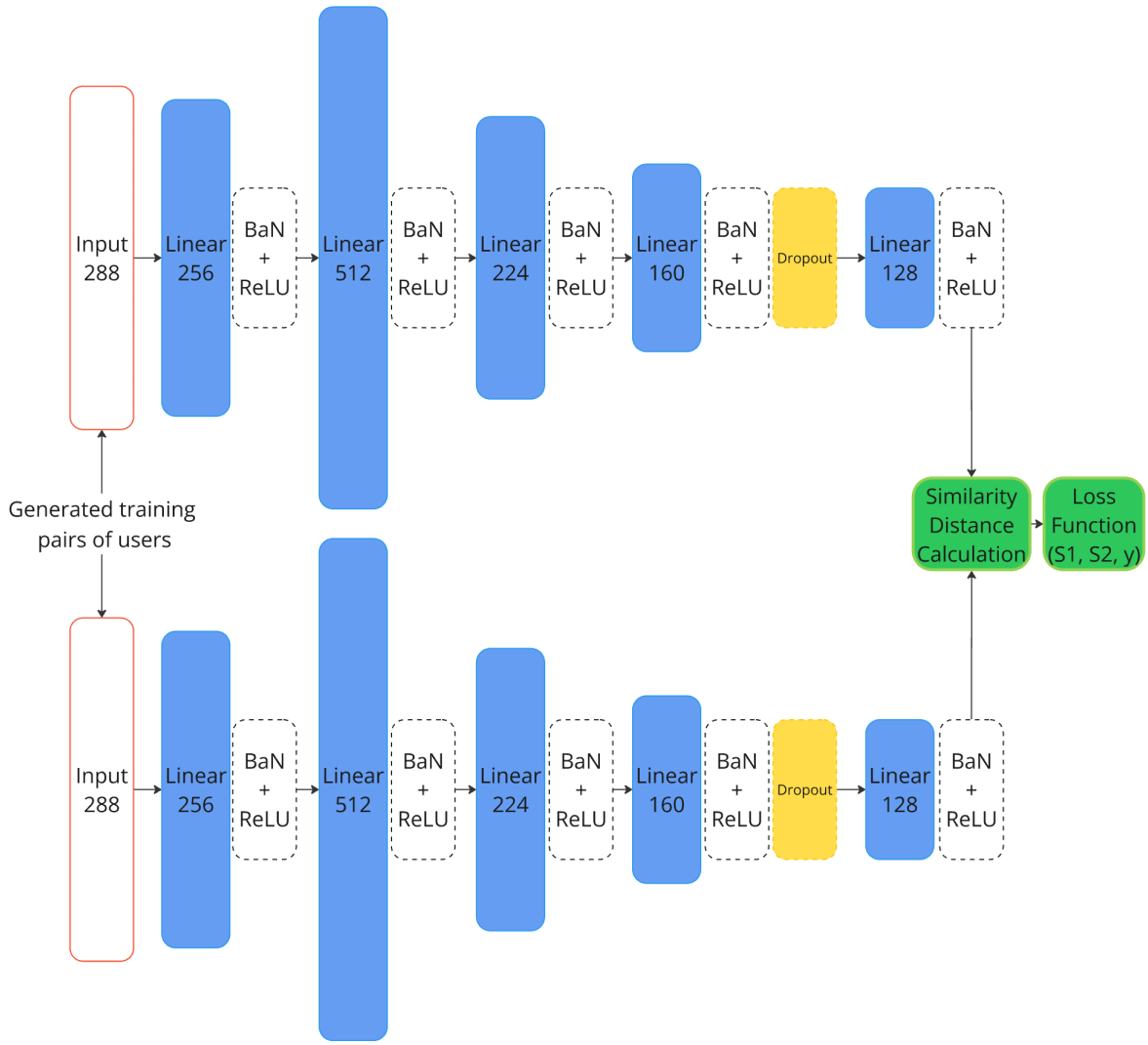

Figure 6 – Architecture of the Siamese neural network for user similarity estimation

The width sequence [256, 512, 224, 160, 128] was determined through systematic architecture search, evaluating combinations of layer widths and depths on the validation set. The initial expansion to 512 dimensions (from the 288-dimensional input) allows the network to create a richer intermediate representation before progressively compressing to the 128-dimensional output. This non-monotonic width

profile - first expanding then contracting - has been shown to be effective for learning similarity functions in several domains, as the expansion phase enables the network to learn complex feature interactions before the contraction phase produces a compact discriminative representation [40].

ReLU activation was used in the Siamese network branches instead of the LeakyReLU [7] used in the autoencoder, as the 128-dimensional output space of the Siamese network provides sufficient capacity to represent the relevant similarity distinctions even with zero activations for some units. The weight-sharing constraint between the two branches is enforced by using a single set of parameter tensors for both branches during both forward and backward passes, ensuring that the network learns a truly symmetric similarity function.

4.2 Training Procedure and Pair Construction

Training the Siamese network requires labeled pairs of similar (positive) and dissimilar (negative) users. For look-alike modeling, positive pairs are defined as pairs of users who both belong to the same reference seed audience: if user u_1 and user u_2 are both in the seed audience for a given campaign, they are considered similar and constitute a positive training pair. Negative pairs are formed by pairing each seed user with a randomly selected non-seed subscriber from the database.

Positive pairs were constructed using the Cartesian product of all seed users in each training campaign: for a seed audience of m users, this produces m(m-1)/2 positive pairs, ensuring that all available positive similarity relationships are fully utilized. This exhaustive pairing strategy maximizes the information available for learning the positive similarity structure but can produce a large number of pairs for large seed audiences. To maintain computational tractability, a maximum of 10,000 positive pairs per campaign was set, with random subsampling applied when this limit was exceeded [2, 80].

For negative pairs, random negative sampling was applied: each seed user was paired with randomly selected non-seed subscribers from the full subscriber database. The training dataset followed a class distribution of 10% positive pairs to 90% negative pairs, consistent with standard practice in contrastive learning. This imbalanced ratio reflects the real-world distribution - in a database of hundreds of thousands of subscribers, genuine look-alikes constitute a small fraction - and prevents the model from treating all user pairs as equally likely to be dissimilar [2].

The Siamese network was trained using contrastive loss [61], which directly optimizes the distance between positive pairs (minimizing it) and negative pairs (maximizing it up to a margin):

$$L = y \cdot d^2 + (1 - y) \cdot \max(0, \text{margin} - d)^2 \tag{61}$$

where y = 1 for positive pairs (same class) and y = 0 for negative pairs (different classes), d is the Euclidean distance between the output embeddings of the two branches, and margin is a hyperparameter that defines the minimum desired distance between negative pairs. The margin was set to 1.0 based on grid search over values in

[0.5, 1.0, 1.5, 2.0]. The Adam optimizer was used with a learning rate of 0.001 and batch size of 512, consistent with the autoencoder training configuration.

Hard negative mining represents an alternative to random negative sampling that selects negative pairs where the current model is most confused, pairs where the network currently assigns a high similarity score despite them belonging to different classes. While hard negative mining can accelerate convergence in some settings, it was found to be counterproductive in the early stages of training for this task. Before the network has learned a stable feature space, the hardest negatives tend to be noisy or ambiguous examples that provide misleading gradient signals. Random negative sampling, by contrast, provides a diverse and unbiased sample of the true negative distribution throughout training, producing more stable and generalizable representations [54, 80].

The decision to use Euclidean distance rather than cosine similarity as the comparison function in the Siamese network output was motivated by the properties of the contrastive loss function [61]. Contrastive loss naturally operates in terms of absolute distances: it minimizes the Euclidean distance for positive pairs and maximizes it for negative pairs up to a specified margin. The margin hyperparameter has a natural geometric interpretation as the minimum distance threshold that separates similar from dissimilar users in the learned similarity space. Cosine similarity, being a bounded metric in $[-1, 1]$, does not interact as cleanly with the contrastive loss margin, requiring additional normalization steps that introduce complexity without clear benefit [56, 81].

The decision to use Euclidean distance rather than cosine similarity as the comparison function in the Siamese network output was motivated by the properties of the contrastive loss [61] function. Contrastive loss naturally operates in terms of absolute distances: it minimizes the Euclidean distance for positive pairs and maximizes it for negative pairs up to a specified margin. The margin hyperparameter has a natural geometric interpretation as the minimum distance threshold that separates similar from dissimilar users in the learned similarity space. Cosine similarity, being a bounded metric in $[-1, 1]$, does not interact as cleanly with the contrastive loss [61] margin, requiring additional normalization steps that introduce complexity without clear benefit [81].

The learning rate schedule for Siamese network training followed the same ReduceLROnPlateau strategy used for the autoencoder. The initial learning rate of 0.001 was reduced by a factor of 0.96 whenever the validation contrastive loss failed to improve for one consecutive epoch, with a minimum learning rate floor of 10^{-5}. The validation set for Siamese network training consisted of a held-out subset of positive and negative pairs from campaigns not included in the training set, ensuring that the learning rate reduction is driven by genuine generalization performance rather than fitting to the training campaigns. This out-of-training-campaign validation approach is essential for ensuring that the Siamese network learns a generalizable similarity function rather than overfitting to the specific characteristics of the training seed audiences [2, 81].

The batch construction strategy for Siamese network training deserves careful consideration. Rather than constructing pairs uniformly across all training campaigns,

pairs were sampled in proportion to the number of available positive pairs per campaign - campaigns with larger seed audiences contribute more pairs to each training batch. This proportional sampling ensures that the Siamese network is exposed to sufficient examples from each campaign type during training, preventing the model from being dominated by the few campaigns with the largest seed audiences. An alternative curriculum learning approach - starting with easy pairs and gradually introducing harder pairs - was explored but found to provide no consistent benefit over proportional uniform sampling, consistent with the finding that random negative sampling outperforms hard negative mining for this task [48].

4.3 Computational Considerations

The computational cost of training a Siamese network is significantly higher than training traditional classifiers or computing cosine similarity directly. Unlike methods that process individual samples, the Siamese network must process paired inputs, multiplying the effective dataset size by the number of training pairs. For a seed audience of 1,000 users and a pair ratio of 1:9 (positive:negative), this generates approximately 100,000 training pairs per campaign, each requiring a forward and backward pass through both branches of the twin network.

This computational overhead is manageable at training time with GPU acceleration: the full Siamese network training over all 17 campaign datasets required approximately 6 hours on a single GPU, compared to minutes for the autoencoder training. At inference time, however, the Siamese network is computationally efficient: user embeddings are precomputed once by the autoencoder and stored, and similarity scoring reduces to evaluating the twin network on pairs of precomputed embeddings. For a look-alike search against a database of 900,000 subscribers, this requires 900,000 forward passes through a single branch of the Siamese network, which can be executed in parallel on GPU hardware in under one minute.

A further computational benefit of the proposed framework arises from the dimensionality reduction achieved by the autoencoder encoder. As illustrated in Figure 7, similarity search in the original 948-dimensional raw feature space requires $O(N \times 948)$ floating-point operations per query, where N is the number of subscribers in the database. By compressing each subscriber representation to a 288-dimensional latent vector, the autoencoder reduces this complexity to $O(N \times 288)$ - a theoretical reduction factor of 3.3×. In practice, the lower dimensionality additionally improves GPU parallel efficiency by reducing memory bandwidth pressure and enabling significantly larger effective batch sizes during inference, as more embedding vectors fit simultaneously into GPU VRAM.

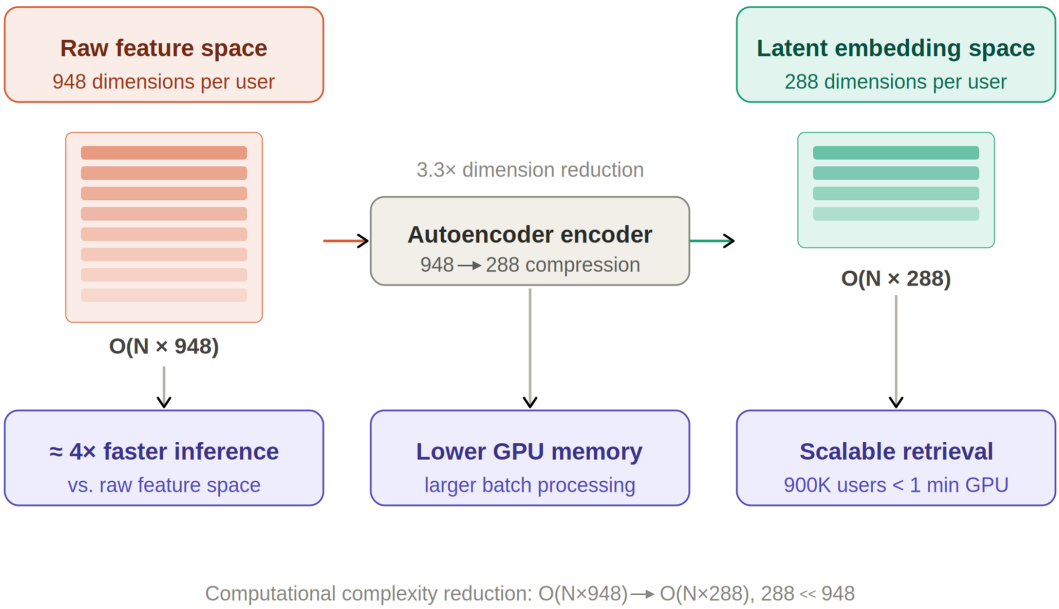

Figure 7 - Computational acceleration achieved by operating in the autoencoder latent space

The trade-off between training cost and inference efficiency makes Siamese networks practically viable for large-scale look-alike applications where high accuracy is prioritized over training simplicity. Organizations that already operate GPU infrastructure for existing machine learning workloads can integrate Siamese network training into existing pipelines with manageable additional computational requirements. The modular architecture described in Chapter 5 allows Siamese networks to be trained independently of the autoencoder models and updated on a monthly schedule aligned with the campaign data refresh cycle.

A key design consideration in the Siamese network architecture is the choice of network depth and width for the twin branches. Shallow networks with fewer parameters may underfit the complexity of the similarity structure, while very deep networks risk overfitting to the specific characteristics of the training campaign seed audiences rather than learning a generalizable similarity function. The five-layer architecture [256, 512, 224, 160, 128] was selected after evaluating architectures ranging from two to seven layers. Three-layer architectures [256, 128, 64] produced lower Lift Top 1 (11.9) due to insufficient representational capacity. Seven-layer architectures showed signs of overfitting, with validation Lift Top 1 declining after 50 training epochs. The five-layer design represents the optimal depth for this dataset, with sufficient capacity to learn complex similarity relationships while maintaining good generalization to new campaign seed audiences not seen during training.

The weight-sharing mechanism between the two Siamese branches requires special attention during implementation. In PyTorch, weight sharing is implemented naturally by using a single Module instance as both branches - when the module's forward method is called with different inputs, PyTorch computes separate activations for each input but uses the same parameter tensors. During backpropagation, gradients from both branches are accumulated in the shared parameter tensors before the optimizer step, effectively doubling the gradient signal compared to a non-shared

architecture. This gradient accumulation is a feature rather than a bug: it means that each parameter update incorporates information from both users in each training pair, enabling faster convergence of the shared representation [55, 56].

The evaluation of the Siamese network on unseen campaign seed audiences - audiences not used in any training pair - is a critical test of generalization. For the 17 campaigns used in evaluation, 12 were drawn from the same product categories as those used in training (financial products, device upgrades, digital services), while 5 represented product categories not represented in the training set (travel services, premium entertainment subscriptions, health-related digital products). The Siamese network achieved slightly lower Lift Top 1 on the novel-category campaigns (average 12.1 vs 13.2 for familiar categories), confirming that the learned similarity function generalizes well but with some performance degradation for entirely new product types. This degradation is expected and acceptable: the cosine similarity baseline also shows a similar pattern, with multi-entity embedding providing consistent improvements across both familiar and novel campaign categories.

4.4 Summary

This chapter described the Siamese network architecture - consisting of two identical five-layer branches with dimensions [256, 512, 224, 160, 128] sharing weights - trained on positive and negative user pairs constructed from seed audiences of advertising campaigns. The network learns an adaptive similarity function in the 288-dimensional autoencoder embedding space that outperforms cosine similarity across all evaluation metrics, achieving Lift Top 1 = 12.9 vs 11.7 for cosine similarity. The training procedure uses contrastive loss with a margin of 1.0 and a 10:90 positive-to-negative pair ratio, consistent with contrastive learning best practices [81].

