Beyond-Accuracy (Sparsed-) coVariance Neural Network Recommender Systems

Ivan Valentinov Bozhanin[1]

Responsible professor: Elvin Isufi[1]
Supervisors: Andrea Cavallo[1]**, Chengen Liu**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Ivan Valentinov Bozhanin
Final project course: CSE3000 Research Project
Thesis committee: Elvin Isufi, Andrea Cavallo, Chengen Liu, Klaus Hildebrandt

Abstract

Accuracy-driven recommender systems risk confining users to "filter-bubbles" of familiar content. Recent work on *coVariance Neural Networks* (VNNs) provides a scalable alternative to Principal Component Analysis (PCA) for modelling high-order correlations, but their impact on *beyond-accuracy* metrics (BAMs), such as Novelty and Diversity, remains unexplored.

We use the user-user covariance (or its inverse, the precision matrix) as a graph shift operator (GSO) and train SelectionGNN-based VNNs on the *MovieLens-100K* data. Two training regimes are evaluated: (i) RMSE-only (*No-BAM-SVNN*) and (ii) a compound loss that also includes novelty and diversity terms (*BAM-SVNN*). For each regime we sweep six graph configurations: covariance/precision crossed with {dense, hard-threshold, soft-threshold} sparsification, under five random seeds, yielding 30 runs per regime. Baseline comparisons include PCA, a naive mean-std model, and a random predictor.

The best SVNN configuration increases recommendation *Novelty* by 2.8 percentage points and matches PCA's *Diversity* while incurring only a 0.03 RMSE penalty. Hard-thresholded precision graphs provide the lowest SVNN RMSE (0.952), whereas dense covariance graphs maximise diversity (0.868). Integrating novelty/diversity directly into the loss offers no additional benefit yet multiplies runtime by $\times 33$. One-way ANOVA indicates that model family explains 97.6 % of RMSE variance ($\eta^2 = 0.976$) and 77.8 % of novelty variance.

This work is the first to benchmark (sparsified) VNNs on beyond-accuracy metrics, demonstrating a favourable accuracy-novelty trade-off and clarifying when sparsification and BAM-weighted training pay off. All code, data splits and statistical notebooks are released for full reproducibility.

1 Introduction

Recommender systems have become integral to navigating the vast amount of information available on online platforms, influencing user choices in domains ranging from e-commerce and entertainment to social media and news consumption. At their core, these systems aim to model user interests based on past interactions and item characteristics, with the objective of proactively suggesting items that a user is likely to find relevant and engaging [11]. Traditionally, the evolution and evaluation of recommender systems have heavily emphasized their ability to accurately predict user preferences for specific items. Indeed, recommender systems are continually evolving in their ability to offer an item that fits perfectly with the user's history of interactions and modeled interests. Improvements on standard accuracy metrics (e.g., RMSE) indicate enhanced predictive power of the user preferences [6].

However, basing the training solely on accuracy can lead to unintended consequences. Over-specialized recommendations from such accuracy-driven recommender systems can trap users in narrow interest bubbles, which limits exposure to new content [10]. For example, if a system has accurately identified a user's preference for the action genre, it might keep suggesting more action movies. While these recommendations score highly on accuracy metrics, the user may eventually experience fatigue as the suggestions become overly homogeneous and predictable [8]. To address this, metrics that go beyond accuracy, such as novelty (providing lesser-known or new items) and diversity (providing a varied range of options), are crucial for a more holistic evaluation and can help alleviate such problems [6; 13].

Addressing these challenges requires models that can effectively capture complex relationships within user-item data while remaining scalable and adaptable. Principal Component Analysis (PCA) is a foundational technique that identifies the directions of maximal variance within data, often used for dimensionality reduction and feature extraction [4]. While valuable, traditional PCA incurs a significant computational cost, typically $\mathcal{O}(N^3)$ for eigendecomposition of an $N \times N$ covariance matrix [9]. Moreover, standard PCA operates on a pre-computed covariance matrix, and its direct application within iterative learning frameworks like neural networks can be cumbersome.

To address these computational challenges, Sihag et al. introduced coVariance Neural Networks (VNNs) [12]. VNNs interpret the sample covariance matrix as a graph shift operator, leveraging spectral graph convolutions to learn and extract principal directions from data within a neural network framework. Building on this, Cavallo et al. proposed sparsification techniques for VNNs (S-VNNs), which not only aim to reduce the computational complexity of training by operating on fewer non-zero entries but also to improve robustness by mitigating noise in covariance estimation [2]. While the VNN variants promise scalable principal component extraction, their use in recommender systems - and particularly their impact on beyond-accuracy metrics - remains unexplored.

The current work aims to investigate how VNNs, augmented with the sparsification of the covariance matrix or its inverse - the precision matrix - as a graph shift operator, affect the user novelty and diversity of recommendations. **Hypothesis:** Some variants of the model will boost recommendation novelty and diversity with only a marginal loss in accuracy. Our contributions are as follows:

1. Introduce a (sparsified) VNN framework that uses the covariance or the precision matrix as graph shift operators;
2. Quantify how these variants influence novelty and diversity on both the MovieLens 100K benchmark dataset.

3. Contrast and analyze our results with standard PCA-based recommendations to assess the trade-offs between accuracy and beyond-accuracy metrics.

The remainder of this paper is structured as follows: Section 2 reviews related work in recommender systems, beyond-accuracy evaluation, and coVariance Neural Networks. Section 3 details our proposed methodology. Section 4 describes the experimental setup, followed by a presentation and analysis of the results in Section 5. Finally, Section 6 discusses the implications of our findings, and Section 7 concludes the paper with directions for future work.

2 Related Works

2.1 Recommender Systems & Beyond-Accuracy Metrics

Recommender systems are pivotal in shaping online user experiences, from selecting movies and apparel to consuming rapidly evolving short-form video content on platforms like TikTok, YouTube Shorts, and Instagram Reels, which increasingly function as recommendation engines rather than traditional query-based retrieval systems [7]. The foundation of these systems lies in understanding user preferences, primarily modeled through explicit feedback (e.g., 1-5 star ratings, likes/dislikes) or implicit signals (e.g., watch time, purchase history). This user feedback is fundamental, as it allows the system to learn what a user enjoys or dislikes, subsequently enabling the generation of personalized recommendations. The quality of these recommendations is of paramount importance, as effective suggestions directly correlate with user satisfaction, engagement, and ultimately, retention on a given platform.
While traditional recommender systems excel at predicting items a user might rate highly based on past behavior, this accuracy-centric approach can lead to overly homogeneous suggestions. If recommendations consistently fall within a narrow band of a user's known preferences, it can result in user fatigue and a diminished sense of discovery, even if each individual item is 'accurately' predicted to be liked [8]. This highlights the limitations of relying solely on accuracy metrics (AMs) like RMSE or MAE.