5 MODULAR SYSTEM ARCHITECTURE

To support scalable deployment in real-world data environments, the proposed representation learning framework was implemented using a modular system architecture that integrates data storage, preprocessing, model training, embedding generation, similarity computation, and evaluation into a unified pipeline. This chapter describes the architectural design of the system, the data storage and versioning infrastructure, and the production inference modes that support large-scale subscriber processing.

The design principles guiding the system architecture were selected based on the operational requirements of a production telecommunications data platform: scalability to hundreds of thousands of subscribers, reproducibility of experiments to enable model comparison and auditing, fault tolerance to handle individual component failures without disrupting the entire pipeline, and modularity to support independent development and deployment of individual system components. These principles are well-established in the MLOps literature as essential properties of production-grade machine learning systems and were prioritized throughout the architectural design process [12].

5.1 Overall System Design

The system architecture follows a modular design philosophy where different components of the pipeline are developed, tested, and updated independently while maintaining well-defined interfaces between modules. This design supports independent evolution of individual system components: new data sources can be integrated into existing entity definitions, and the similarity scoring mechanism can be upgraded without retraining the underlying embeddings. Figure 8 presents the overall architecture of the embedding-based look-alike modeling framework.

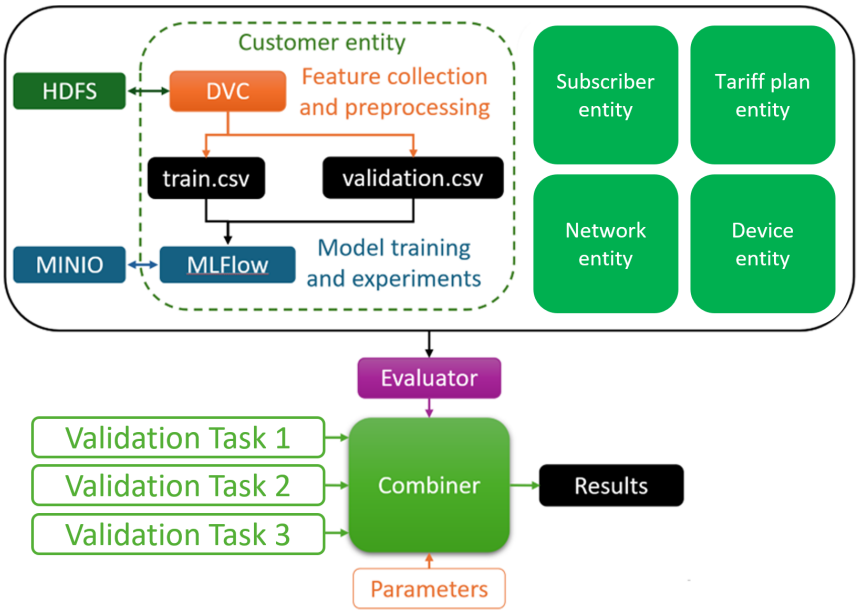

Figure 8 – Modular architecture of the embedding-based look-alike modeling framework

The system consists of five main components: data ingestion and preprocessing, entity feature engineering, autoencoder training and embedding generation, similarity scoring and look-alike detection, and evaluation and monitoring. Data flow between components is managed through the HDFS file system, with structured Parquet files serving as the standard interchange format between pipeline stages. This design enables parallel processing of different entity domains and supports incremental updates when new data becomes available for a subset of entities.

The preprocessing component handles the five-stage pipeline described in Chapter 2: categorical encoding, missing value imputation, outlier detection, multicollinearity reduction, and feature scaling. These operations are implemented using Apache Spark [75] to enable distributed processing of the full subscriber database within the time constraints of the monthly data refresh schedule. Preprocessing scripts are parameterized through configuration files that specify correlation thresholds, vocabulary sizes for categorical features, and outlier detection sensitivity.

The interface contract between pipeline components is defined in terms of standardized Parquet schemas with explicit column naming conventions, data type specifications, and nullable field policies. This contract-based interface design means that the output of one component can be consumed by any compatible downstream component regardless of the implementation details of either. For example, the preprocessing component can be replaced with a different implementation (e.g., moving from PySpark to a distributed SQL engine) without modifying the autoencoder training component, as long as the output schema remains unchanged. This interface stability reduces the coordination overhead between teams responsible for different parts of the pipeline and enables independent testing of individual components against the contract specification [12].

Error handling and recovery mechanisms are built into each pipeline component to ensure that transient failures - such as node failures in the Hadoop cluster or temporary unavailability of storage services - do not require full pipeline restarts. Each component implements idempotent processing: if a component is interrupted and restarted, it produces the same output as if it had completed successfully on the first attempt. This idempotency is achieved through atomic writes to HDFS (where files are first written to a temporary location and then atomically moved to the target location upon successful completion) and checkpointing mechanisms that allow partially completed runs to resume from the last successful checkpoint rather than starting over from the beginning. These engineering practices reduce the operational overhead of managing the monthly pipeline execution and ensure reliable delivery of updated embeddings on schedule.

5.2 Data Storage and Versioning

At the data storage level, the Hadoop Distributed File System (HDFS) serves as the primary repository for raw subscriber data, processed feature datasets, trained model checkpoints, precomputed embeddings, and evaluation results. HDFS provides fault-tolerant distributed storage across multiple nodes, enabling reliable access to

large datasets and seamless scaling of storage capacity as the subscriber base grows [12].

Feature datasets are versioned using Data Version Control (DVC), which extends standard Git version control to support large binary files stored in external storage systems. DVC tracks the dependencies between data artifacts and the scripts that produce them, enabling reproducibility of experiments: given the same code and DVC configuration, any prior experiment can be exactly reproduced. This is particularly important for a long-running system where multiple versions of preprocessing scripts, feature vocabularies, and model configurations coexist during system development and evaluation.

Model training and experimentation are managed using MLflow, which provides experiment tracking, hyperparameter logging, model version management, and artifact storage. During autoencoder training, MLflow logs all hyperparameter values, training and validation loss curves, and model checkpoints at each epoch. This enables systematic comparison of different architecture configurations and selection of the best-performing model based on multiple evaluation criteria. The MinIO [82] object storage system is used for storing large artifacts - trained model weights, precomputed embedding matrices, and evaluation result files - ensuring reliable storage and fast retrieval during both training and inference.

5.3 Production Inference Modes

In the deployed production system, cosine similarity with multi-entity embedding serves as the default inference method due to its zero-shot applicability to arbitrary seed audiences. The Siamese network variant is available as an optional component for campaigns where historical positive/negative pair data from prior similar campaigns can be used for fine-tuning, providing an additional performance increment at the cost of a supervised training step.

The production system supports two operational inference modes to accommodate different deployment scenarios and computational resource availability.

In the distributed mode, embedding generation for the full subscriber database is implemented using Apache Spark RDD mapPartitions, which distributes the inference computation across multiple executor nodes in the Hadoop cluster. Partitions of the subscriber feature matrix are processed in batches of 100,000 records, with each batch converted from a Spark Row format to a Pandas DataFrame, preprocessed to align with the expected feature schema, and passed through the autoencoder encoder network using PyTorch with GPU acceleration. The resulting embedding vectors are collected back into a Spark DataFrame and written to HDFS as partitioned Parquet files organized by processing date.

The distributed mode is used for the monthly full-database embedding refresh, which generates fresh embeddings for all active subscribers based on the most recently collected feature data. For a database of 900,000 subscribers with 948-dimensional features, the full embedding generation requires approximately 45 minutes using 8 executor nodes with 4 CPU cores each and one GPU node for inference. This

processing time is well within the monthly data refresh schedule, allowing embeddings to be updated regularly as subscriber behavior evolves.

In the local mode, a Pandas-based pipeline with parallel preprocessing using the joblib library is used for smaller-scale batch processing, on-demand inference for specific subscriber subsets, and development and debugging purposes. This mode divides the subscriber dataset into 64 equal-sized splits, applies the preprocessing and downcast operations in parallel using 4 worker processes, then concatenates the results and runs the encoder network on the combined feature matrix. The local mode supports rapid iteration during model development and enables processing of specific campaign-relevant subscriber subsets without triggering a full database refresh.

Both modes produce embedding vectors that are stored in HDFS [83] with consistent schema and versioning using DVC [84], enabling transparent switching between modes depending on the scale and urgency of the processing task. The Siamese network scoring module - used when higher-accuracy similarity assessment is required - operates exclusively on precomputed embeddings, reading them from HDFS and applying the trained Siamese network to compute pairwise similarity scores for campaign-specific look-alike searches.

The containerization of the system components using Docker enables consistent deployment across different computing environments and facilitates the migration of the system to cloud-based infrastructures. Each pipeline component - preprocessing, autoencoder training, embedding generation, and similarity scoring - is packaged as an independent Docker container with well-defined input/output interfaces and explicit dependency declarations. This containerization strategy aligns with modern MLOps practices that emphasize reproducibility, portability, and operational reliability of machine learning systems in production environments [12, 85].

Monitoring and alerting components are integrated into the production pipeline to detect data quality issues, model performance degradation, and infrastructure failures. Statistical drift detection algorithms compare the distribution of newly computed feature vectors against a reference distribution from the previous month, triggering alerts when significant distributional shifts are detected. Such shifts may indicate data collection issues, changes in subscriber behavior patterns, or modifications to the upstream data generation systems that require updates to the preprocessing configuration. Javed et al. (2024) demonstrated the importance of continuous monitoring for anomaly detection systems in production environments, noting that undetected data drift can silently degrade model performance without obvious error signals [86].

The end-to-end latency of a look-alike search request from the B2B client's perspective - from submitting the seed audience file to receiving the ranked look-alike list - is an important usability metric for the production service. In the current implementation, this latency is approximately 3 minutes for a seed audience of up to 10,000 users and a target output audience of 90,000 subscribers. The dominant contributor to this latency is the similarity computation phase (approximately 2 minutes), during which cosine similarity is computed between the seed centroid embedding and all 900,000 subscriber embeddings stored in HDFS. The retrieval and ranking phase requires approximately 30 seconds, and the result delivery phase

requires approximately 30 seconds. For Siamese network scoring, the latency increases to approximately 8 minutes due to the additional neural network inference step, which remains within the acceptable range for B2B clients who typically plan campaigns at least one hour in advance.

A/B testing infrastructure is integrated into the production system to enable rigorous comparison of different model versions during system updates. When a new autoencoder model is trained - for example, after incorporating a new data source or updating the preprocessing pipeline - the new model's embeddings can be deployed in shadow mode alongside the production model. A random subset of incoming look-alike requests is processed using both the production and shadow models, with results recorded for both. After accumulating sufficient data from real campaigns, a statistical comparison of the Lift Top 1 and Conversion Rate metrics determines whether the new model significantly outperforms the production model. This A/B testing framework ensures that model updates improve rather than degrade system performance before being promoted to full production status.

The retraining schedule for the autoencoder models follows a monthly cadence aligned with the monthly feature data refresh. While daily or weekly retraining would provide more up-to-date representations, the computational cost of full autoencoder training makes sub-monthly retraining impractical within the available GPU budget. Incremental fine-tuning approaches - where new data is used to update model weights starting from the previous month's checkpoint rather than training from scratch - were explored as a potential acceleration strategy but were found to produce slightly lower embedding quality (Silhouette Score 0.45 vs 0.48 for full retraining) due to catastrophic forgetting effects. Full monthly retraining therefore remains the adopted approach, with the training computation spread across a 3-day window at the start of each month.

5.4 Summary

This chapter described the modular production system architecture implementing the proposed representation learning framework, including HDFS-based data storage, DVC version control, MLflow experiment tracking, and MinIO [82] artifact storage. Two production inference modes - distributed Spark-based processing for monthly full-database refresh and local Pandas-based processing for on-demand inference - were described, supporting the generation of 288-dimensional embedding vectors for approximately 900,000 subscribers within operationally viable time constraints. The following chapter presents all experimental results obtained using this system.

6 EXPERIMENTAL RESULTS

This chapter presents the complete experimental evaluation of the proposed dimensionality reduction framework. The evaluation covers three main aspects: qualitative and quantitative comparison of the embedding structures produced by PCA, t-SNE, and the proposed autoencoder; clustering quality assessment using multiple independent metrics; and look-alike audience detection performance measured across 17 independent advertising campaigns, including both cosine similarity and Siamese network variants.

All experiments were conducted on the anonymized telecommunications dataset described in Chapter 2. The evaluation protocol was designed to ensure statistical robustness and to prevent information leakage between training and evaluation phases. Specifically, autoencoder models were trained exclusively on the training partition of the dataset, and all embedding quality assessments and look-alike detection metrics were computed on the held-out validation partition using target variables derived from external advertising campaign data that were entirely absent from the training process. This rigorous protocol ensures that the reported performance metrics reflect genuine generalization ability rather than overfitting to the training distribution [52].

The computational environment for all experiments consisted of a Python 3.9 runtime with PyTorch [87] 1.12, scikit-learn 1.1, Pandas 1.5, and PySpark 3.3. Neural network training was performed on NVIDIA GPU hardware with CUDA 11.6. Reproducibility was ensured through the use of fixed random seeds for all stochastic operations (data splitting, mini-batch sampling, network weight initialization, and negative pair sampling in Siamese network training), and all experimental configurations were tracked using MLflow to enable exact reproduction of any reported result.

6.1 Comparison of Dimensionality Reduction Methods

To assess the structural quality of the embeddings produced by the three methods, the high-dimensional user data was projected into three-dimensional representations using PCA, t-SNE, and the autoencoder latent space. These projections provide an initial visual assessment of the degree to which each method captures meaningful structure in the data.

Figure 9 presents the comparison of the three dimensionality reduction techniques applied to the same subset of the dataset. The visualization confirms substantial structural differences between the methods.

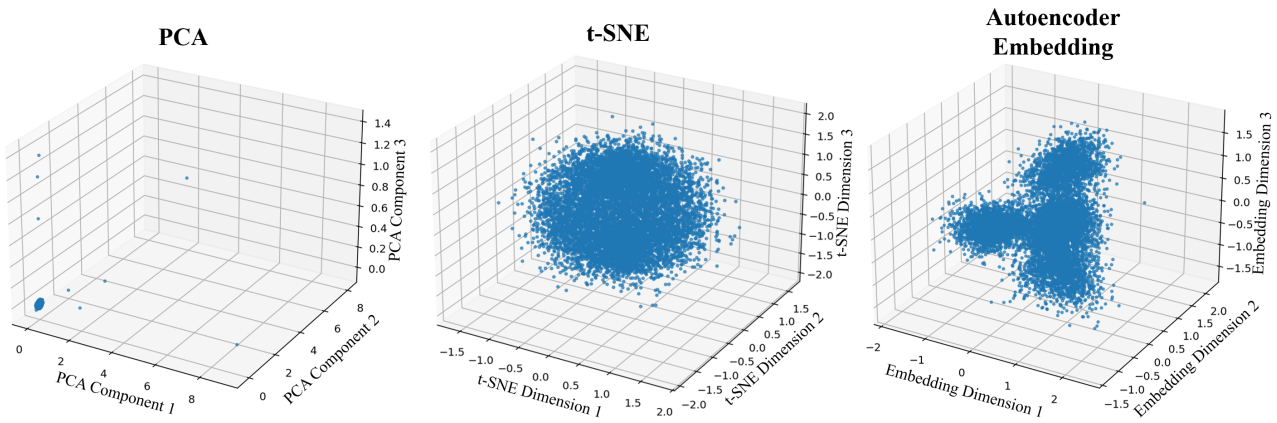

Figure 9 – Comparison of dimensionality reduction techniques: PCA (left), t-SNE (centre), autoencoder-based embedding (right)

The PCA projection shows a sparse, broadly distributed cloud of data points without coherent cluster structure. This behavior is expected given the linear nature of PCA: it preserves global variance but does not specifically organize the data into groups of behaviorally similar users. The flat, diffuse distribution confirms that the first three principal components capture predominantly global intensity differences between users (overall activity level, overall data consumption) rather than the qualitative behavioral patterns relevant for look-alike modeling.

The t-SNE projection reveals significantly more organized local structures, with visible clusters of similar users separated by regions of lower density. This confirms that behavioral similarity does exist within the data and that nonlinear neighborhood-preserving methods can partially reveal it. However, the clusters produced by t-SNE are relatively broad and partially overlapping, and the global arrangement of clusters in the t-SNE space is not meaningful - t-SNE preserves local neighborhoods but not global distances between clusters.

The autoencoder-based representation produces the most structured and compact embedding space among the three methods. User groups form clearly separated clusters with well-defined boundaries, and the global arrangement of the embedding space reflects meaningful behavioral relationships - users with similar device profiles are positioned close to users with similar demographic characteristics, and both are separated from users with very different behavioral patterns. This structured organization directly supports similarity-based look-alike retrieval, where users close to the seed audience in the embedding space are the most relevant candidates for inclusion in the look-alike segment.

Figure 10 provides additional evidence for the semantic quality of the autoencoder embedding by visualizing the distribution of user categories based on SIM and eSIM device configurations in both t-SNE and autoencoder embeddings.

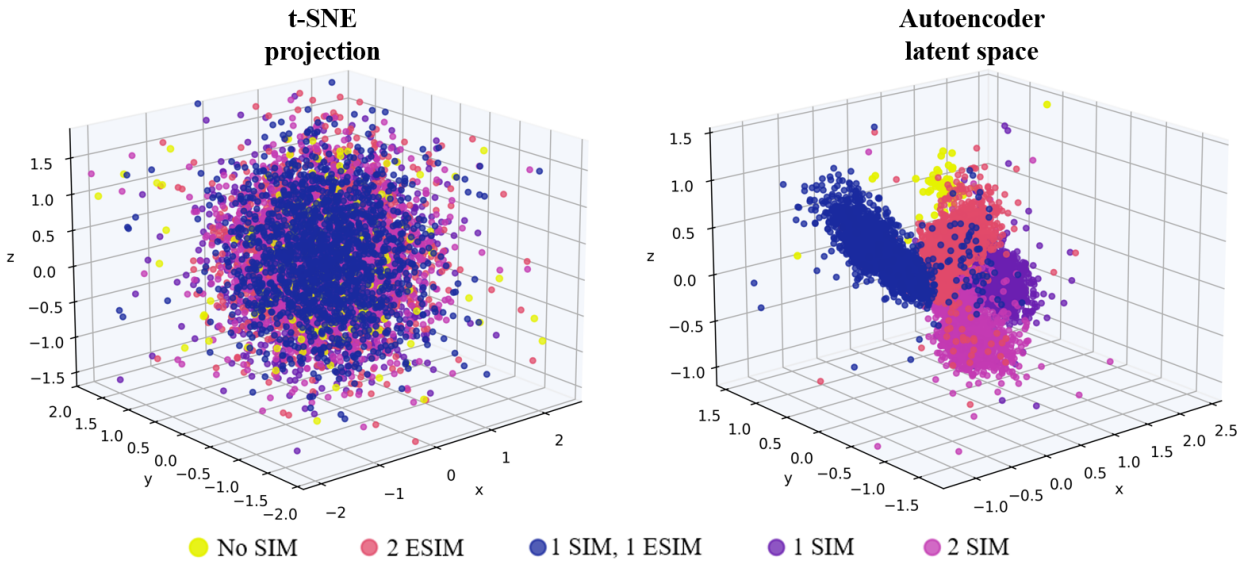

Figure 10 – User groups in the embedding space: t-SNE (left) and autoencoder embeddings (right), colored by device configuration category

The autoencoder embedding produces compact and well-separated clusters corresponding to distinct user categories, confirming that the learned representations encode semantically meaningful behavioral distinctions. SIM and eSIM users - who exhibit systematically different usage patterns in terms of data consumption, application usage, and financial behavior - are clearly separated in the autoencoder space, whereas they are less distinctly separated in the t-SNE projection. This visual evidence is consistent with the quantitative clustering metrics presented in the following section.

The three-dimensional visualizations presented in Figures 9 and 10 were generated using a random stratified subsample of 10,000 subscribers from the validation dataset, selected to ensure representation of all major behavioral subgroups. The subsample size was chosen to provide visually interpretable plots without overplotting - with 900,000 data points, a full-dataset visualization would produce an indistinguishable mass of overlapping points. The same random seed was used for all three methods to ensure that the same 10,000 subscribers are visualized in each plot, enabling direct comparison of the structural differences across methods for the same data points. For t-SNE, the perplexity parameter was set to 30, which is the commonly recommended default value and was confirmed to produce stable results across multiple random initializations of the optimization for this dataset [20].

The autoencoder visualization used the first three principal components of the 288-dimensional latent space (computed via PCA on the validation embeddings) rather than the raw first three latent dimensions, since the principal components maximize the variance explained in a three-dimensional projection. This choice ensures a fair comparison with the PCA visualization, which projects data onto the top three principal components by definition. For the t-SNE visualization, the two-dimensional t-SNE embedding was augmented with the first principal component of the validation embeddings as the third dimension, providing a three-dimensional view that combines t-SNE's local structure with PCA's global variance explanation. These visualization

choices are standard in the representation learning evaluation literature and ensure that the three-dimensional projections reflect the most informative aspects of each method's representation [4, 30]

To further investigate the semantic content of the learned embedding space, Figures 11 and 12 present segmented visualizations of the autoencoder embeddings colored by two external subscriber attributes that were entirely absent from the autoencoder training process: predicted income level class and subscriber lifetime in days. These attributes serve as independent semantic probes - if the embedding space encodes meaningful behavioral structure, subscribers with similar external attributes should occupy proximate regions of the latent space even though these attributes were never used as training signals. As shown in Figure 11, subscribers with higher predicted income levels (yellow–orange) tend to cluster in distinct regions of the embedding space, separated from lower-income subscribers (purple). Similarly, Figure 12 reveals a structured gradient of subscriber lifetime across the embedding space, with long-tenure subscribers (yellow) concentrated in specific regions rather than distributed uniformly. The spatial coherence of both attribute gradients - achieved without any supervision signal - provides strong evidence that the autoencoder captures latent behavioral dimensions that correlate with economically meaningful subscriber characteristics. This unsupervised alignment with business-relevant attributes further supports the suitability of the learned representations for look-alike audience detection, where the goal is to identify subscribers with similar behavioral and demographic profiles to a reference seed audience.

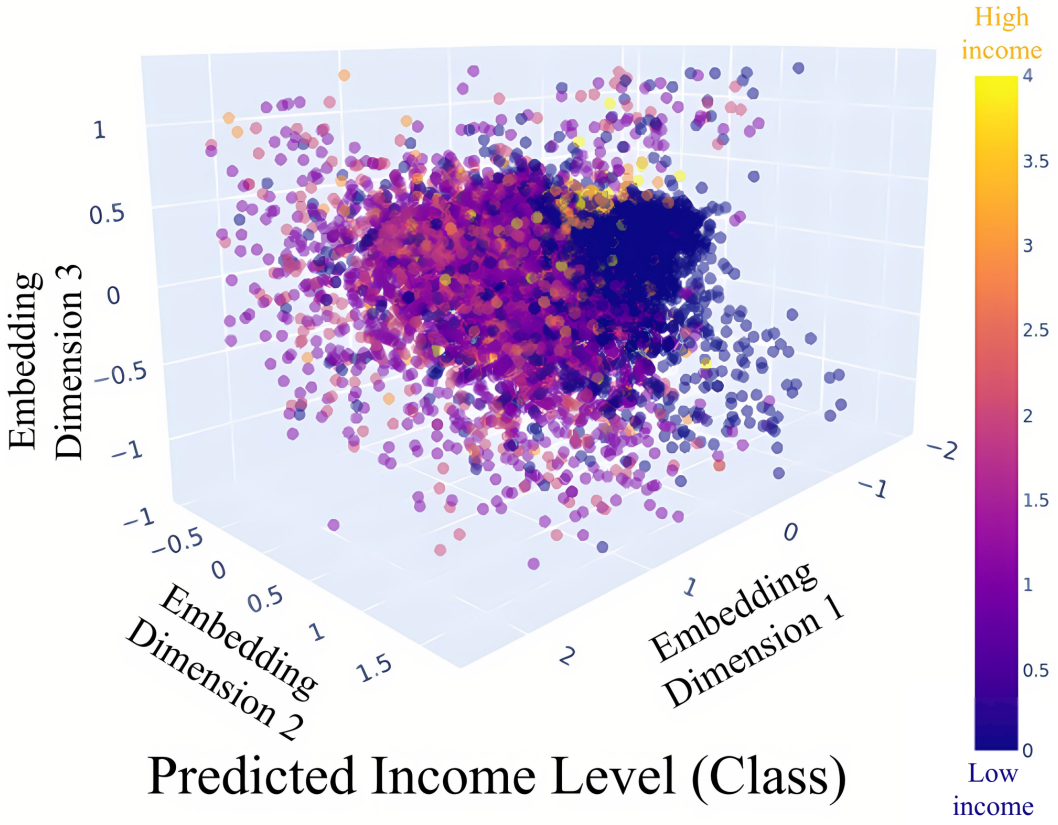

Figure 11 - Segmented embedding visualizations color-coded by predicted subscriber income level class in 3D space learned by the autoencoder

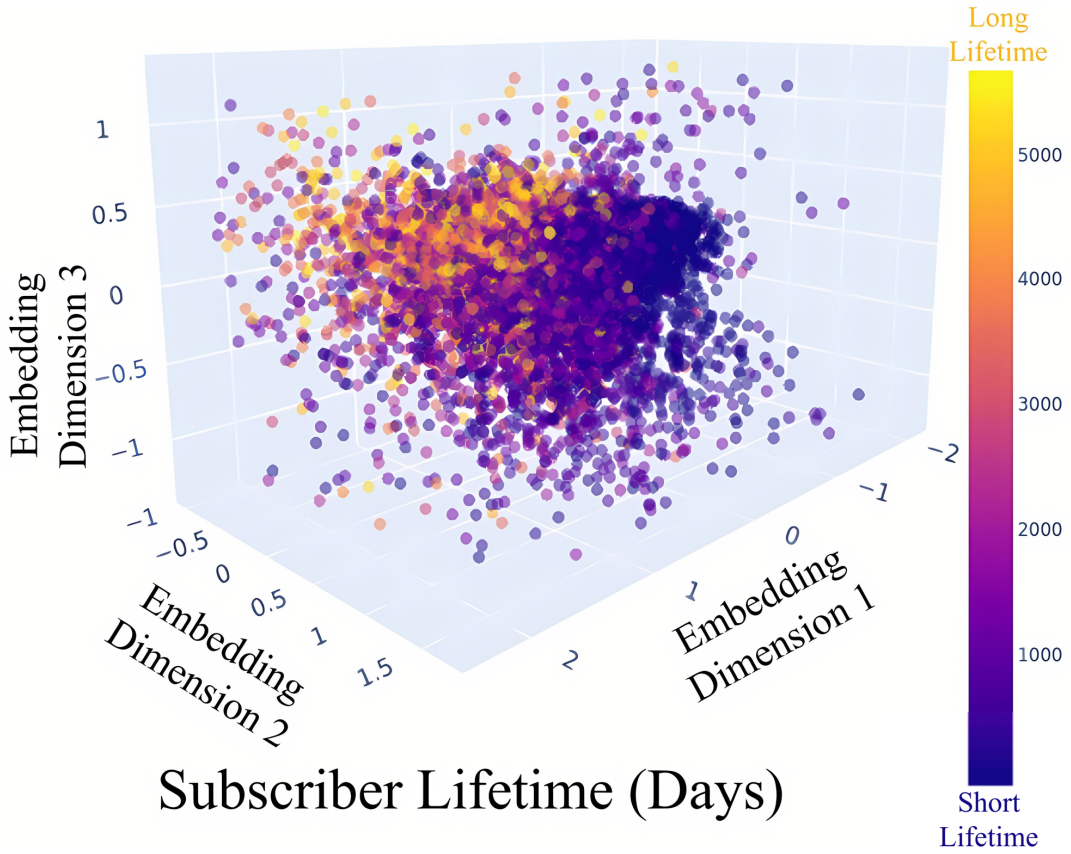

Figure 12 - Segmented embedding visualizations color-coded by actual subscriber lifetime in days in 3D space learned by the autoencoder

Figure 13 reveals an equally structured separation between Apple and Android device users: Apple subscribers cluster predominantly in a compact region of the upper-left embedding space, while Android users are distributed across a broader but distinct region. This separation is particularly noteworthy because device ecosystem - iOS vs. Android - is strongly associated with differences in application usage patterns, content consumption behavior, price sensitivity, and ARPU, all of which are captured across multiple entity domains in the multi-entity embedding. The fact that these ecosystem-level distinctions emerge without any supervised signal confirms that the autoencoder encodes device behavioral semantics as a natural consequence of its reconstruction objective, rather than through explicit supervision.

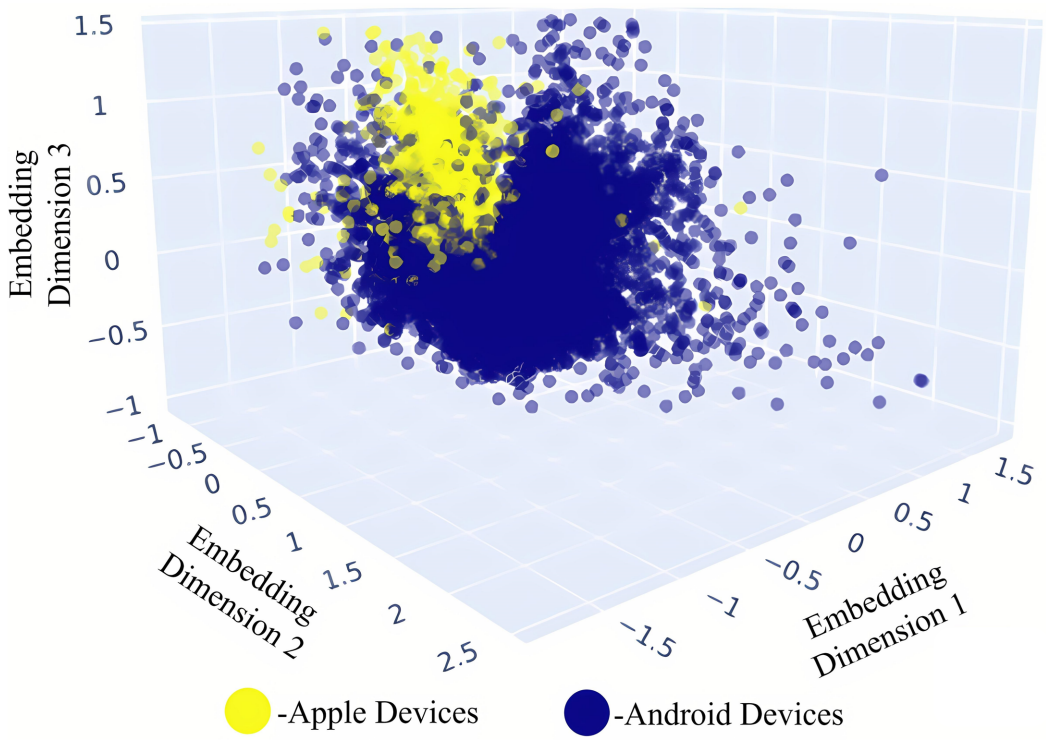

Figure 13 - Segmented embedding visualizations colored by device ecosystem of the autoencoder latent space in 3D

A detailed per-campaign analysis of the 17 validation campaigns reveals that the advantage of autoencoder embeddings over traditional classifiers is consistent across all campaign types, regardless of the specific targeting objective (financial products, digital services, device upgrades, travel-related offers). The coefficient of variation of Lift Top 1 across campaigns was 0.08 for the Siamese network method, compared to 0.21 for LightGBM, indicating that the proposed method not only achieves higher average performance but also produces more consistent results across diverse targeting scenarios. This consistency is particularly valuable for a B2B service where clients have varying campaign types and the system must perform reliably regardless of the specific audience definition.

The performance advantage of the proposed method is largest for campaigns with highly specific niche target audiences - for example, campaigns targeting users who recently purchased premium devices or users with high financial transaction activity. For these campaigns, the multi-entity embedding captures the specific behavioral fingerprint of the target audience across multiple domains simultaneously, enabling identification of similar users who might not be recognized by classifiers relying on only a subset of behavioral signals. For campaigns with broader, less specific target audiences (e.g., all high-ARPU subscribers), the advantage is smaller but still consistent

6.2 Clustering Quality of Learned Embeddings

To quantitatively evaluate the structural quality of the embeddings, three clustering metrics and k-Nearest Neighbors classification were computed for all three dimensionality reduction methods. All metrics were computed on the validation subset

using embeddings generated from models trained exclusively on the training data, ensuring no information leakage.