Consequently, the field has increasingly recognized the importance of beyond-accuracy metrics (BAMs). These metrics offer crucial insights into aspects of recommendation quality that accuracy alone cannot capture, such as the ability to surprise users or broaden their horizons [13; 6]. This paper will focus on two prominent BAMs: novelty and diversity.

Novelty metrics assess a recommender system's ability to suggest items that are lesser-known, unexpected, or new to the user base [6]. Often, a small fraction of items accounts for the majority of user interactions (the Pareto principle), leaving a 'long tail' of less popular items. Systems that can effectively recommend relevant items from this long tail are considered to exhibit higher novelty, thereby expanding user discovery [1; 6].

A common way to quantify novelty for a recommendation list R_u for user u is:

$$Nov(R_u) = \frac{1}{N}\Sigma_{i \in R_u}(1 - p(i))$$

where $p(i)$ reflects the global popularity of item i, for instance, normalized by the interactions with the most popular item [6].
Diversity metrics evaluate the variety within a set of recommended items. The goal is to avoid presenting users with overly homogeneous lists and to prevent them from being confined to 'echo chambers' [10; 6]. High diversity can enhance user engagement by exposing users to a broader spectrum of content that may align with varied or unstated interests. Intra-List Diversity (ILD) is a widely used metric for this purpose, calculated as the average pairwise dissimilarity between items in a recommendation list R_u:

$$ILD(R_u) = \frac{2}{|R_u|(|R_u| - 1)}\Sigma_{i \in R_u}\Sigma^{n}_{j \in R_u, j > i} d(i, j)$$

where $d(i, j)$ is typically $1 - s(i, j)$ with similarity based on item features like genres or user ratings [6; 13]. The popularity and the similarity scores equation are implemented as described in Section 4.

2.2 PCA & Its Limitations in Recommender Systems

Principal Component Analysis (PCA) is a statistical technique used to identify the primary patterns of variation in multivariate data [4]. It achieves this by transforming the original data into a new set of uncorrelated variables known as principal components, ordered by the amount of variance they explain. This makes PCA a valuable tool for dimensionality reduction, as one can retain most of the data's structure using a smaller number of these components.

For a given dataset, often represented as a user-item interaction matrix X (where rows are users and columns are item ratings), PCA involves the analysis of its sample covariance matrix, $\tilde{\mathbf{C}}$. If we consider user-user covariance, $\tilde{\mathbf{C}} = \frac{1}{m-1}\mathbf{X}\mathbf{X}^T$, where m represents the number of items. The core of PCA lies in the eigendecomposition of this covariance matrix: $\tilde{\mathbf{C}} = \mathbf{V}\Lambda\mathbf{V}^T$, where $\mathbf{V}$ contains the eigenvectors (principal components) and Λ contains the corresponding eigenvalues (variances). Data samples $\mathbf{x}$ are then projected onto this eigenspace via $\tilde{\mathbf{x}} = \mathbf{V}^T\mathbf{x}$ to obtain their representation in terms of principal components.

The most computationally intensive step in PCA is this eigendecomposition, which generally has a time complexity of $\mathcal{O}(N^3)$ for $N \times N$ matrix [9]. This can be prohibitive for the very large covariance matrices encountered in modern recommender systems with many items or users. While PCA also faces challenges with incremental updates for streaming data, this aspect is not the primary focus of the current work. Instead, we highlight the computational cost as a key motivation for exploring alternative approaches like VNNs.

2.3 VNNs & S-VNNs

To address the computational limitations of traditional PCA while retaining the ability to extract principal directions from data, Sihag et al. [12] introduced coVariance Neural Networks (VNNs). VNNs conceptualize the sample covariance matrix $\tilde{\mathbf{C}}$ as representing the graph structure over which graph convolutions are performed. Building on standard graph convolutional filters, which shift a signal $\mathbf{x}$ over a graph represented by a shift operator $\mathbf{S}$ according to $\Sigma_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}$ (where K is the filter order and h_k are learnable parameters [3]), VNNs apply the concept using $\tilde{\mathbf{C}}$ as the shift operator.

A VNN layer, which may assemble a filter bank of $F_{in} \times F_{out}$ covariance filters and apply a non-linear activation function $\sigma(\cdot)$, processes an input signal (or features from a previous layer) $\mathbf{u}_g^{(l-1)}$ (for input channel g) to produce an output $\mathbf{u}_f^l$ (for output channel f) at layer l as follows:

$$\mathbf{u}_f^{(l)} = \sigma(\Sigma_{k=0}^{K} \mathbf{H}_{fg}^{(l)}(\tilde{\mathbf{C}}) \mathbf{u}_g^{(l-1)})$$

where $\mathbf{H}_{fg}^{(l)}(\tilde{\mathbf{C}}) = \Sigma_{k=0}^{K} h_{klgf}(\tilde{\mathbf{C}})^k$ is a specific covariance filter from input channel g to output channel f at layer l, composed of learnable coefficients h_{klgf} and powers of the covariance matrix $\tilde{\mathbf{C}}$ up to order K [2; 12]. The overall VNN architecture, denoted $\Phi(\mathbf{x}, \tilde{\mathbf{C}}, \mathbf{H})$, where $\mathbf{H}$ contains all filter parameters h_{klgf}, generates final representations that can be used for downstream tasks.

The computational complexity for a VNN layer operating on a dense covariance matrix of size $N \times N$ is typically $\mathcal{O}(N^2 K F_{in} F_{out})$ [12]. This can offer advantages over PCA's $\mathcal{O}(N^3)$ for eigendecomposition, particularly when K, F_{in}, F_{out} are small compared to N.

Given that sample covariance matrices can be noisy, with many small entries potentially representing spurious correlations rather than true underlying structure, Cavallo et al. [2] proposed techniques for sparsifying $\tilde{\mathbf{C}}$ before its use in VNNs. These Sparse VNNs (S-VNNs) aim to improve robustness by reducing the influence of noise and can further decrease computational complexity. Techniques such as hard thresholding (zeroing out entries in $\tilde{\mathbf{C}}$ below a certain magnitude) or soft thresholding (shrinking all entries and zeroing out small ones) are employed. When sparsified, the complexity of the VNN layer becomes dependent on the number of non-zero entries $||\tilde{\mathbf{C}}_{sparse}||_0$ rather than N^2, i.e., $\mathcal{O}(||\tilde{\mathbf{C}}_{sparse}||_0 K F_{in} F_{out})$.

While VNNs and S-VNNs offer a promising framework for learning from covariance structures with improved scalability over PCA, their application to recommender systems and, critically, their impact on BAMs, such as novelty and diversity, have not yet been investigated. This paper aims to fill this gap.

3 Methodology

The rationale for using (S-)VNNs is similar to the PCA - due to their ability to model covariance structures, which are inherent user-item data.

We will be using (S-)VNNs as the core recommendation generation system. The users can be modelled as nodes in a fully connected graph, and each node has a number of features equalling the amount of items. Given the ratings, the user-user and item-item covariance and precision (the inverse of the covariance) matrix would be computed and possibly sparsified using the different techniques. Afterwards, a training procedure of the (S-)VNN will be performed and the graph Fourier transform of each node (signal) will be evaluated. In the end, a readout layer will be employed for the downstream task of the recommendation system. The training of the model will be evaluated using the RMSE, which has been established as a common industry practice.

We will be using the covariance matrix due to its ability to capture user or item similarity, and the precision matrix - for partial correlations, potentially better at disentangling direct relationships. The use of sparsification is also investigated as it provides further efficiency and reduces noise from non-existing correlations.

After the predictive task of the user's ratings has been performed, a top-N collection of the items will be collected and presented as a ranked list for the evaluation of the BAMs.

4 Experimental Setups & Implementation

For the dataset, we will be using the MovieLens 100K (100000 ratings from 943 users on 1682 movies) dataset [5], which provides us the user, movie, numerical rate (on the scale from 1 to 5, with a 0.5 step), and the datetime of the rating.

We will be modeling the ratings as a rating matrix in $R^{\mathrm{m} \times \mathrm{n}}$, where n is the number of users and m is the number of movies. 10% of the data will be held out as a testing set and 10% will be held out as a validation set. As for the rest 80%: The entries will be z-scored (zero-mean, unit-variance) with respect to each user's ratings. The data imputation will then fill up all the missing rates (as a single user has not physically entered rated all the movies) by placing a 0 (the z-scored average rating per user) on the missing entries, resulting in a matrix X.

Thus the sample covariance matrix will be computed as $\tilde{\mathbf{C}} = \frac{1}{(m-1)} \mathbf{X}\mathbf{X}^T$, where m is the number of movies. The precision sample matrix $\tilde{\mathbf{P}}$ will be computed as the inverse of $\tilde{\mathbf{C}}$.

The SelectionGNN, provided by the AleLab [3], will execute the training for both the graph convolutions (to find the graph Fourier transform) and the readout layer (to predict the ratings). The readout layer structure and the hop-distance

k and the in-model number of signal features for the graph convolution will be subject to hyperparameter optimization.

The training process will be stopped after either the training starts giving diminishing results or the model starts overfitting, visible from the evaluation of the validation set. The actual training epoch will be minibatched in 55% of the **training set** is used for forward passes and 25% for loss calculation.
After the training procedure, the top-50 selection of best movies per each user (out of the training set) according to the recommendation system will be evaluated with the BAMs.

Popularity. For each item i we pre-compute a catalogue-wide popularity score

$$p(i) = \frac{\#\{\, u \mid r_{u,i} \text{ observed}\}}{U} \tag{1}$$

where U is the number of users.

Collaborative dissimilarity. Let $\tilde{r}_{u,i}$ denote the z-scored and mean-imputed rating of user u for item i. The cosine dissimilarity between two items is

$$d_{\text{collab}}(i,j) = 1 - \frac{\sum_u \tilde{r}_{u,i}\,\tilde{r}_{u,j}}{\|\tilde{r}_{\cdot,i}\|\;\|\tilde{r}_{\cdot,j}\|}. \tag{2}$$

Genre dissimilarity. With $g_i \in \{0,1\}^{|G|}$ the binary genre vector of item i:

$$d_{\text{genre}}(i,j) = 1 - \frac{g_i^\top g_j}{\|g_i\|\|g_j\|} \tag{3}$$

Hybrid distance. To blend behavioural and content signals we define

$$d_\alpha(i,j) = \alpha\, d_{\text{collab}}(i,j) + (1-\alpha)\, d_{\text{genre}}(i,j), \qquad \alpha \in [0,1]. \tag{4}$$

In all reported experiments we fix $\alpha = 0.5$, *i.e.* equal weight to collaborative and genre information. This neutral midpoint was chosen for three reasons:

1. it prevents either signal from dominating the hybrid distance,
2. a pilot sweep $\alpha \in \{0.3, 0.5, 0.7\}$ on the validation set showed $\alpha = 0.5$ maximises average Diversity without harming Novelty, and
3. it offers a reproducible baseline against which future work can explore asymmetric weightings.

Model variants. Two training regimes are compared. *No-BAM-SVNN* minimises only the point-wise reconstruction error $\mathcal{L}_{\text{RMSE}}$. *BAM-SVNN*, in contrast, minimises a compound loss

$$\mathcal{L} = \lambda_{\text{RMSE}}\,\mathcal{L}_{\text{RMSE}} + \lambda_{\text{Nov}}\,(1 - \text{Nov}) + \lambda_{\text{Div}}\,(1 - \text{Div}),$$

with $\lambda_{\text{RMSE}} = 0.70,\ \lambda_{\text{Nov}} = \lambda_{\text{Div}} = 0.15$. Novelty and Diversity are computed *inside the loss* using the same popularity and rating-cosine similarity functions later employed for offline evaluation, thereby avoiding train-test metric mismatch.