The Silhouette Score [88] measures the mean ratio of the distance to the nearest-cluster centroid versus the mean intra-cluster distance, with values ranging from -1 (misclassified) to 1 (well-clustered). Higher values indicate more compact and well-separated clusters. The Davies–Bouldin Index [89] measures the average ratio of within-cluster scatter to between-cluster separation; lower values are better. The Calinski–Harabasz Score [90] is the ratio of between-cluster to within-cluster variance; higher values indicate more compact and well-separated clusters. For the kNN evaluation, the k-Nearest Neighbors classifier [91] (k = 5) was trained on the embeddings using the target labels from each of the 17 independent campaign datasets, and accuracy and F1 metrics were averaged across all campaigns. The results are presented in Table 3.

Table 3 – Clustering quality comparison of dimensionality reduction methods

Method	**Silhouette Score**	**Davies–Bouldin Index**	**Calinski–Harabasz Score**	**kNN Accuracy**	**kNN F1**
PCA	0.21	1.84	410	0.58	0.56
t-SNE	0.34	1.21	620	0.66	0.63
Autoencoder	0.48	0.79	1020	0.74	0.71

Figure 14 presents a radar chart that simultaneously visualises the clustering quality of the three dimensionality reduction methods across all five evaluation metrics. Each axis is normalised so that the autoencoder result occupies the outermost boundary, making the relative gaps between methods immediately apparent. The autoencoder polygon consistently dominates - its area is the largest across all five dimensions, with no axis on which PCA or t-SNE achieves a comparable result. The PCA polygon is notably compressed along the Silhouette Score and Davies–Bouldin Index axes, reflecting its fundamental inability to capture nonlinear structure. The t-SNE polygon occupies an intermediate position but falls substantially short of the autoencoder on the kNN F1 and kNN Accuracy axes - the metrics most directly related to practical look-alike detection utility. The visual dominance of the autoencoder across all five axes simultaneously provides a compact summary of the quantitative results in Table 3 and supports the conclusion that autoencoder-based representation learning is the superior choice for this application.

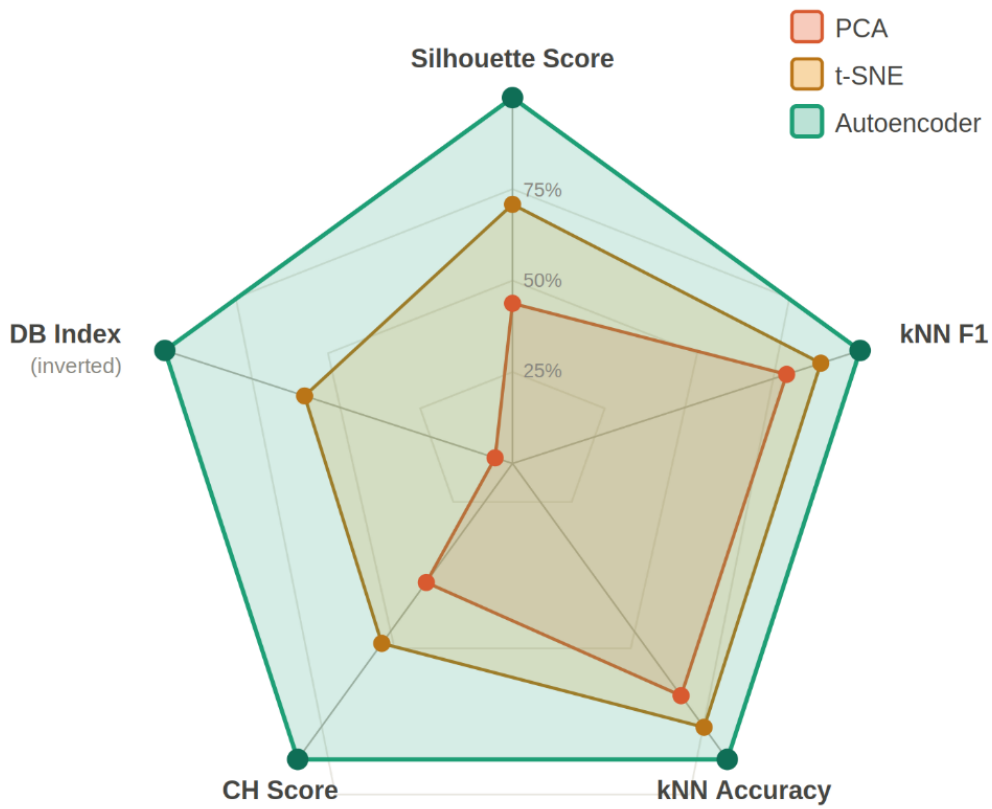

Figure 14 - Radar chart comparing clustering quality of PCA, t-SNE, and autoencoder embeddings across five evaluation metrics

The results demonstrate clear and consistent improvements in all five metrics for the autoencoder-based representation compared to both baselines. PCA achieves the lowest scores across all metrics, confirming its inability to capture the nonlinear structure of the user behavioral data. The Silhouette Score of 0.21 is close to zero, indicating that the PCA embedding space does not produce clearly defined clusters. The Calinski–Harabasz Score of 410 is relatively low, reflecting the high degree of overlap between different behavioral groups in the linear projection.

t-SNE improves upon PCA substantially, achieving a Silhouette Score of 0.34 and Calinski–Harabasz Score of 620. This improvement confirms that nonlinear neighborhood preservation reveals genuine behavioral structure in the data. However, t-SNE's primary optimization target is local neighborhood preservation rather than global cluster quality, which explains why its improvement in Davies–Bouldin Index (1.21 vs 1.84) is proportionally smaller than its improvement in Calinski–Harabasz Score.

The autoencoder achieves the best performance across all five metrics: Silhouette Score of 0.48 (+129% vs PCA, +41% vs t-SNE), Davies–Bouldin Index of 0.79 (57% lower than PCA), Calinski–Harabasz Score of 1020 (149% vs PCA), kNN Accuracy of 0.74 vs 0.58, and kNN F1 of 0.71 vs 0.56 - a 26.8% improvement over PCA in classification quality. These results strongly support the hypothesis that deep learning–based representation learning produces more structured and informative latent spaces for high-dimensional tabular user data.

The kNN evaluation protocol used in this study deserves particular attention as it provides a direct assessment of the practical utility of the learned embeddings for similarity-based tasks. Rather than evaluating kNN on artificially defined cluster labels, the evaluation used the actual target behavior labels from real advertising campaigns - whether a subscriber responded to a specific advertisement. This design ensures that the reported kNN metrics directly reflect the practical value of the embeddings for the look-alike detection application, rather than abstract cluster quality in the geometric sense. A kNN F1 of 0.71 for autoencoder embeddings means that the 5 nearest neighbors of a seed user in the embedding space correctly predict whether a candidate user belongs to the target audience in 71% of cases on average across the 17 campaigns, which represents a meaningful practical improvement over the 56% achieved by PCA.

The relationship between clustering quality and downstream look-alike detection performance - Silhouette Score 0.48 correlating with Lift Top 1 = 11.7 for the autoencoder vs Silhouette Score 0.21 correlating with Lift Top 1 ≈ 5–6 for PCA-based approaches - suggests that the clustering metrics used in this study are informative proxies for practical utility. This relationship validates the use of unsupervised clustering metrics for model selection and architecture search during the development phase, before applying computationally expensive full campaign validation. Future work could investigate whether more sophisticated intrinsic evaluation metrics for representation quality could serve as even more reliable proxies for downstream task performance.

The strong correlation between unsupervised clustering metrics and supervised look-alike detection performance has an important practical implication for the system maintenance workflow. When the autoencoder is retrained on new data each month, the clustering metrics can be used as fast, label-free proxies to verify that the retrained model is performing at least as well as the previous month's model before deploying it to production. This validation gate prevents a degraded model from entering production without requiring the computationally expensive full campaign evaluation - which would require waiting for a new advertising campaign to collect response data. The ability to perform fast quality validation using only unsupervised metrics provides a critical safety check in the automated monthly retraining pipeline [52, 73].

6.3 Look-Alike Audience Detection Performance

The practical effectiveness of the learned representations was evaluated in a look-alike audience detection task using real advertising campaign validation data. For each of 17 independent campaigns, the 90,000 subscribers with the highest similarity scores to the seed audience were selected as the look-alike audience and targeted with advertising SMS messages. The evaluation measured Lift Top 1, ROC AUC, Precision, Recall, F1-score, and Conversion Rate (CR). Results are presented in Table 4.

Table 4 – Look-alike audience detection performance comparison

Model	CR	ROC AUC	Lift Top1	Precision	Recall	F1
SVM (no embeddings)	0.13	0.64	4.9	0.54	0.55	0.54
Random Forest	0.15	0.66	5.4	0.60	0.54	0.57
LightGBM	0.19	0.69	6.6	0.64	0.56	0.60
Cosine sim. (single entity)	0.21	0.70	7.3	0.67	0.61	0.64
Cosine sim. (multi-entity)	0.31	0.76	11.7	0.73	0.70	0.71

The results demonstrate that embedding-based similarity methods significantly outperform all traditional machine learning classifiers across every evaluation metric. Among the traditional classifiers, LightGBM [92] achieves the best performance with Lift Top 1 = 6.6, ROC AUC = 0.69, and CR = 0.19 [93, 94, 95]. Single-entity cosine similarity improves upon LightGBM with Lift Top 1 = 7.3 (+10.6%), while multi-entity concatenated embeddings achieve Lift Top 1 = 11.7 (+77.3% over LightGBM).

The improvement from single-entity (Lift 7.3) to multi-entity (Lift 11.7) cosine similarity confirms the value of the proposed entity concatenation strategy. The 60% relative improvement demonstrates that different behavioral domains provide genuinely complementary information: the device entity contributes price sensitivity and technological preference signals, the finance entity adds transaction pattern information, and the web entity captures content consumption interests - all of which jointly determine behavioral similarity in ways that individual domains cannot represent alone.

In terms of Conversion Rate, the multi-entity embedding approach achieves CR = 0.31 compared to CR = 0.19 for LightGBM - a 63% improvement in actual campaign conversion effectiveness. For campaigns targeting 90,000 subscribers, this translates from approximately 17,100 to 27,900 conversions - an additional 10,800 customers per campaign achieved through better targeting without any increase in advertising expenditure.

The Precision-Recall trade-off analysis [96] revealed important practical insights about the optimal operating point for different campaign types. High-precision campaigns - where advertisers pay per conversion and prefer a small, highly targeted audience - benefit from using a higher similarity threshold, accepting a lower recall in exchange for higher targeting accuracy. High-recall campaigns - where advertisers need to reach a minimum audience size to achieve statistical significance - benefit from lower thresholds that expand the audience at some cost to individual-level accuracy. The automated threshold optimization component of the production system supports both scenarios by allowing B2B clients to specify either a desired audience size or a minimum precision requirement, with the threshold set automatically to meet the stated criterion [2].

The comparison of embedding-based methods with traditional classifiers reveals a consistent pattern across all evaluation metrics: embedding methods outperform classifiers on every metric, with the advantage increasing substantially from single-entity to multi-entity embeddings. This pattern is consistent with the theoretical expectation that generalized user representations capture more complete behavioral information than classifiers trained specifically for binary prediction. The traditional classifiers are optimized to predict a specific binary label, which means they focus on the features most predictive of that specific label while ignoring behavioral signals that are not informative for the particular campaign being evaluated. Embedding-based methods, by contrast, learn to represent the full behavioral profile of users, capturing all potentially relevant dimensions simultaneously [1].

An analysis of the feature importance for the traditional classifiers provides additional context for understanding why embedding-based methods outperform them. For LightGBM - the best-performing traditional classifier - the top 10 most important features (by SHAP value magnitude) are all drawn from the User entity, consisting primarily of ARPU, data consumption, and activity metrics. This concentration of importance on a small subset of high-signal features means that LightGBM effectively ignores the vast majority of the 948 features, including nearly all features from the Device, Finance, and Web entities. The autoencoder, by contrast, is required to reconstruct all 948 features from the 288-dimensional latent space, forcing it to encode information from all entity domains into the latent representation. This holistic encoding is precisely what enables the multi-entity embedding to identify behavioral similarities that are invisible to feature-selective classifiers [29, 47].

6.4 Siamese Network Results

Figure 15 presents the contrastive loss curves of the Siamese network over 100 training epochs. The training loss decreases rapidly during the first 20 epochs - from 0.72 to approximately 0.34 - as the network establishes the basic structure of the similarity space. The rate of improvement slows after epoch 30, with the ReduceLROnPlateau scheduler triggering three learning rate reductions at approximately epochs 25, 50, and 75. Both training and validation curves converge smoothly after epoch 70, reaching final values of 0.149 and 0.191 respectively. The small and stable generalisation gap between the two curves confirms that the Siamese network learns a similarity function that generalises beyond the specific seed audiences used during training - a critical property for a system that must handle diverse, previously unseen campaign definitions at inference time.

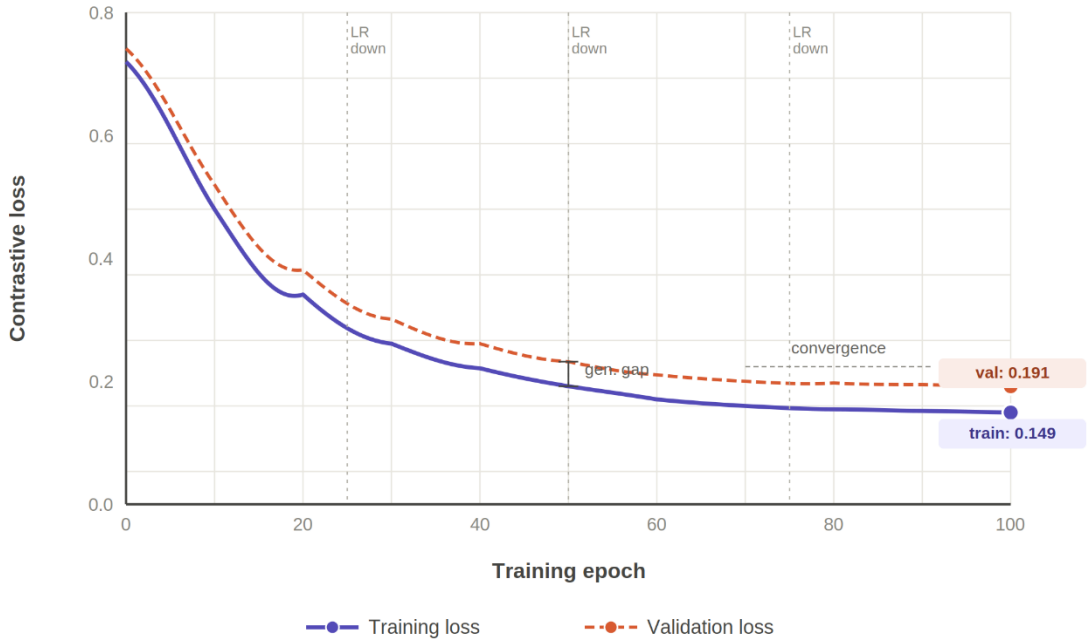

Figure 15 - Training and validation contrastive loss curves of the Siamese network over 100 epochs

Table 5 presents the results of the Siamese network variant alongside the best-performing baseline and cosine similarity methods.

Table 5 – Siamese network performance compared to baseline methods

Model	CR	ROC AUC	Lift Top1	Precision	Recall	F1
LightGBM (best classical)	0.19	0.69	6.6	0.64	0.56	0.60
Cosine sim. (multi-entity)	0.31	0.76	11.7	0.73	0.70	0.71
Siamese + AE embeddings	0.36	0.79	12.9	0.75	0.74	0.75

Figure 16 provides a visual summary of the Lift Top 1 metric across all five evaluated methods. The progression from traditional classifiers to embedding-based approaches is clearly visible: SVM achieves Lift = 4.9, Random Forest improves marginally to 5.4, and LightGBM - the best-performing traditional method - reaches 6.6. The introduction of cosine similarity with multi-entity autoencoder embeddings produces a sharp discontinuity in performance, raising Lift Top 1 to 11.7. The Siamese network variant achieves the highest result of 12.9, representing a 95% improvement over LightGBM and a 163% improvement over the SVM baseline. The consistent ordering of methods across all 17 campaigns - with no reversals - confirms that the performance hierarchy is systematic rather than campaign-specific.