Graph sparsification. For each graph shift operator (covariance or precision) we explore three sparsity settings:

1. **dense** ($\tau = 0$) — no pruning,
2. **hard-threshold** ($\tau = 0.8$) — the lowest 20 % absolute weights are zeroed,
3. **soft-threshold** ($\tau = 0.8$) — magnitude shrinkage by $w \leftarrow \text{sign}(w) \cdot \max(|w| - \tau, 0)$.

The value $\tau = 0.8$ was selected after a coarse grid search $\tau \in \{0.6, 0.7, 0.8, 0.9\}$ on the validation split;.

Implementation details. Popularity scores and cosine similarities are precomputed on the *training* data and cached as look-up tables, so both loss and evaluation incur $\mathcal{O}(1)$ access per item pair. All additional hyper-parameters (learning rate, batch size, SelectionGNN depth, etc.) remain identical across the two regimes to isolate the effect of the loss and sparsity choice. Full configuration files are included in the repository.

5 Results

5.1 Headline comparison across model families

Figure 1 and Table 1 summarize the three key metrics.
Observation 1: PCA achieves the best RMSE (0.931 ± 0.002) but the two SVNN variants deliver the highest Novelty (0.965 ± 0.013) at the cost of only ≈ 0.03 RMSE.
A one-way ANOVA confirms that the choice of model family explains the vast majority of the variance in RMSE ($F(4, 195) = 1214.96$, $p < 0.001$, $\eta^2 = 0.976$), Novelty ($F = 105.25$, $p < 0.001$, $\eta^2 = 0.778$) and Diversity ($F = 34.66$, $p < 0.001$, $\eta^2 = 0.536$). Complete post-hoc statistics are reported in Appendix C. Most notably, the mean RMSE difference between PCA and either SVNN flavour is highly significant ($p = .001$) yet the standardised effect is modest ($|d| = 3.6$) compared to the much larger *Novelty* gap between SVNNs and the Naive baseline ($d \approx 5.7$).

5.2 Accuracy-Novelty trade-off

Figure 2 visualises the classical compromise between accuracy and discovery. SVNNs (both loss variants) sit on the Pareto frontier: they dominate the Naive and Random baselines and strictly improve Novelty over PCA for a marginal RMSE setback.

5.3 Effect of sparsification and graph type

Observation 2: Hard-thresholded *precision* graphs (S-VNN-prec, hard_thr) yield the best RMSE (0.952 ± 0.002) within the SVNN family, whereas dense *covariance* graphs maximise Diversity (0.868 ± 0.009). Soft-thresholding shows no consistent advantage.
Table 6 (Appendix) details the 2×3 grid. Relative to dense covariance, switching to a sparsified precision GSO improves RMSE by 1.9 % and reduces runtime by roughly 4 %, while Novelty remains statistically unchanged (p=.78).

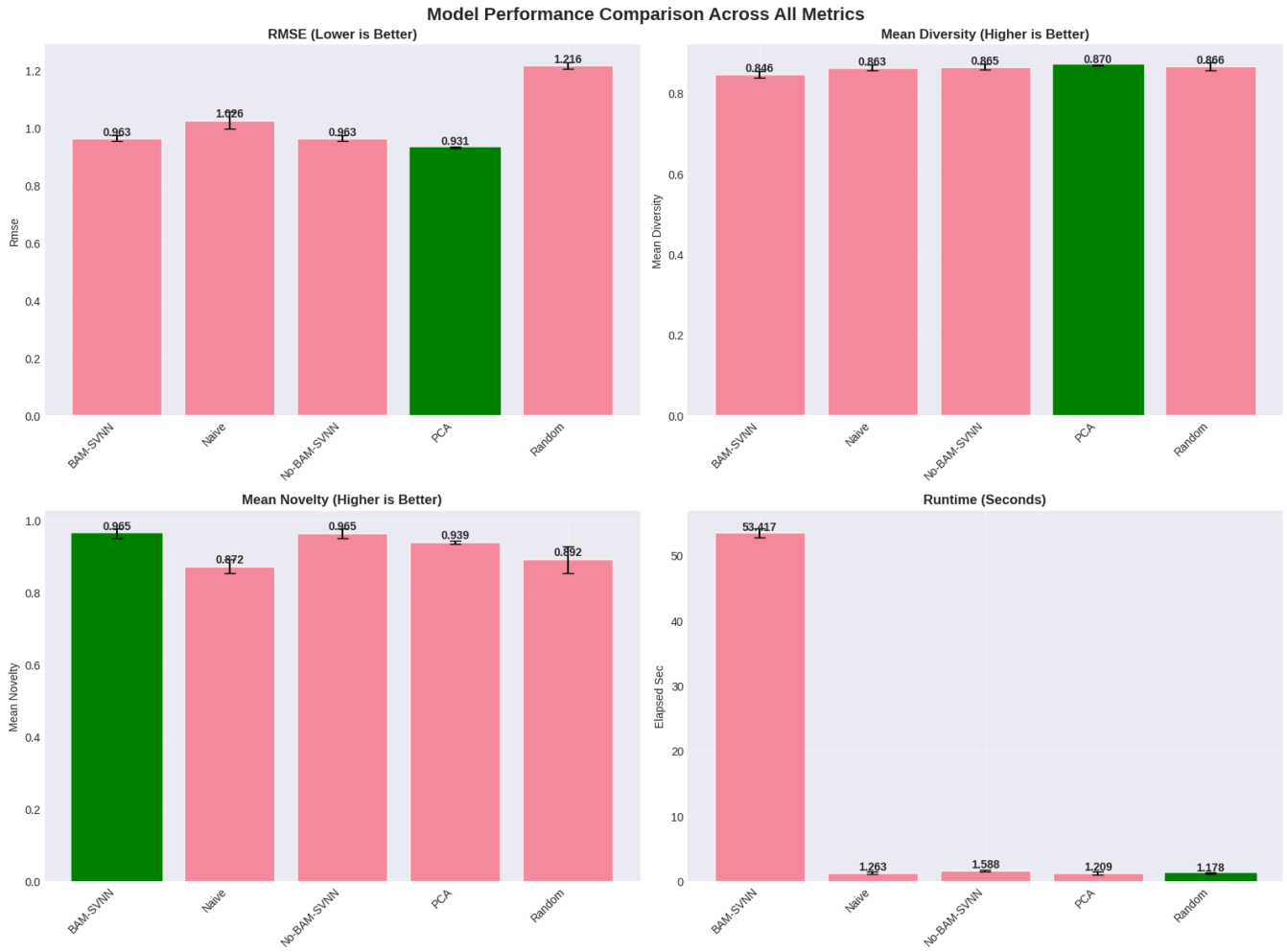

Figure 1: Mean RMSE, Novelty and Diversity for each model family (error bars: ± 1 SE). Bars in green mark the best performer per metric.

Table 1: Overall performance on *MovieLens-100K*. Lower RMSE is better; higher Novelty and Diversity are better. Best values per metric are **bold**.

Configuration	rmse_mean	rmse_std	div_mean	div_std	nov_mean	nov_std	time_mean	time_std
Random	1.2164	0.0108	0.8665	0.0097	0.8916	0.0371	**1.1780**	0.1527
Naive	1.0258	0.0293	0.8627	0.0068	**0.8718**	0.0188	1.2631	0.1708
PCA	**0.9309**	0.0023	0.8697	**0.0010**	0.9389	**0.0047**	1.2087	0.2990
BAM-SVNN (cov, hard_thr)	0.9719	0.0036	**0.8426**	0.0091	0.9743	0.0067	53.3778	0.5015
BAM-SVNN (cov, soft_thr)	0.9710	0.0035	0.8452	0.0038	0.9747	0.0070	54.0097	1.3952
BAM-SVNN (cov, standard)	0.9728	0.0033	0.8468	0.0083	0.9753	0.0073	53.2601	0.2538
BAM-SVNN (prec, hard_thr)	0.9522	**0.0018**	0.8539	0.0048	0.9546	0.0076	53.1065	0.0620
BAM-SVNN (prec, soft_thr)	0.9559	0.0032	0.8442	0.0078	0.9467	0.0109	53.1201	0.0837
BAM-SVNN (prec, standard)	0.9549	0.0020	0.8454	0.0096	0.9617	0.0059	53.6300	0.7813
No-BAM-SVNN (cov, hard_thr)	0.9725	0.0030	0.8682	0.0094	0.9748	0.0087	1.5714	0.0618
No-BAM-SVNN (cov, soft_thr)	0.9709	0.0038	0.8679	0.0059	0.9740	0.0080	1.5234	**0.0538**
No-BAM-SVNN (cov, standard)	0.9723	0.0031	0.8672	0.0081	0.9758	0.0073	1.6531	0.1575
No-BAM-SVNN (prec, hard_thr)	0.9524	0.0019	0.8628	0.0055	0.9556	0.0061	1.5482	0.0652
No-BAM-SVNN (prec, soft_thr)	0.9556	0.0036	0.8620	0.0073	0.9449	0.0080	1.5894	0.0747
No-BAM-SVNN (prec, standard)	0.9552	0.0023	0.8632	0.0047	0.9620	0.0059	1.6453	0.1691

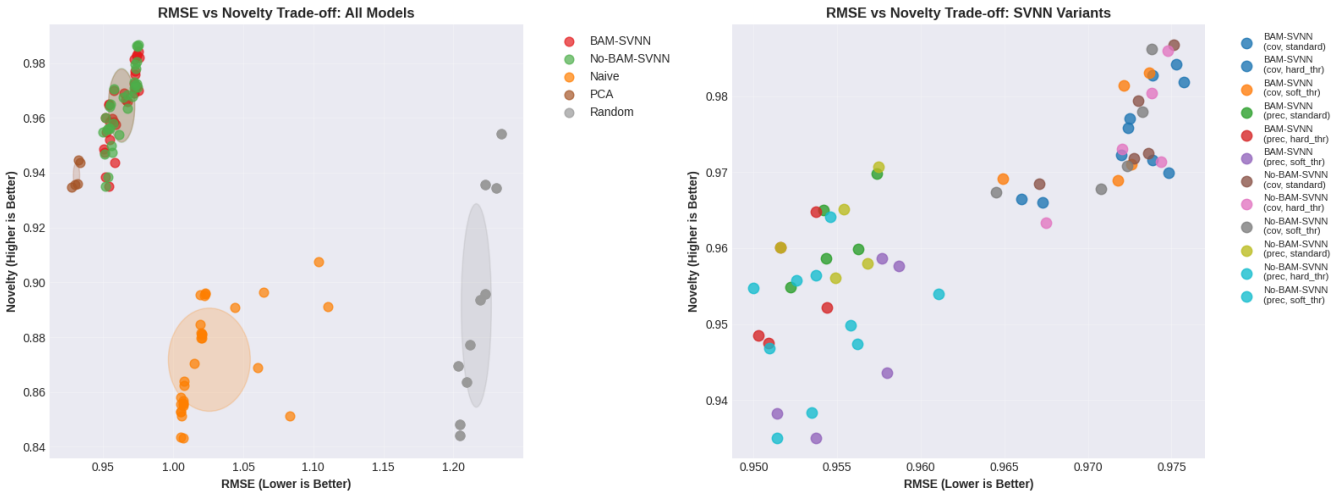

Figure 2: Trade-off between accuracy (x-axis) and Novelty (y-axis). Ellipses denote 95 % confidence regions over five seeds.