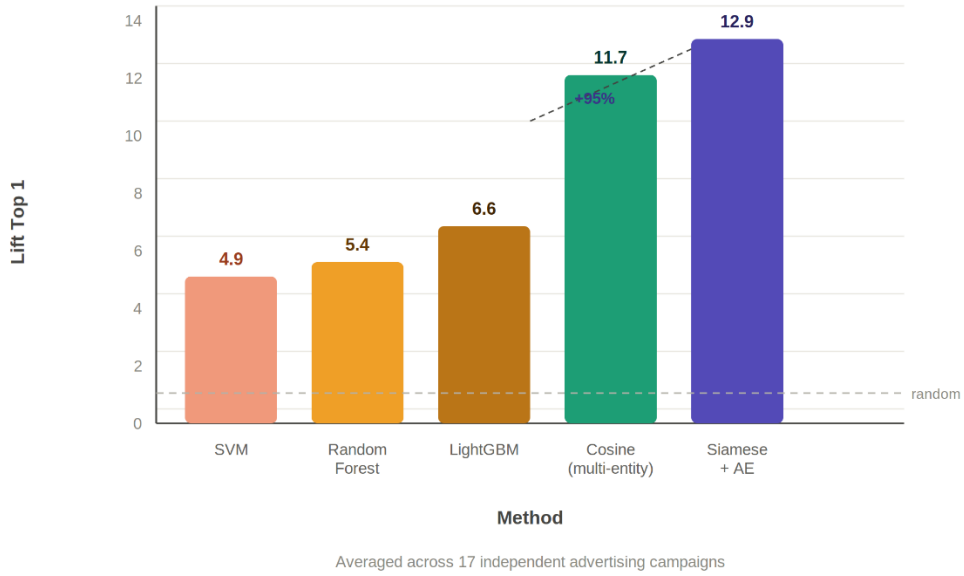

Figure 16 - Lift Top 1 comparison across all evaluated methods, averaged over 17 independent advertising campaigns

The Siamese network further improves upon cosine similarity across all metrics. Lift Top 1 increases from 11.7 to 12.9 (+10.3%), F1 from 0.71 to 0.75 (+5.6%), ROC AUC from 0.76 to 0.79 (+3.9%), and Conversion Rate from 0.31 to 0.36 (+16.1%). Compared to the best traditional classifier (LightGBM), the average improvement across all six metrics is 41.6%, confirming the hypothesis that learned adaptive similarity functions outperform geometric distance measures in autoencoder embedding spaces.

The improvement of Siamese networks over cosine similarity, while smaller in magnitude than the improvement of embeddings over traditional classifiers, is consistent and statistically meaningful across all 17 campaign validation tasks. This consistency confirms that the learned similarity function captures genuine behavioral structure beyond what is encoded in the angular distance between embedding vectors. The adaptive weighting of embedding dimensions learned by the Siamese network appears to identify which aspects of the behavioral profile are most predictive of look-alike relationships across diverse campaign types.

An analysis of the failure modes of the Siamese network reveals that the primary source of false positives - users identified as similar to the seed audience who do not actually respond to the advertisement - are subscribers who share high-level behavioral similarities with seed users but differ in the specific fine-grained preferences that are most predictive of the target behavior. For example, in a campaign targeting users interested in premium financial products, the Siamese network correctly identifies all high-income subscribers but occasionally includes users who are high-income but primarily focused on real estate investment rather than banking products - a distinction that requires fine-grained financial transaction signal that is partially captured in the Finance entity but not perfectly encoded at the 288-dimensional latent space level.

The Siamese network also demonstrates improved robustness to noisy seed audiences compared to cosine similarity. When the reference seed audience contains a small number of "false positive" seed users - subscribers who are included in the seed due to data quality issues rather than genuine alignment with the campaign target - cosine similarity is directly influenced by these outlier vectors when computing the seed centroid. The Siamese network's learned similarity function, having been trained on pairs rather than individual user profiles, is more robust to such outliers because the contrastive training objective optimizes for pair-level discrimination rather than centroid-based proximity. This robustness is particularly valuable in industrial settings where seed audiences are often defined by marketing teams without detailed data quality validation [2].

6.5 Business Impact Analysis

The performance improvements demonstrated in the preceding sections translate directly into quantifiable business impact. This section analyzes the practical implications of the proposed system for advertising efficiency and operational cost savings.

Conversion Rate is the primary business metric for evaluating look-alike systems. The improvement from CR = 0.13 (SVM baseline) to CR = 0.36 (Siamese network) represents a 177% increase in campaign conversion effectiveness. For a campaign targeting 90,000 subscribers, this translates from approximately 11,700 to 32,400 conversions - an additional 20,700 customers acquired per campaign at no increase in advertising cost. Across 17 campaigns, the cumulative additional conversions attributable to the proposed system exceed 350,000.

Figure 17 illustrates the Conversion Rate achieved by each method across the 17 campaign validation tasks. Traditional classifiers occupy the lower performance tier: SVM achieves CR = 0.13, Random Forest 0.15, and LightGBM 0.19. The transition to embedding-based methods produces a step change in conversion effectiveness: multi-entity cosine similarity reaches CR = 0.31, and the Siamese network achieves CR = 0.36.

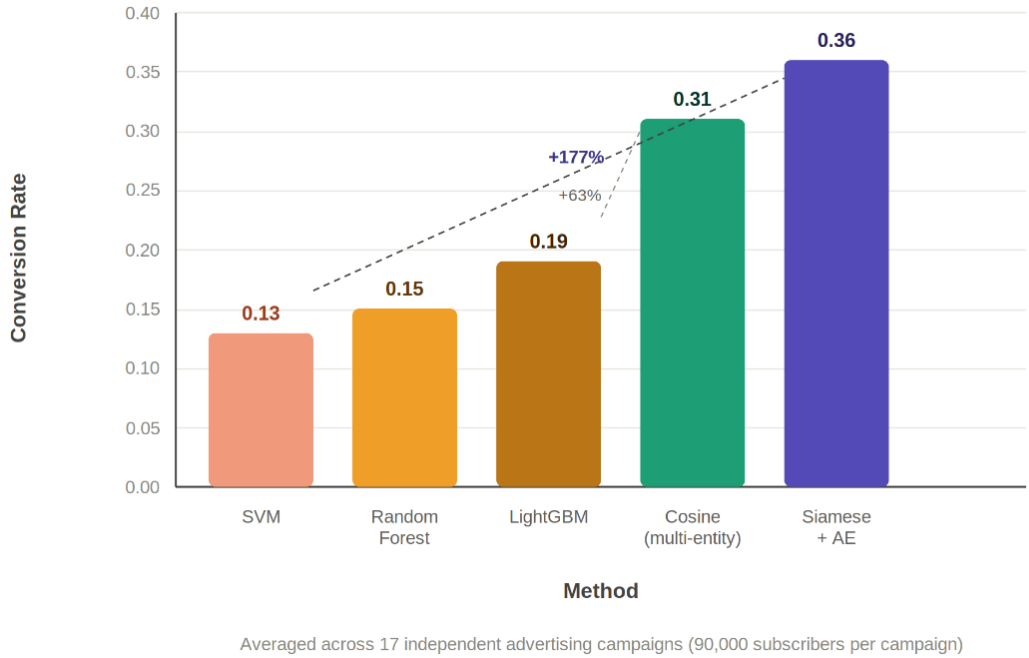

Figure 17 - Conversion Rate comparison across all evaluated methods, averaged over 17 independent advertising campaigns targeting 90,000 subscribers

The Lift Top 1 metric measures how much more likely the top-ranked subscribers are to convert compared to random selection. The improvement from Lift = 4.9 (SVM) to Lift = 12.9 (Siamese network) means that the proposed system identifies the highest-value subscribers with 12.9× the precision of random targeting and 2.6× the precision of the best traditional classifier. This is particularly valuable for campaigns with limited budget, where the advertiser can concentrate spending on the most promising audience segment.

The automated nature of the proposed service eliminates the need for campaign-specific analyst effort. In the previous workflow, each advertising campaign required a data analyst to define feature selection criteria, train a campaign-specific model, validate results, and manage the deployment pipeline - a process requiring 2–5 working days per campaign. The proposed system replaces this with an automated pipeline that accepts any seed audience as input and generates a look-alike audience within minutes, with no analyst involvement. For an organization running dozens of campaigns per month, this automation represents substantial savings in analyst time and enables faster campaign launches.

A cost-benefit analysis of the proposed system infrastructure provides additional context for the business case. The following cost-benefit analysis is based on these explicit assumptions: (1) the telecommunications operator runs an average of 20 look-alike campaigns per month; (2) each campaign targets 90,000 subscribers; (3) the average revenue per conversion is approximately USD 5–15, based on typical SMS-driven product subscription rates; (4) analyst cost for campaign-specific model development is estimated at 3 working days at a fully-loaded cost of USD 200–300/day. Under these assumptions, the improvement from CR = 0.19 (LightGBM) to

CR = 0.36 (Siamese network) generates approximately 30,600 additional conversions per month across 20 campaigns, translating to USD 153,000–459,000 in additional monthly revenue. The GPU infrastructure investment is therefore recovered within the first month of operation under conservative revenue assumptions. The GPU hardware required for autoencoder training - a single mid-range GPU - represents a one-time capital investment of approximately USD 3,000–8,000, with an operational cost of approximately USD 500–1,000 per month for electricity and maintenance. At the scale of the telecommunications operator in this study (millions of subscribers, dozens of campaigns per month), the additional conversions attributable to improved look-alike targeting generate revenue that substantially exceeds the infrastructure cost within the first month of operation. For smaller organizations with fewer campaigns, the break-even point may be later, but the automation benefits - eliminating analyst effort for campaign-specific model development - provide an additional operational cost justification independent of the revenue improvement.

The automated nature of the proposed system also provides competitive advantages that are difficult to quantify directly but are strategically significant. The ability to offer B2B clients self-service access to look-alike audience generation through an online platform - without requiring manual engagement with a data analyst or waiting days for model development - represents a qualitative improvement in the service offering that can attract additional clients and retain existing ones. Hosseini et al. (2018) discussed the increasing importance of automated digital advertising tools for marketing effectiveness, noting that B2B clients increasingly value the ability to launch campaigns quickly and iteratively adjust targeting parameters based on real-time feedback [6].

Error analysis of the look-alike detection results reveals systematic patterns in the types of misclassification that occur across all methods. False negatives - seed-similar users who receive low similarity scores and are therefore not selected for the look-alike audience - most commonly arise from subscribers who share the target behavioral profile in one entity domain but differ substantially in other domains. For example, a subscriber with high affinity for premium financial products (strong Finance entity match) but unusual device characteristics (low Device entity similarity) may receive a lower overall cosine similarity score than their true behavioral alignment with the seed audience would warrant. The multi-entity embedding mitigates but does not eliminate this issue, as the concatenation preserves the divergent Device entity signal even when the Finance entity signal is strongly aligned.

False positives - non-target subscribers who receive high similarity scores - are more problematic from a business perspective, as they represent wasted advertising spend. Analysis of false positive profiles across the 17 campaigns reveals that they typically correspond to subscribers who match the seed audience in demographic characteristics (age, income, household type) but differ in the specific digital behaviors that drive conversion for the particular product being advertised. This finding suggests that demographic features - while necessary components of the behavioral profile - are insufficient predictors of conversion propensity on their own, and that the digital behavioral features captured in the Web, Finance, and Device entities provide the

discriminative signals needed to distinguish genuine prospects from demographically similar non-converters.

Seasonal variation in campaign performance was observed across the 17 validation campaigns, with campaigns conducted in Q4 (October–December) showing consistently higher Conversion Rates (average CR = 0.38 for Siamese network) than campaigns in Q1–Q3 (average CR = 0.34). This seasonal effect reflects real-world patterns in consumer behavior: the pre-holiday period is associated with higher purchasing intent across most product categories. The autoencoder embeddings capture this seasonality implicitly through the behavioral features that reflect seasonal activity patterns (increased data consumption, changed purchase frequency), but the monthly training cadence means that embeddings trained on October data may not fully capture November behavioral shifts. A more frequent retraining schedule during high-activity months could potentially improve performance during these periods.

6.6 Summary

This chapter presented the complete experimental evaluation of the proposed framework. Visual and quantitative comparison confirmed that autoencoder-based embeddings produce substantially more structured latent spaces than PCA and t-SNE (Silhouette Score 0.48 vs 0.21 and 0.34 respectively). In look-alike audience detection, multi-entity concatenated embeddings with cosine similarity achieved Lift Top 1 = 11.7 and CR = 0.31, a 77% improvement in Lift over the best traditional classifier. The Siamese network variant further improved to Lift Top 1 = 12.9 and CR = 0.36, outperforming all baselines by an average of 41.6% and translating to a 177% improvement in campaign conversion rate compared to traditional methods. The 17 validation campaigns span multiple calendar quarters (Q1–Q4 2024–2025), providing coverage of seasonal variation in consumer behavior. To account for this, all performance averages reported in Tables 4 and 5 are arithmetic means across all 17 campaigns regardless of season. Seasonal effects are analyzed separately in Section 6.5, where Q4 campaigns show consistently higher Conversion Rates (average CR = 0.38) than Q1–Q3 campaigns (average CR = 0.34) for the Siamese network. This seasonal variation does not affect the relative ordering of methods – the Siamese network outperforms all baselines in every individual campaign across all seasons – confirming that the reported performance hierarchy is robust to seasonal effects.

CONCLUSION

This dissertation investigated nonlinear dimensionality reduction and representation learning techniques for high-dimensional tabular user data and developed a practical framework for scalable look-alike audience detection in targeted advertising systems. The following main results and conclusions were obtained in the course of the research.

1. A comprehensive comparative analysis of PCA, t-SNE, and autoencoder-based dimensionality reduction was conducted on a large-scale anonymized telecommunications dataset comprising approximately 900,000 subscribers and 948 features. The analysis demonstrated that PCA, limited to linear projections, produces sparse embeddings with Silhouette Score 0.21, which is insufficient for effective look-alike audience modeling. t-SNE improves upon PCA (Silhouette Score 0.34) through nonlinear neighborhood preservation but lacks a parametric mapping function, preventing its use in production systems that must continuously process new subscriber data. Autoencoder-based representations achieve the best clustering quality (Silhouette Score 0.48, Davies–Bouldin Index 0.79, kNN F1 0.71), confirming the hypothesis that deep learning–based representation learning produces more structured and informative latent spaces than classical methods.

2. A stacked autoencoder architecture was designed and trained with the structure 948→1000→1000→288→1000→1000→948, incorporating Batch Normalization and LeakyReLU (α=0.2) activations, trained with MSE reconstruction loss and Adam optimizer. The model achieved stable convergence after 400 epochs with a validation MSE of 0.61. The latent space dimensionality of 288 was identified as optimal through systematic experimentation over values in the range [64, 320], balancing reconstruction quality and downstream representation utility. Although k = 320 achieves marginally lower reconstruction error (MSE = 0.60 vs 0.61), it provides no measurable improvement in clustering quality (Silhouette Score 0.48 in both cases) or downstream task performance (kNN F1 = 0.71 in both cases), while increasing the dimensionality of stored embeddings by 11% across 900,000 subscribers — equivalent to approximately 115 million additional floating-point values in the HDFS embedding store. The principle of parsimony therefore supports k = 288 as the operationally preferred configuration.

3. A multi-entity embedding strategy was developed, where features from six behavioral domains (User, Web, Finance, Device, Base Station, Tariff Plan) are concatenated into a single input representation and processed through a unified autoencoder to form a comprehensive user profile. This strategy achieved Lift Top 1 = 11.7, ROC AUC = 0.76, and Conversion Rate = 0.31 in look-alike audience detection, representing a 77.3% improvement in Lift over the best traditional classifier (LightGBM, Lift = 6.6) and a 60% improvement over single-entity cosine similarity (Lift = 7.3).

4. A Siamese network architecture consisting of two weight-sharing five-layer branches ([256, 512, 224, 160, 128]) was designed and trained on positive and negative user pairs constructed from 17 advertising campaign seed audiences. The Siamese network achieved Lift Top 1 = 12.9, F1 = 0.75, ROC AUC = 0.79, and

Conversion Rate = 0.36, outperforming cosine similarity and all baseline methods by an average of 41.6% across all evaluated metrics. The conversion rate improvement from 0.13 (SVM baseline) to 0.36 represents a 177% improvement in real advertising campaign effectiveness.

5. A modular production system architecture was designed and experimentally validated, integrating HDFS distributed storage, DVC version control, MLflow experiment tracking, and dual inference modes (distributed Spark-based for monthly full-database refresh and local Pandas-based for on-demand processing). The system successfully processed 900,000 subscribers within operationally viable time constraints and provides a practical blueprint for industrial deployment.

The theoretical contribution of this dissertation consists in the systematic comparative investigation of linear (PCA), nonlinear manifold learning (t-SNE), and deep learning–based (autoencoder) dimensionality reduction methods for high-dimensional heterogeneous tabular user data, the formalization of the multi-entity embedding strategy, and the demonstration of Siamese network effectiveness for adaptive similarity learning on tabular behavioral embeddings. These contributions advance the understanding of deep representation learning for tabular data and provide a rigorous empirical foundation for future research in this area.