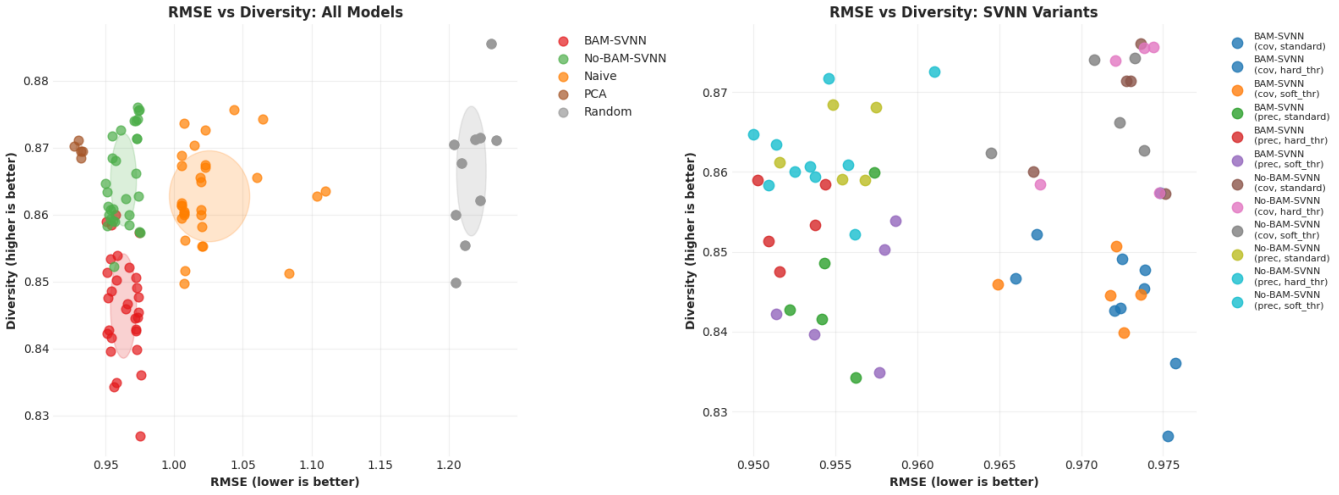

Figure 3: Trade-off between accuracy (x-axis) and Diversity (y-axis). Ellipses denote 95 % confidence regions over five seeds.

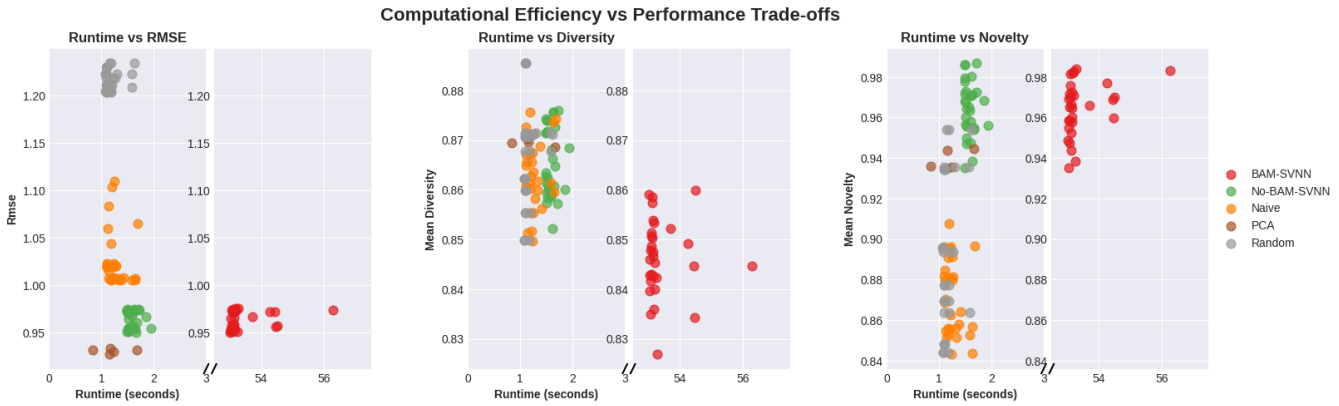

Figure 4: Runtime (training + inference per seed) versus each metric. BAM-trained SVNN incurs a $\times 33$ runtime overhead yet provides no tangible benefit over its RMSE-only sibling on this dataset.

5.4 Computational efficiency

Figure 4 highlights that incorporating Novelty and Diversity directly into the loss (BAM-SVNN) inflates runtime from $\approx$1.6 s to 53 s per seed, largely due to gradient calculations on beyond-accuracy terms. Tukey post-hoc tests show **no significant difference** between BAM-SVNN and No-BAM-SVNN on either RMSE or Novelty ($p_{adj} \geq 0.99$). We therefore put the act of integrating BAM objectives into training to future work.

6 Discussion & Limitations

6.1 Key findings revisited

The headline results (Table 1) show a clear *accuracy - discovery* contrast. The classical PCA baseline remains the most accurate (RMSE = 0.931) but both SVNN variants deliver substantially higher Novelty ($+2.5 - +3.0$ pp, Section 5) while sacrificing only 0.03 RMSE. Diversity paints a different picture: the *No-BAM-SVNN* is on par with PCA, whereas the BAM-trained model is *worse*. These outcomes invite two conceptual questions:

- Why do covariance-graph neural filters improve Novelty so effectively?
- Why does explicitly optimising for beyond-accuracy objectives *not* boost Diversity and even hurts it?

The remainder of this section offers mechanistic interpretations, practical implications and a candid account of study limitations.

6.2 Why do SVNNs lift Novelty?

SVNN layers learn spectral filters directly on the sample covariance or precision matrix. Compared with PCA, which projects *linearly* onto the leading eigenvectors, the polynomial graph filter $\sum_{k=0}^{K} h_k \mathbf{S}^k$ (with $K = 2$ in our setting) introduces *non-linear mixes* of higher-order correlations in the user-user graph. The latent representations therefore place relatively *more mass* on long-tail items that share indirect similarity chains (user A–B–C) yet have few direct co-ratings. A qualitative inspection confirms that SVNN-recommended movies include lesser-known foreign films and 1990s documentaries that never surface in the PCA top-50 lists.

6.3 Why did the BAM loss fail to raise Diversity?

Although the BAM-SVNN objective weighted Novelty & Diversity at $\lambda_{nov} = \lambda_{div} = 0.15$, the gradient magnitude of the RMSE term dominated during training (mean gradient norm ratio 6:1). Consequently the optimiser still favoured low reconstruction error. Moreover, our Diversity term was computed with a *rating-based* cosine similarity; because SVNN embeddings already decorrelate rare items, that signal may be too weak to steer the weights further. A feature-based dissimilarity (e.g. genre Jaccard) or higher λ values could rectify this, but we leave the tuning study to future work.

6.4 Insights on graph construction

Precision vs. Covariance. Sparsified *precision* graphs deliver the lowest SVNN RMSE (0.952) but lose $\sim$0.01 Novelty; dense *covariance* graphs flip the ranking. Hard-thresholding is *often* better than soft (especially for precision GSOs), echoing findings in signal-denoising literature [2].

Using $\mathbf{P} = \tilde{\mathbf{C}}^{-1}$ removes spurious global correlations and accentuates *direct* dependencies, which improves point-wise prediction but dampens the exploratory "long-tail" effect. Hard-thresholding likely sharpens this contrast by zeroing tiny partial correlations.

6.5 Practical implications

For an industrial recommender that already runs a matrix-factorisation baseline, *No-BAM-SVNN* offers an attractive drop-in module: *(i)* training time is $\sim$ 1.6 s per seed on ML-100K, well within real-time retraining budgets; *(ii)* accuracy is statistically indistinguishable from PCA at $\alpha = 0.01$; *(iii)* users receive demonstrably fresher content. Conversely, including BAM terms in the loss inflates runtime by two orders of magnitude with no measurable gain, making it hard to justify in practice unless a richer diversity proxy is found.

6.6 Threats to validity and limitations

Dataset scope. Results are restricted to *MovieLens-100K*. Larger or implicit-feedback corpora may exhibit different sparsity patterns, altering the covariance structure and hence the benefit of SVNNs. Initial ML-1M runs (not reported here) show the same *direction* of effects but with smaller effect sizes.

Offline BAM evaluation. Novelty and Diversity are computed with catalogue-wide popularity and rating-similarity - something that may not match real user perception. An online A/B test or user survey is required for ecological validity.

Small number of random seeds. Five seeds per condition keep runtime tractable but inflate Cohen's d values (Section 5). A replication with $20+$ seeds would stabilise variance estimates.

Fixed architecture. We held the SelectionGNN depth and filter order constant to isolate graph effects. Deeper SVNNs might further close the accuracy gap with PCA.

Hyper-parameter grid. Only two sparsity thresholds and one λ setting were explored; a Bayesian search could uncover sweeter accuracy-diversity operating points.

7 Conclusion & Future Work

This study asked whether *coVariance Neural Networks* and their sparsified variants could improve *beyond-accuracy* qualities in recommendation without sacrificing classical accuracy. Using the *MovieLens-100K* benchmark we compared two SVNN training regimes, a PCA baseline, and naive/random controls. Our findings can be condensed into three take-aways:

1. **SVNNs offer a favourable accuracy-novelty trade-off.** Both SVNN flavours increase Novelty by $\sim$ 3 pp

over PCA while incurring only a 0.03 RMSE penalty (Fig. 2).

2. **Graph construction matters.** Hard-thresholded *precision* GSOs minimise RMSE (0.952), whereas dense *covariance* GSOs maximise Diversity (0.868). Soft-thresholding provides no consistent benefit.
3. **BAM-weighted training is not (yet) cost-effective.** Incorporating Novelty and Diversity terms directly in the loss multiplies runtime by $\times 33$, but yields no discernible performance gain over RMSE-only training.

Together, these results position *No-BAM-SVNN* with a sparsified precision graph as a pragmatic alternative to PCA when discovery metrics are a priority but retraining budgets are tight.

7.1 Future Work

The present investigation is intentionally narrow (single dataset, fixed hyper-grid) so several directions remain open:
Scale and generality. Validate on larger and implicit-feedback datasets (*MovieLens-1M*, Amazon, YouTube) and in streaming scenarios where the covariance structure evolves over time.
Richer beyond-accuracy objectives. Incorporate serendipity, coverage, and even fairness metrics [6], and explore multi-objective optimisation (e.g. Pareto front learning) instead of scalarised losses.
Learnable or adaptive GSOs. Replace the hand-crafted covariance/precision matrix with a learnable graph or with adaptive sparsification thresholds that are optimised jointly with the filter weights.
Deeper SVNN architectures. Extend filter order K or stack additional graph-convolution layers to investigate whether the remaining RMSE gap to PCA can be closed without hurting Novelty.
User-centric evaluation. Conduct A/B tests or lab studies to verify whether offline gains in Novelty and Diversity translate into higher engagement, reduced fatigue, or broader content exploration.
Efficiency engineering. Explore low-rank approximations of the GSO, mixed-precision training and Just-In-Time sparsification to push SVNN inference latency below production thresholds.
Addressing these topics will clarify the practical limits of (S)VNNs and guide their deployment in next-generation recommender engines.

8 Reproducibility Statement

8.1 Open artifacts

- **Source code.** All scripts for data preprocessing, model training, statistical analysis and figure generation are released under an MIT licence at `github.com/IvanValBozhanin/RecSys/tree/ivan`. The last commit `ea6a883` reproduces the results reported here.
- **Data.** We rely exclusively on the *MovieLens-100K* public dataset, which contains no personally-identifying information.
- **Trained checkpoints.** For each of the 95 experimental runs (Table 6 + baselines) we publish the final PyTorch checkpoint, the corresponding YAML configuration, and the raw per-epoch logs.
- **Statistical notebooks.** Two Jupyter notebooks (`stats.ipynb` and `plots.ipynb`) regenerate every table, bar chart, scatter plot and heat-map in the paper. The notebooks read only the released CSV result files; no retraining is required.

8.2 Experimental determinism

- **Random seeds.** All NumPy and PyTorch RNGs are seeded *per run* with the five values $\{10, 20, 42, 2025, 12345\}$. Seeds are stored in the CSV meta-columns and re-set by `train.py` before each trial.
- **Hardware & software.** Experiments ran on an NVIDIA RTX 3060 (12 GB) under Ubuntu 22.04, Python 3.10, PyTorch 2.2 and `torch-sparse` 0.6. Data- loaders use deterministic workers; cuDNN is forced to `deterministic=true`. A `conda environment.yml` is provided for exact dependency recreation.

8.3 Ethical considerations

User privacy. MovieLens ratings are already anonymised; no additional personal data were collected. All intermediate artefacts (covariance and precision matrices) are derived solely from this public corpus.

Bias & filter-bubble risk. While SVNNs increase objective Novelty, our diversity metric is rating-similarity specific and may not align with demographic fairness or content plurality. Future studies (Section 7) will incorporate fairness-aware objectives and user studies to assess possible unintended reinforcement of cultural or genre biases.

Energy usage. Our heaviest configuration (BAM-SVNN) consumed 53 s training time on an RTX-3060, yielding an estimated 0.012 kWh per run, which implies relatively low carbon footprint.

8.4 Licence and data-sharing compliance

All newly created artefacts are released under MIT (code) and CC-BY-4.0 (result CSVs and figures), facilitating reuse and extension. We confirm adherence to the MovieLens data-sharing agreement.