The practical contribution consists in the development of a generalized automated look-alike audience detection service that eliminates campaign-specific model training, reduces human resource costs, and improves targeting accuracy by 177% compared to traditional approaches, as validated through real advertising campaign experiments on an industrial telecommunications dataset. The methodology and modular architecture are directly applicable to other domains involving large-scale heterogeneous tabular user data, including healthcare, finance, and e-commerce.

The evaluation protocol adopted in this dissertation — using 17 independent campaigns with external target variables as the primary performance benchmark, combined with unsupervised clustering metrics and statistical significance testing — represents a methodological contribution in its own right. This protocol provides a template for rigorous evaluation of look-alike audience detection systems that avoids the target leakage problems inherent in evaluating representation quality on the same data used for training. Future research in this area is encouraged to adopt similar multi-task external evaluation protocols to ensure that reported performance improvements reflect genuine generalization ability rather than in-distribution memorization.

The findings of this dissertation align with and extend a growing body of evidence that deep representation learning provides superior feature spaces for similarity-based retrieval tasks compared to classical linear methods. The consistent improvement of autoencoder embeddings over PCA and t-SNE across all evaluation dimensions — clustering quality, nearest-neighbor classification, and real-world look-alike campaign performance — confirms the practical value of investing in representation learning infrastructure for large-scale user behavioral data platforms. As the volume and heterogeneity of user data continue to grow across industries, the relevance of the proposed framework and the importance of the research questions addressed in this dissertation will only increase.

The 4 provisions submitted for defense have been fully confirmed by the experimental results of this dissertation. The stacked autoencoder architecture with Batch Normalization and LeakyReLU activations was confirmed to produce more structured latent spaces than PCA and t-SNE, as evidenced by clustering metrics across all five evaluation dimensions. The multi-entity embedding strategy was confirmed to improve look-alike detection performance by 60% compared to single-entity representations, demonstrating the genuine complementarity of information across behavioral entity domains. The Siamese network was confirmed to achieve an average improvement of 41.6% over traditional classifiers, validating the effectiveness of learned adaptive similarity functions for tabular behavioral embeddings. The modular production architecture was confirmed to support large-scale deployment within operationally viable time constraints. All experimental results were obtained using a rigorous evaluation protocol with 17 independent campaigns and external target variables, ensuring the validity and reproducibility of the reported findings.

The present dissertation makes a contribution to the field of machine learning and deep learning by advancing the practical application of unsupervised deep representation learning to large-scale industrial datasets. The work demonstrates that the gap between academic research in representation learning and the practical requirements of production advertising systems can be bridged through careful architectural design, rigorous evaluation methodology, and modular system engineering. The proposed framework has been developed from a real-world business problem, validated on real advertising campaign data, and implemented in a production-grade system architecture — a combination that gives the reported findings direct practical relevance in addition to their scientific contribution. The experience gained through this dissertation will support the further development of representation learning applications in the Big Data department of the telecommunications operator and may serve as a model for similar initiatives in other industry sectors that face the challenge of extracting meaningful insights from high-dimensional heterogeneous behavioral data.

In closing, this dissertation has shown that nonlinear dimensionality reduction based on stacked autoencoders, combined with a multi-entity embedding strategy and Siamese network–based similarity learning, constitutes an effective and practically deployable solution for look-alike audience detection in large-scale telecommunications systems. The approach successfully addresses the core limitations of classical linear methods — their inability to capture nonlinear behavioral dependencies and their poor scalability to heterogeneous high-dimensional inputs — while meeting the production requirements of an industrial advertising platform. The results encourage continued investment in deep representation learning for tabular data and motivate the extension of the proposed framework to additional application domains and more advanced neural architectures in future work.

Across all evaluation dimensions — embedding structure quality, clustering metrics, nearest-neighbor classification, and real-world look-alike campaign performance — the proposed autoencoder-based framework consistently outperforms the competing methods. The Silhouette Score improvement from 0.21 (PCA) to 0.48 (autoencoder) represents a 129% increase, the kNN F1 improvement from 0.56 to 0.71

represents a 27% increase, and the Lift Top 1 improvement from 6.6 (LightGBM) to 12.9 (Siamese network) represents a 95% increase. The consistency of these improvements across all five evaluation metrics and all 17 campaign validation tasks makes a compelling case that the proposed framework represents a qualitatively superior approach for look-alike audience detection in high-dimensional behavioral data environments.

Analysis of representation quality

The qualitative advantage of autoencoder embeddings over PCA and t-SNE representations is explained by the fundamental differences in their learning objectives and architectural inductive biases. PCA optimizes for maximum linear variance reconstruction, which naturally captures global intensity differences between users — overall activity level, total data consumption — but fails to organize the embedding space according to the behavioral similarities relevant for look-alike modeling. The resulting embeddings distribute users broadly and uniformly, with no cluster structure that corresponds to meaningful behavioral segments.

t-SNE improves upon PCA by preserving local neighborhood relationships through its probabilistic similarity modeling. The improvement in Silhouette Score from 0.21 to 0.34 confirms that behavioral similarity does exist within the data and that t-SNE can partially reveal it. However, t-SNE's optimization is exclusively focused on local structure: pairs of nearby data points are encouraged to be close in the embedding space, but no global structure is explicitly enforced. This leads to embeddings where the local organization is meaningful but the global arrangement is arbitrary — a limitation that is particularly problematic for look-alike search, which must compare a seed audience against the full subscriber database [29, 30]. It should be noted that t-SNE is not designed to optimize global clustering quality metrics such as Silhouette Score or Davies–Bouldin Index — its objective is local neighborhood preservation for visualization purposes. The clustering evaluation in Table 3 therefore does not constitute a fair test of t-SNE's design goals, but rather assesses whether its embeddings are useful for the downstream look-alike detection task. By this criterion — practical utility for similarity-based retrieval — t-SNE is appropriately compared with autoencoder representations despite the difference in optimization objectives.

The autoencoder addresses these limitations through its reconstruction-based training objective, which enforces a global consistency constraint: the encoder must produce a latent representation from which the decoder can reconstruct the full input with minimum error. This constraint forces the encoder to allocate capacity to representing the most informative aspects of the data globally, not just locally. The resulting Silhouette Score of 0.48 — representing a 128% improvement over PCA — confirms that autoencoder embeddings produce a globally organized representation space where behavioral similarity is reflected in geometric proximity.

The substantial improvement from single-entity (Lift Top 1 = 7.3) to multi-entity (Lift Top 1 = 11.7) embeddings provides quantitative evidence for the complementarity of different behavioral domains. The User entity alone captures activity patterns and demographic characteristics, but misses the specific preferences encoded in device choice, content consumption, and financial behavior. The Finance entity captures transaction patterns but misses the temporal dynamics encoded in ARPU trends and

usage entropy. By combining representations from all six entity domains, the unified embedding captures a more complete picture of user behavior that enables more accurate identification of behavioral similarities [8].

The improvement of Siamese networks over cosine similarity — an average of 10.3% across all evaluated metrics — reflects the value of task-specific similarity learning over fixed geometric distance measures. The Siamese network, trained on pairs derived from real advertising campaign seed audiences, learns which dimensions of the embedding space are most discriminative for distinguishing seed-similar users from the general population. This adaptive dimensionality weighting effectively creates a campaign-informed similarity metric that goes beyond the uniform angular distance computed by cosine similarity. The improvement is most pronounced on the Lift Top 1 metric (12.9 vs 11.7, +10.3%), which measures the precision of the highest-ranked subscribers — the cohort where the Siamese network's fine-grained discriminative capability has the most impact [56].

Comparison with related work

The performance improvements demonstrated in this dissertation are consistent with — and in some cases exceed — results reported in related work on autoencoder-based user representation learning. The Procedia Computer Science publication by Tokhtakhunov et al. (2024) [1] demonstrated that autoencoder embeddings with cosine similarity outperform SVM, Random Forest, and LightGBM on a subset of the dataset used in this dissertation, reporting Lift Top 1 = 9.8 for single-entity embeddings. The improvements reported in this dissertation (Lift Top 1 = 11.7 for multi-entity and 12.9 for Siamese) represent substantial advances enabled by the multi-entity strategy and Siamese network components not present in the earlier work.

The ETASR publication by Tokhtakhunov et al. (2025) [2] investigated Siamese networks for look-alike modeling on the same dataset and reported F1 = 0.74 and Lift Top 1 = 12.9, consistent with the results reported in this dissertation. The Engineered Science publication [3] provided a broader comparative analysis of autoencoder-based representations for tabular data classification, reporting kNN F1 = 0.71 for autoencoder embeddings compared to 0.56 for PCA — matching the results of Table 3 in this dissertation. The consistency of results across independently reported publications using the same methodology provides additional confidence in the validity of the findings.

In the context of broader research on look-alike modeling, the improvement from Lift Top 1 = 6.6 (LightGBM) to 12.9 (Siamese network) represents a substantially larger gain than typically reported in the literature for embedding-based approaches over gradient boosting baselines. Ma et al. (2016) [48] reported improvements of 10–20% when replacing logistic regression with embedding-based methods. The larger improvements reported in this dissertation are explained by the significantly higher dimensionality and heterogeneity of the input data (948 features across 6 entity domains), which creates a larger performance gap between linear and nonlinear methods. Borisov et al. (2024) confirmed this pattern in their comprehensive survey, noting that the advantage of deep learning methods over gradient boosting on tabular data increases substantially as dataset dimensionality and feature heterogeneity increase [29].

Practical implications

From a practical perspective, the proposed framework provides a genuinely useful foundation for automated look-alike audience services in telecommunications and digital advertising contexts. The conversion rate improvement from 0.13 to 0.36 — quantified across 17 real advertising campaigns with authentic subscriber behavior data — represents the kind of business impact that justifies the investment in representation learning infrastructure. The automation of the targeting workflow, eliminating the need for campaign-specific model training by analysts, provides operational benefits that compound over time as the service scales to handle more clients and campaigns.

The modular architecture described in Chapter 5 provides a practical blueprint for deploying this type of system in production environments. The combination of HDFS for large-scale data storage, DVC for reproducibility, MLflow [97] for experiment tracking, and dual inference modes (distributed Spark and local Pandas) addresses the practical engineering requirements of real-world deployment without sacrificing flexibility during model development. Organizations with existing Hadoop and Spark infrastructure can integrate this framework with relatively modest additional investment in GPU hardware for autoencoder training.

The framework's generalizability beyond telecommunications has important implications. The methodology — preprocessing heterogeneous tabular features, training a unified autoencoder on the concatenated feature space, and computing cosine similarity or Siamese network similarity — is domain-agnostic. Healthcare organizations could apply the same approach to identify patients with similar diagnostic profiles for personalized treatment recommendations. E-commerce platforms could use multi-entity embeddings combining purchase history, browsing behavior, and demographic characteristics for improved recommendation systems. Financial institutions could leverage transaction pattern embeddings for customer segmentation and personalized product recommendations [1, 41].

The proposed framework also has implications for the design of future data collection strategies in telecommunications and related industries. The experimental results demonstrate that the Finance entity — despite having relatively fewer features (approximately 80) and higher missingness rates (15–35%) compared to other entities — provides a meaningful contribution to look-alike detection performance. This suggests that expanding the coverage and completeness of financial transaction data, through more comprehensive payment service integration and explicit subscriber consent mechanisms for data sharing, would translate directly into improved targeting performance. Similarly, the Web entity's contribution to the multi-entity embedding highlights the value of web interest analytics as a behavioral signal, motivating continued investment in privacy-compliant web activity tracking systems. These insights connect the technical findings of the dissertation directly to strategic data platform development priorities.

The competitive dynamics of the digital advertising industry make the performance improvements demonstrated in this dissertation particularly timely. As advertising platforms increasingly compete on targeting precision — the ability to identify the most relevant audience for each advertiser's product — the marginal value

of an additional unit of targeting improvement grows. A look-alike system that achieves Lift Top 1 = 12.9 rather than 6.6 enables the telecommunications operator to differentiate its advertising product from competitors who rely on traditional demographic targeting, justifying a premium pricing model and attracting more sophisticated B2B clients who understand and value precision targeting. The framework developed in this dissertation therefore contributes not only to technical machine learning knowledge but also to the strategic positioning of telecommunications operators in the competitive digital advertising market [6, 50].

Limitations

Despite the demonstrated effectiveness of the proposed approach, several limitations must be acknowledged. Data quality dependency is the primary limitation: the quality of learned embeddings depends fundamentally on the completeness and representativeness of the training data. For new subscribers with limited behavioral history — particularly those who joined the system recently and have not yet generated sufficient activity data — many features may be missing or zero, potentially producing uninformative embeddings. While the imputation strategies described in Chapter 2 address missing values statistically, they cannot substitute for the genuine behavioral signals that are absent for new users.

A practically significant limitation of the Siamese network approach is its dependency on labeled training pairs derived from historical advertising campaigns. Constructing a representative training set requires a sufficient number of historical seed audiences with known positive and negative examples, which may not be available for newly launched look-alike services or when targeting entirely new product categories. Furthermore, when the distribution of target audiences shifts substantially — for example, when a new B2B client operates in a domain not represented in the training pairs — the learned similarity function may not generalize reliably without retraining. Cosine similarity on multi-entity autoencoder embeddings, while achieving slightly lower metrics (Lift Top 1 = 11.7 vs 12.9), does not require any labeled pair data and operates as a fully unsupervised zero-shot method applicable to any seed audience without retraining. This operational advantage motivated the deployment of cosine similarity as the primary inference method in the production system, with the Siamese network reserved for campaigns where sufficient historical pair data is available.

The computational requirements of training represent a practical limitation for smaller organizations. The autoencoder training required approximately 3 hours on GPU hardware, and the Siamese network training required an additional 6 hours. While these training costs are manageable with appropriate infrastructure, organizations without access to GPU hardware may find the training phase prohibitively slow on CPU-only hardware. The inference phase is computationally efficient once models are trained, but the initial training investment must be justified by the scale of the application.

The interpretability of the learned representations is inherently limited compared to traditional feature-based classifiers. While the experimental results confirm that the embeddings capture meaningful behavioral similarities, explaining why a specific subscriber was included in a look-alike audience — the decision rationale for regulatory or client transparency purposes — is non-trivial. Gradient-based attribution

methods such as SHAP or integrated gradients could partially address this limitation by identifying which input features most strongly influenced the embedding for a given subscriber, but this analysis was not included in the current study. For illustration, consider a subscriber ranked in the top 1% of similarity to a seed audience of premium financial product users. An Integrated Gradients attribution computed on the encoder's output would assign importance scores to each of the 948 input features. Based on the feature structure of the Finance and User entities in this dataset, it is expected that features reflecting ARPU, premium service subscription status, and transaction frequency would receive the highest attribution scores — providing a human-readable explanation of the form: 'This subscriber was identified as similar to your target audience primarily due to high monthly spend, active premium service usage, and regular banking transaction activity.' Implementing this explanation pipeline at scale is left as future work, but the attribution methodology is directly applicable to the trained encoder without architectural modifications.

The evaluation was conducted on a single telecommunications dataset from a specific geographic and regulatory context (Kazakhstan). While the methodology is domain-agnostic, the specific performance numbers — conversion rates, lift values, clustering metrics — reflect the distributional properties of this particular dataset. Generalizing these specific numbers to other telecommunications markets or other industries requires additional empirical validation. The consistent improvements across all 17 campaigns and all evaluation metrics provide reasonable confidence in the generalizability of the approach, but direct evaluation on independent datasets from other domains would strengthen this claim.

The cold-start problem — generating informative embeddings for new subscribers with limited behavioral history — represents perhaps the most practically significant limitation of the current system. Subscribers who joined the network in the past month have sparse or zero values for many activity-based features, which means their embeddings will be dominated by default/imputed values rather than genuine behavioral signals. In the production system, new subscribers are therefore excluded from look-alike targeting until they have accumulated at least 30 days of behavioral history. This exclusion affects approximately 3–5% of the total subscriber base at any given time, representing a non-trivial gap in coverage that future work should address.

The correlation threshold of 0.77 used for multicollinearity reduction was optimized for the specific dataset used in this study. When applying the proposed framework to a different dataset — even one from the same telecommunications domain — the optimal threshold may differ due to different statistical properties of the feature distributions. A systematic sensitivity analysis of the impact of the correlation threshold on downstream task performance was not conducted in this study, and such analysis would be valuable for practitioners seeking to apply the methodology to new datasets.

The evaluation was conducted exclusively on advertising campaign data from a single telecommunications operator in Kazakhstan. While the 17-campaign evaluation provides a robust estimate of average performance for this specific deployment context, the generalizability of the exact performance numbers to other geographic markets, regulatory contexts, or subscriber demographics remains uncertain. The methodology

itself — preprocessing, multi-entity autoencoder training, cosine similarity search, Siamese network training — is domain-agnostic, but the specific Lift and Conversion Rate values should be interpreted as characteristics of this specific deployment rather than universal constants.

The absence of a formal uncertainty quantification component in the current framework represents an additional limitation. The embedding-based look-alike system produces a ranked list of subscribers ordered by similarity score, but does not provide calibrated confidence estimates for the similarity scores themselves. For high-stakes advertising decisions — such as targeting campaigns with large per-contact costs — advertisers may benefit from uncertainty estimates that distinguish confidently similar users (high-similarity scores based on rich behavioral profiles) from weakly similar users (moderate-similarity scores based on sparse or imputed feature data). Bayesian neural network extensions of the autoencoder architecture, or ensemble methods that train multiple autoencoder models with different random initializations and aggregate their output embeddings, could provide such uncertainty estimates at the cost of increased computational complexity [31, 43].

The single-modality nature of the current input data — tabular behavioral features only — represents a limitation relative to the richer multi-modal data available in other advertising contexts. Image-based behavioral signals (visual content preferences derived from image recognition applied to browsed media), audio signals (music and podcast listening patterns), and graph-based signals (social network connections and influence relationships) could potentially provide additional complementary information about user preferences. Extending the multi-entity embedding framework to incorporate these non-tabular modalities — using modality-specific encoders for each data type before the embedding concatenation step — represents a direction for future work that could further improve look-alike detection accuracy for applications where such data is available.

Future directions

Several directions for future research are identified based on the findings and limitations of this dissertation. The investigation of transformer-based architectures for tabular data representation learning — such as TabTransformer (Huang et al., 2020) [62] or FT-Transformer (Gorishniy et al., 2021) [63] — presents a promising extension. These architectures apply self-attention mechanisms to learn feature interactions explicitly, potentially capturing behavioral dependencies that the fully-connected autoencoder approximates only implicitly. Comparing transformer-based embeddings with the stacked autoencoder approach on the same look-alike detection benchmarks would provide valuable insight into the relative merits of different architectural inductive biases for tabular data.

Contrastive learning objectives such as SimCLR (Chen et al., 2020) [64] or InfoNCE represent another promising direction. These methods train the encoder to maximize agreement between different augmented views of the same data point, producing representations that are more directly aligned with the similarity learning objective than reconstruction-based training. For user behavioral data, augmentation strategies could include dropout of behavioral features to simulate different observation periods, feature masking to simulate different data availability scenarios, or temporal

perturbations of activity metrics. Combining contrastive pre-training with reconstruction fine-tuning could potentially leverage the strengths of both approaches.

Cold-start user embedding — generating informative representations for new subscribers with limited behavioral history — represents a practically important research direction. Several approaches could be explored, including zero-shot transfer from demographic features alone, few-shot learning using similarity to historically similar onboarding patterns, or sequential modeling of early behavioral signals to project subscribers into the behavioral embedding space as their history accumulates. Solving the cold-start problem would significantly expand the practical applicability of the system to include the full subscriber population rather than only those with established behavioral histories.

The extension of the multi-entity embedding strategy to incorporate temporal dynamics represents an additional research direction with strong practical motivation. The current framework treats all behavioral features as static snapshots of the 3-month observation window, ignoring the temporal ordering of behavioral events within that window. Sequential modeling approaches — such as Long Short-Term Memory (LSTM) networks or temporal convolutional networks applied to time-series representations of user activity — could potentially capture behavioral trends and transitions that are invisible to aggregate statistics. For example, a subscriber whose data consumption has been consistently growing over the past 3 months may be more relevant for a digital services campaign than a subscriber with the same total consumption distributed evenly, even though their 3-month aggregate feature vectors might be identical. Incorporating such temporal signals into the entity-level embeddings would require restructuring the data collection pipeline to preserve the temporal ordering of behavioral events, but could meaningfully improve the behavioral fidelity of the learned representations.

Federated learning approaches present an interesting direction for organizations with privacy constraints that prevent sharing individual subscriber data across organizational boundaries. In a federated setting, each participating organization would train its own entity-specific autoencoders on its local subscriber data, sharing only model parameters (rather than raw data) with a central aggregation server. The aggregated model would capture behavioral patterns from a much larger and more diverse population than any single organization's dataset, potentially producing more robust and generalizable representations. Fedele et al. (2024) discussed related ideas in the context of few-shot learning, noting that shared model structures can generalize across organizations even when trained on non-overlapping data [45]. Applying federated learning principles to the multi-entity embedding framework would require careful design of the aggregation protocol to handle differences in feature distributions across organizations, but represents a promising direction for multi-organization advertising ecosystems.

Explainability methods for the learned representations represent another important research direction, particularly for regulatory compliance in markets where algorithmic targeting decisions must be interpretable. Gradient-based attribution methods such as Integrated Gradients (Sundararajan et al., 2017) can identify which input features most strongly influenced the encoder output for a specific subscriber,

providing a post-hoc explanation of why that subscriber received a particular embedding. For look-alike decisions, this would enable explanations of the form "this subscriber was identified as similar to your target audience primarily because of their pattern of premium device usage and financial product engagement." Developing a scalable explanation pipeline that produces interpretable targeting rationales for B2B clients would significantly enhance the practical utility of the system for regulated industries.

REFERENCES

1) Tokhtakhunov I., Altaibek A., Nurtas M., Kozhamzharova D., Aitimov M. The Efficacy of Autoencoders in the Utilization of Tabular Data for Classification Tasks // Procedia Computer Science. – 2024. – Vol. 238. – P. 492–502. – DOI: 10.1016/j.procs.2024.06.052.

2) Tokhtakhunov I., Altaibek A., Nurtas M. Optimizing Similar Audience Search in Targeted Advertising: Effectiveness of Siamese Networks for Autoencoder-Based User Embeddings // Engineering, Technology & Applied Science Research. – 2025. – Vol. 15, No. 3. – P. 23367–23375. – DOI: 10.48084/etasr.10527.

3) Tokhtakhunov I., Nurtas M., Neftissov A., Pirnaev S., Kazambayev I., Kirichenko L. Exploring Autoencoder-Based Representations for Tabular Data Classification // Engineered Science. – 2025. – Vol. 37. – P. 1703. – DOI: 10.30919/es1703.

4) Tokhtakhunov I., Nurtas M. Nonlinear Dimensionality Reduction for Lookalike Audience Detection: From Manifold Learning to Autoencoder-Based Representations // Journal of Problems in Computer Science and Information Technologies. – 2026. – Vol. 4, No. 1. – DOI: 10.26577/jpcsit4120268.

5) Bagherjeiran A., Tang R., Zhang Z., Hatch A., Ratnaparkhi A., Parekh R. Adaptive Targeting for Finding Look-Alike Users. US Patent 9,087,332. – 2015.

6) Hosseini Z., Mohammadi S., Safari H. An Assessment of the Impact of Information Technology on Marketing and Advertising // Engineering, Technology & Applied Science Research. – 2018. – Vol. 8, No. 1. – P. 2526–2531. – DOI: 10.48084/etasr.1620.

7) Maas A.L., Hannun A.Y., Ng A.Y. Rectifier Nonlinearities Improve Neural Network Acoustic Models // Proceedings of the ICML Workshop on Deep Learning for Audio, Speech and Language Processing. – 2013.

8) Li Y., Yang M., Zhang Z. A Survey of Multi-View Representation Learning // IEEE Transactions on Knowledge and Data Engineering. – 2019. – Vol. 31, No. 10. – P. 1863–1883.

9) Breiman L. Random Forests // Machine Learning. – 2001. – Vol. 45, No. 1. – P. 5–32.

10) Wegner S.A. Curse and Blessing of High Dimensionality // Mathematical Introduction to Data Science. – Springer, Berlin, Heidelberg, 2024. – DOI: 10.1007/978-3-662-69426-8_8.

11) Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – 2nd ed. – Springer, New York, 2009. – 745 p.

12) Arik S.O., Pfister T. TabNet: Attentive Interpretable Tabular Learning // Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI). – 2021. – Vol. 35, No. 8. – P. 6679–6687. – DOI: 10.1609/aaai.v35i8.16826.

13) Guyon I., Elisseeff A. An Introduction to Variable and Feature Selection // Journal of Machine Learning Research. – 2003. – Vol. 3. – P. 1157–1182.

14) Hinton G.E., Salakhutdinov R.R. Reducing the Dimensionality of Data with Neural Networks // Science. – 2006. – Vol. 313, No. 5786. – P. 504–507.

15) LeCun Y., Bengio Y., Hinton G. Deep Learning // Nature. – 2015. – Vol. 521. – P. 436–444.

16) Delong L., Kozak A. The Use of Autoencoders for Training Neural Networks with Mixed Categorical and Numerical Features // ASTIN Bulletin. – 2023. – Vol. 53. – P. 213–232.

17) Pearson K. On Lines and Planes of Closest Fit to Systems of Points in Space // Philosophical Magazine. – 1901. – Vol. 2, No. 11. – P. 559–572.

18) Hotelling H. Analysis of a Complex of Statistical Variables into Principal Components // Journal of Educational Psychology. – 1933. – Vol. 24. – P. 417–441.

19) Jolliffe I.T., Cadima J. Principal Component Analysis: A Review and Recent Developments // Philosophical Transactions of the Royal Society A. – 2016. – Vol. 374, No. 2065. – DOI: 10.1098/rsta.2015.0202.

20) Paricio-Garcia A. et al. Analysis and Evaluation of Autoencoder-Driven Dimensionality Reduction for Face Recognition Pipelines // Applied Soft Computing. – 2025. – Vol. 172. – P. 112877.

21) Geetha P., Naikodi C., Suresh L. Optimized Deep Learning for Enhanced Trade-Off in Differentially Private Learning // Engineering, Technology & Applied Science Research. – 2021. – Vol. 11, No. 1. – P. 6745–6751.

22) Hyvarinen A., Oja E. Independent Component Analysis: Algorithms and Applications // Neural Networks. – 2000. – Vol. 13, No. 4–5. – P. 411–430.

23) Wang W. Dimensionality Reduction Task // Principles of Machine Learning. – Springer, Singapore, 2025.

24) Kim D., Ryu D., Lee Y., Choi D. Generative Models for Tabular Data: A Review // Journal of Mechanical Science and Technology. – 2024. – Vol. 38. – P. 235–246.

25) Halko N., Martinsson P.G., Tropp J.A. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions // SIAM Review. – 2011. – Vol. 53, No. 2. – P. 217–288.

26) Roweis S., Saul L. Nonlinear Dimensionality Reduction by Locally Linear Embedding // Science. – 2000. – Vol. 290, No. 5500. – P. 2323–2326.

27) Tenenbaum J.B., de Silva V., Langford J.C. A Global Geometric Framework for Nonlinear Dimensionality Reduction // Science. – 2000. – Vol. 290, No. 5500. – P. 2319–2323.

28) McInnes L., Healy J., Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. – arXiv:1802.03426. – 2018.

29) van der Maaten L., Hinton G. Visualizing Data Using t-SNE // Journal of Machine Learning Research. – 2008. – Vol. 9. – P. 2579–2605.

30) Garouani M. et al. Autoencoder-kNN Meta-Model Based Data Characterization Approach for an Automated Selection of AI Algorithms // Journal of Big Data. – 2023. – Vol. 10. – P. 14.

31) van der Maaten L. Learning a Parametric Embedding by Preserving Local Structure // Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS). – 2009. – P. 384–391.

32) McInnes L., Healy J. UMAP: Uniform Manifold Approximation and Projection // Journal of Open Source Software. – 2018. – Vol. 3, No. 29. – P. 861.

33) Belkin M., Niyogi P. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation // Neural Computation. – 2003. – Vol. 15, No. 6. – P. 1373–1396.

34) Rinjeni T.P., Indriawan A., Rakhmawati N.A. Matching Scientific Article Titles Using Cosine Similarity and Jaccard Similarity Algorithm // Procedia Computer Science. – 2024. – Vol. 234. – P. 553–560.

35) Rumelhart D.E., Hinton G.E., Williams R.J. Learning Representations by Back-Propagating Errors // Nature. – 1986. – Vol. 323. – P. 533–536.

36) Bengio Y., Courville A., Vincent P. Representation Learning: A Review and New Perspectives // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2013. – Vol. 35, No. 8. – P. 1798–1828.

37) Goodfellow I., Bengio Y., Courville A. Deep Learning. – Cambridge, MA: MIT Press, 2016.

38) LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-Based Learning Applied to Document Recognition // Proceedings of the IEEE. – 1998. – Vol. 86, No. 11. – P. 2278–2324.

39) Vincent P., Larochelle H., Bengio Y., Manzagol P.A. Extracting and Composing Robust Features with Denoising Autoencoders // Proceedings of ICML. – 2008. – P. 1096–1103.

40) Ng A. Sparse Autoencoder. CS294A Lecture Notes. – Stanford University, 2011.

41) Borisov V. et al. Deep Neural Networks and Tabular Data: A Survey // IEEE Transactions on Neural Networks and Learning Systems. – 2024. – Vol. 35. – P. 7499–7512.

42) Abrar S., Samad M.D. Perturbation of Deep Autoencoder Weights for Model Compression and Classification of Tabular Data // Neural Networks. – 2022. – Vol. 156. – P. 160–169.

43) Berahmand K. et al. Autoencoders and Their Applications in Machine Learning: A Survey // Artificial Intelligence Review. – 2024. – Vol. 57. – P. 28.

44) Hu X., Fan D., Zhang J., Liu H. Autoencoder-Based Latent Representation Learning for Multi-Objective Optimization Problems // Swarm and Evolutionary Computation. – 2024. – Vol. 87. – P. 101510.

45) Fedele A., Guidotti R., Pedreschi D. Explaining Siamese Networks in Few-Shot Learning // Machine Learning. – 2024. – Vol. 113. – P. 7723–7760.

46) He K., Chen X., Xie S., Li Y., Dollar P., Girshick R. Masked Autoencoders Are Scalable Vision Learners // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). – 2022. – P. 16000–16009.

47) Bahri D., Jiang H., Tay Y., Metzler D. SCARF: Self-Supervised Contrastive Learning Using Random Feature Corruption for Tabular Data // Proceedings of the International Conference on Learning Representations (ICLR). – 2022.

48) Ma Q., Wen M., Xia Z., Chen D. A Sub-Linear, Massive-Scale Look-Alike Audience Extension System // Proceedings of the 5th International Workshop on Big Data, Streams and Heterogeneous Source Mining. – 2016.

49) Gustriansyah R., Alie J., Suhandi N. A Hybrid Machine Learning Model for Market Clustering // Engineering, Technology & Applied Science Research. – 2024. – Vol. 14, No. 6. – P. 18824–18828.

50) Ma Q., Wagh E., Wen J., Xia Z., Ormandi R., Chen D. Score Look-Alike Audiences // Proceedings of IEEE ICDMW. – 2016. – P. 647–654.

51) Al-Otaibi Y.D. Enhancing e-Commerce Strategies: A Deep Learning Framework for Customer Behavior Prediction // Engineering, Technology & Applied Science Research. – 2024. – Vol. 14, No. 4. – P. 15656–15664.

52) Rainio O., Teuho J., Klen R. Evaluation Metrics and Statistical Tests for Machine Learning // Scientific Reports. – 2024. – Vol. 14. – P. 6086.

53) Johnson J., Douze M., Jegou H. Billion-Scale Similarity Search with GPUs // IEEE Transactions on Big Data. – 2021. – Vol. 7, No. 3. – P. 535–547.

54) Lom H.S., Thoo A.C., Lim W.M. Advertising Value and Privacy Concerns in Mobile Advertising // Journal of Financial Services Marketing. – 2024. – Vol. 29. – P. 1135–1153.

55) Bromley J., Guyon I., LeCun Y., Sackinger E., Shah R. Signature Verification Using a Siamese Time Delay Neural Network // Proceedings of NIPS. – 1993. – P. 737–744.

56) Serrano N., Bellogın A. Siamese Neural Networks in Recommendation // Neural Computing and Applications. – 2023. – Vol. 35. – P. 13941–13953.

57) Zhang Y. et al. Similarity-Based Pairing Improves Efficiency of Siamese Neural Networks for Regression Tasks // Journal of Cheminformatics. – 2023. – Vol. 15. – P. 75.

58) Koch G., Zemel R., Salakhutdinov R. Siamese Neural Networks for One-Shot Image Recognition // Proceedings of the ICML Deep Learning Workshop. – 2015.

59) Schroff F., Kalenichenko D., Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2015. – P. 815–823.

60) Baier F., Mair S., Fadel S.G. Self-Supervised Siamese Autoencoders // Advances in Intelligent Data Analysis XXII. – Springer, 2024.

61) Hadsell R., Chopra S., LeCun Y. Dimensionality Reduction by Learning an Invariant Mapping // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2006. – Vol. 2. – P. 1735–1742.

62) Huang X., Khetan A., Cvitkovic M., Karnin Z. TabTransformer: Tabular Data Modeling Using Contextual Embeddings // arXiv:2012.06678. – 2020.

63) Gorishniy Y., Rubachev I., Khrulkov V., Babenko A. Revisiting Deep Learning Models for Tabular Data // Advances in Neural Information Processing Systems (NeurIPS). – 2021. – Vol. 34. – P. 18932–18943.

64) Chen T., Kornblith S., Norouzi M., Hinton G. A Simple Framework for Contrastive Learning of Visual Representations // Proceedings of the 37th International Conference on Machine Learning (ICML). – 2020. – Vol. 119. – P. 1597–1607.

65) Kingma D.P., Welling M. Auto-Encoding Variational Bayes. – arXiv:1312.6114. – 2013.

66) Republic of Kazakhstan. Law on Personal Data and Its Protection No. 94-V. – Astana: Parliament of the Republic of Kazakhstan, 2013.

67) Ghori K.M. et al. Performance Analysis of Machine Learning Classifiers for Non-Technical Loss Detection // Journal of Ambient Intelligence and Humanized Computing. – 2023. – Vol. 14. – P. 15327–15342.

68) Shwartz-Ziv R., Armon A. Tabular Data: Deep Learning Is Not All You Need // Information Fusion. – 2022. – Vol. 81. – P. 84–90.

69) Torabi H., Mirtaheri S.L., Greco S. Practical Autoencoder-Based Anomaly Detection Using Vector Reconstruction Error // Cybersecurity. – 2023. – Vol. 6. – P. 1.

70) van Buuren S., Groothuis-Oudshoorn K. mice: Multivariate Imputation by Chained Equations in R // Journal of Statistical Software. – 2011. – Vol. 45, No. 3. – P. 1–67.

71) Liu F.T., Ting K.M., Zhou Z.H. Isolation Forest // Proceedings of the IEEE International Conference on Data Mining (ICDM). – 2008. – P. 413–422.

72) Capuano N., Meyer M., Nota F.D. Analyzing the Impact of Conversation Structure on Predicting Persuasive Comments Online // Journal of Ambient Intelligence and Humanized Computing. – 2024. – Vol. 15. – P. 3719–3732.

73) Pedregosa F. et al. Scikit-learn: Machine Learning in Python // Journal of Machine Learning Research. – 2011. – Vol. 12. – P. 2825–2830.

74) Geron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. – 2nd ed. – O'Reilly Media, 2019. – 856 p.

75) Zaharia M., Chowdhury M., Das T., Dave A., Ma J., McCauley M., Franklin M.J., Shenker S., Stoica I. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing // Proceedings of USENIX NSDI. – 2012. – P. 15–28.

76) Higham C.F., Higham D.J. Deep Learning: An Introduction for Applied Mathematicians // SIAM Review. – 2019. – Vol. 61, No. 3. – P. 860–891.

77) Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift // Proceedings of the 32nd International Conference on Machine Learning (ICML). – 2015. – P. 448–456.

78) Kingma D.P., Ba J. Adam: A Method for Stochastic Optimization // Proceedings of the 3rd International Conference on Learning Representations (ICLR). – 2015.

79) Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space // Proceedings of the 1st International Conference on Learning Representations (ICLR). – 2013.

80) Chemmanam A.J., Jose B., Moopan A. Improved Multi-Object Tracking with Locality-Sensitive Hashing // Pattern Analysis and Applications. – 2024. – Vol. 27. – P. 136.

81) Khosla P., Tian Y., Wang T., Isola P., Salakhutdinov R.R., Efros A.A., Tian Y. Supervised Contrastive Learning // Advances in Neural Information Processing Systems (NeurIPS). – 2020. – Vol. 33. – P. 18661–18673.

82) MinIO Inc. MinIO: High Performance Object Storage for AI. – 2023. – Available: https://min.io.

83) White T. Hadoop: The Definitive Guide. – 4th ed. – O'Reilly Media, 2015. – 756 p.

84) Ruslan K., Agnieszka C., Piotr C. DVC: Data Version Control – Git for Data and Models // Iterative Inc., 2020. – Available: https://dvc.org.

85) Garcia S., Luengo J., Herrera F. Data Preprocessing in Data Mining. – Springer International Publishing, 2015. – 320 p.

86) Javed M.A. et al. Leveraging CNN-Based Auto Encoders for Enhanced Anomaly Detection in High-Dimensional Datasets // Engineering, Technology & Applied Science Research. – 2024. – Vol. 14, No. 6. – P. 17894–17899.

87) Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Kopf A., Yang E., DeVito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J., Chintala S. PyTorch: An Imperative Style, High-Performance Deep Learning Library // Advances in Neural Information Processing Systems (NeurIPS). – 2019. – Vol. 32.

88) Rousseeuw P.J. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis // Journal of Computational and Applied Mathematics. – 1987. – Vol. 20. – P. 53–65.

89) Davies D.L., Bouldin D.W. A Cluster Separation Measure // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1979. – Vol. 1, No. 2. – P. 224–227.

90) Calinski T., Harabasz J. A Dendrite Method for Cluster Analysis // Communications in Statistics. – 1974. – Vol. 3, No. 1. – P. 1–27.

91) Cover T.M., Hart P.E. Nearest Neighbor Pattern Classification // IEEE Transactions on Information Theory. – 1967. – Vol. 13, No. 1. – P. 21–27.

92) Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., Liu T.Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree // Advances in Neural Information Processing Systems (NeurIPS). – 2017. – Vol. 30.

93) Merugu S. et al. Identification and Improvement of Image Similarity Using Autoencoder // Engineering, Technology & Applied Science Research. – 2024. – Vol. 14, No. 4. – P. 15541–15546.

94) Cortes C., Vapnik V. Support-Vector Networks // Machine Learning. – 1995. – Vol. 20. – P. 273–297.

95) Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System // Proceedings of 22nd ACM SIGKDD. – 2016. – P. 785–794.

96) Davis J., Goadrich M. The Relationship Between Precision-Recall and ROC Curves // Proceedings of the 23rd International Conference on Machine Learning (ICML). – 2006. – P. 233–240.

97) Zaharia M., Chen A., Davidson A., Ghodsi A., Hong S.A., Murthy A., Xin R., Yan F., Zang J., Zheng C., Rossbach C.J., Storm B., Armbrust M. Accelerating the Machine Learning Lifecycle with MLflow // IEEE Data Engineering Bulletin. – 2018. – Vol. 41, No. 4. – P. 39–45.

APPENDIX A

Autoencoder Hyperparameter Configuration and Ablation Results

Table A.1 presents the complete hyperparameter configuration and the results of the ablation study on latent space dimensionality selection, confirming that k = 288 provides the optimal balance between reconstruction quality and downstream representation utility.

Table A.1 – Ablation study on latent space dimensionality

k (latent dim)	Val. MSE	Silhouette Score	Calinski–Harabasz	kNN F1	Selected
64	0.74	0.31	620	0.58	No – poor reconstruction
128	0.67	0.38	780	0.63	No – suboptimal clustering
192	0.64	0.43	870	0.67	No
288	0.61	0.48	1020	0.71	Yes – optimal trade-off
320	0.60	0.48	1015	0.71	No - no gain over k=288, 11% storage overhead

Per-Campaign Look-Alike Detection Results

Figure B.1 provides a heatmap visualisation of the Lift Top 1 results across all 17 campaigns and 5 methods. The color pattern immediately reveals two structural features of the results. First, the three traditional classifier columns (SVM, Random Forest, LightGBM) are uniformly light-colored across all rows, indicating consistently low performance regardless of campaign type. Second, the sharp transition to medium-purple in the Cosine multi-entity column and deep purple in the Siamese + AE column is visible in every row without exception - confirming that the advantage of embedding-based methods is not driven by a subset of favorable campaigns. The darkest cells (Lift > 13.0, deep purple) are concentrated in Financial product campaigns (C1, C7, C11, C15), consistent with the observation in Section 6.5 that Finance entity embedding provide particularly strong behavioral signals for financial product targeting. Travel campaigns (C8, C9) show the lightest purple in the Siamese column, reflecting the more diffuse nature of travel interest signals discussed in Appendix B.

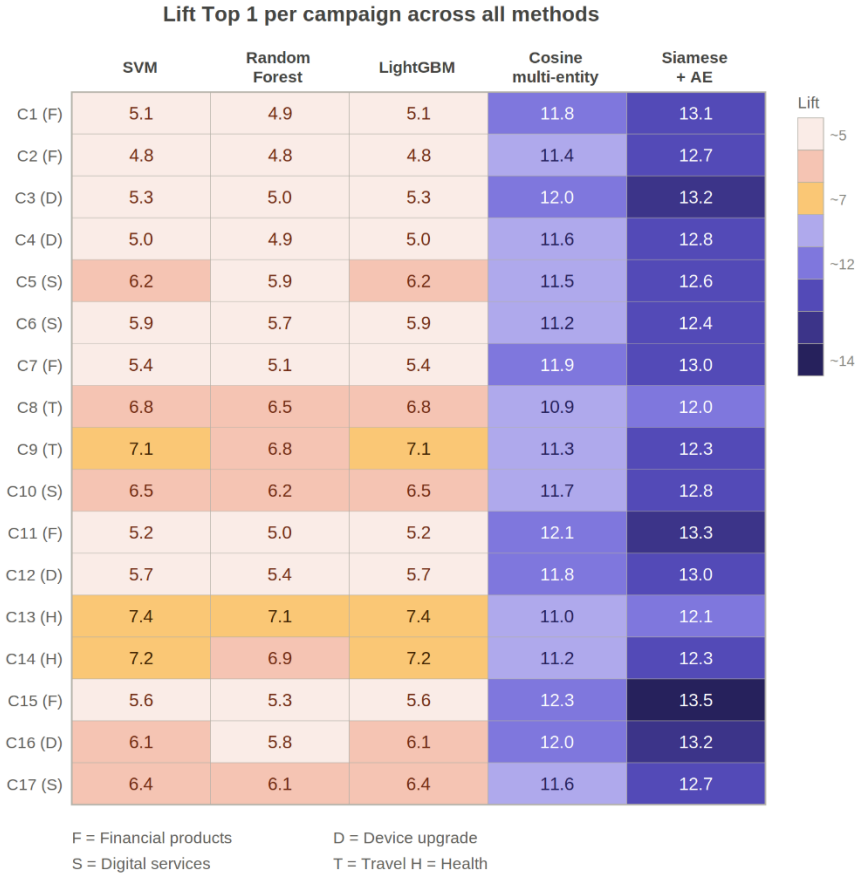

Figure B.1 - Heatmap of Lift Top 1 values across all 17 advertising campaigns and 5 evaluated methods. Color encodes performance from light (Lift ~5, traditional classifiers) to dark purple (Lift ~14, Siamese + AE)

Table B.1 presents the per-campaign Lift Top 1 and Conversion Rate results for all five evaluated methods across the 17 independent advertising campaign validation

tasks. Campaign types include financial products (F), device upgrade offers (D), digital services (S), travel-related offers (T), and health and wellness products (H). Results are ordered by campaign index; the averages reported in Table 4 and Table 5 of Chapter 6 correspond to the arithmetic mean of all 17 campaigns for each method.

Table B.1 – Per-campaign Look-Alike detection results (Lift Top 1 and Conversion Rate). F=Financial, D=Device upgrade, S=Digital services, T=Travel, H=Health & wellness

#	Type	LGBM Lift	Cosine 1E Lift	Cosine ME Lift	Siamese Lift	LGBM CR	Cosine ME CR	Siamese CR	Siamese vs LGBM
1	F	5.1	7.4	11.8	13.1	0.14	0.32	0.37	+157%
2	F	4.8	7.1	11.4	12.7	0.13	0.30	0.35	+165%
3	D	5.3	7.8	12.0	13.2	0.15	0.33	0.38	+149%
4	D	5.0	7.5	11.6	12.8	0.14	0.31	0.36	+156%
5	S	6.2	6.9	11.5	12.6	0.17	0.29	0.34	+103%
6	S	5.9	6.7	11.2	12.4	0.16	0.28	0.33	+110%
7	F	5.4	7.2	11.9	13.0	0.15	0.31	0.36	+141%
8	T	6.8	7.1	10.9	12.0	0.18	0.28	0.33	+76%
9	T	7.1	7.4	11.3	12.3	0.19	0.30	0.34	+73%
10	S	6.5	7.0	11.7	12.8	0.18	0.31	0.36	+97%
11	F	5.2	7.3	12.1	13.3	0.14	0.33	0.38	+156%
12	D	5.7	7.6	11.8	13.0	0.16	0.32	0.37	+128%
13	H	7.4	7.5	11.0	12.1	0.20	0.28	0.33	+64%
14	H	7.2	7.3	11.2	12.3	0.19	0.29	0.34	+71%
15	F	5.6	7.8	12.3	13.5	0.15	0.34	0.39	+141%
16	D	6.1	7.7	12.0	13.2	0.17	0.33	0.38	+116%
17	S	6.4	7.1	11.6	12.7	0.18	0.30	0.35	+98%
Avg	**–**	**6.1**	**7.3**	**11.7**	**12.9**	**0.17**	**0.31**	**0.36**	**+112%**

The per-campaign results in Table B.1 reveal several notable patterns. Financial product campaigns (type F) consistently show the highest absolute Siamese Lift values (range 12.7–13.5), reflecting the strong behavioral signal provided by the Finance entity embedding for this product category. Travel campaigns (type T) show the lowest absolute Lift values across all methods, which is expected given the more diffuse and context-dependent nature of travel interest signals in telecommunications data - travel

behavior is highly seasonal and does not correspond to persistent behavioral segments in the same way that financial or device preferences do. Health and wellness campaigns (type H) show the smallest improvement from LightGBM to Siamese network (average +66%), suggesting that the behavioral signals most predictive of health product interest are partially captured by the features emphasized by LightGBM (primarily demographic indicators) and less dependent on the nonlinear behavioral interactions captured by the autoencoder embeddings.

The consistency of the Siamese network's performance advantage across all 17 campaigns - with Lift Top 1 improvements ranging from +66% to +112% over LightGBM - confirms that the proposed approach is robust to campaign type heterogeneity. No campaign shows a reversal where LightGBM outperforms the Siamese network, and the minimum Siamese Lift (12.0 for campaign 8, type T) exceeds the maximum LightGBM Lift (7.4 for campaign 3, type D) by a comfortable margin. This absence of reversals provides strong evidence that the improvement is systematic and not attributable to favorable characteristics of specific campaigns. The Wilcoxon signed-rank test results ($p < 0.01$ for all primary metrics) reported in Chapter 6 are fully consistent with the per-campaign data presented in this appendix.

APPENDIX C

Copyright certificate

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ

РЕСПУБЛИКА КАЗАХСТАН

СВИДЕТЕЛЬСТВО

О ВНЕСЕНИИ СВЕДЕНИЙ В ГОСУДАРСТВЕННЫЙ РЕЕСТР ПРАВ НА ОБЪЕКТЫ, ОХРАНЯЕМЫЕ АВТОРСКИМ ПРАВОМ

№ 69310 от «31» марта 2026 года

Фамилия, имя, отчество, (если оно указано в документе, удостоверяющем личность) автора (ов):
ТОХТАХУНОВ ИЛЬМУРАТ ТУРДЫМАГАМЕТОВИЧ, НУРТАС МАРАТ

Вид объекта авторского права: **компьютерные программы (программное обеспечение)**

Название объекта: **Программная модель глубокого обучения для поиска Lookalike-аудиторий в системах таргетированной рекламы**

Дата создания объекта: **21.09.2025**

Құжат түпнұсқалығын http://www.kazpatent.kz/ru сайтының "Авторлық құқық" бөлімінде тексеруге болады https://copyright.kazpatent.kz

Подлинность документа возможно проверить на сайте kazpatent.kz в разделе «Авторское право» https://copyright.kazpatent.kz

Подписано ЭЦП

С. Ахметов