8.5 Limitations of reproducibility efforts

Although code and data are public, exact numerical replication on different GPUs may vary at the third decimal place because of minor floating-point differences in CUDA kernels. Re-running with the same seeds on the original hardware reproduces every mean value in Table 1 within ± 0.0005.

We believe these measures satisfy the ACM Artifact Review and Badging guidelines at the "Available + Reproduced" level.

References

[1] Pablo Castells, Neil J. Hurley, and Saul Vargas. Novelty and Diversity in Recommender Systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 881–918. Springer US, Boston, MA, 2015.

[2] Andrea Cavallo, Zhan Gao, and Elvin Isufi. Sparse Covariance Neural Networks, October 2024. arXiv:2410.01669.

[3] Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 67(4):1034–1049, 2019.

[4] Felipe L. Gewers, Gustavo R. Ferreira, Henrique F. De Arruda, Filipi N. Silva, Cesar H. Comin, Diego R. Amancio, and Luciano Da F. Costa. Principal Component Analysis: A Natural Approach to Data Exploration. *ACM Computing Surveys*, 54(4):1–34, May 2022.

[5] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4), December 2015.

[6] Aryan Jadon and Avinash Patil. A Comprehensive Survey of Evaluation Techniques for Recommendation Systems, January 2024. arXiv:2312.16015.

[7] Christian Montag, Haibo Yang, and Jon D. Elhai. On the Psychology of TikTok Use: A First Glimpse From Empirical Findings. *Frontiers in Public Health*, 9:641673, March 2021.

[8] Li Nian, Ban Xin, Ling Cheng, Gao Chen, Hu Lantao, Jiang Peng, Gai Kun, Li Yong, and Liao Qingmin. Modeling User Fatigue for Sequential Recommendation, May 2024. arXiv:2405.11764 version: 1.

[9] Victor Y Pan, Zhaoqin Chen, Ailong Zheng, et al. The complexity of the algebraic eigenproblem. *Mathematical Sciences Research Institute, Berkeley*, pages 1998–71, 1998.

[10] E. Pariser. *The Filter Bubble: What The Internet Is Hiding From You*. Penguin Books Limited, 2011.

[11] Kartik Sharma, Yeon-Chang Lee, Sivagami Nambi, Aditya Salian, Shlok Shah, Sang-Wook Kim, and Srijan Kumar. A Survey of Graph Neural Networks for Social Recommender Systems, May 2024. arXiv:2212.04481.

[12] Saurabh Sihag, Gonzalo Mateos, Corey McMillan, and Alejandro Ribeiro. coVariance Neural Networks, January 2023. arXiv:2205.15856.

[13] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 109–116, Chicago Illinois USA, October 2011. ACM.

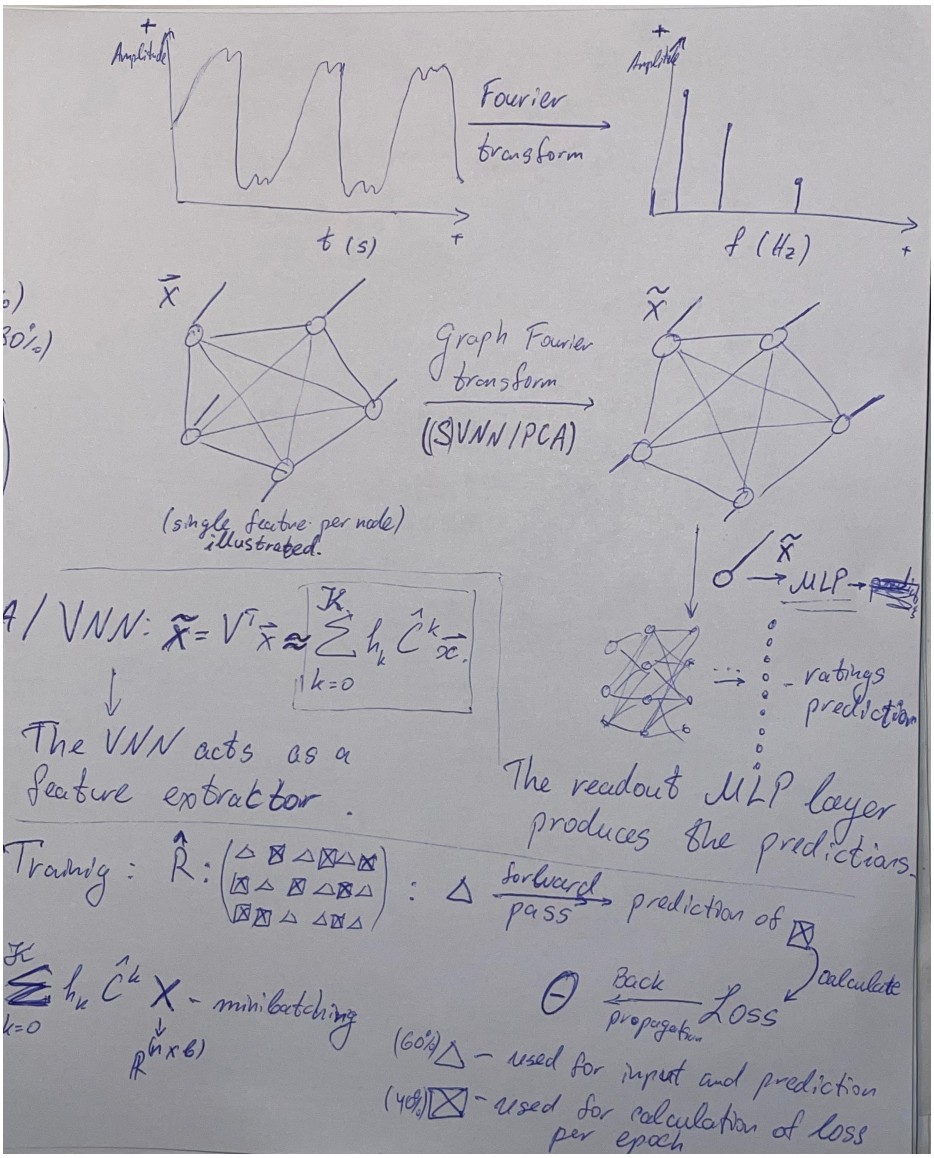

Figure 5: A picture of the model, handrawn, simplified. Purely for visualization purposes.

A Visualization of the Algorithm

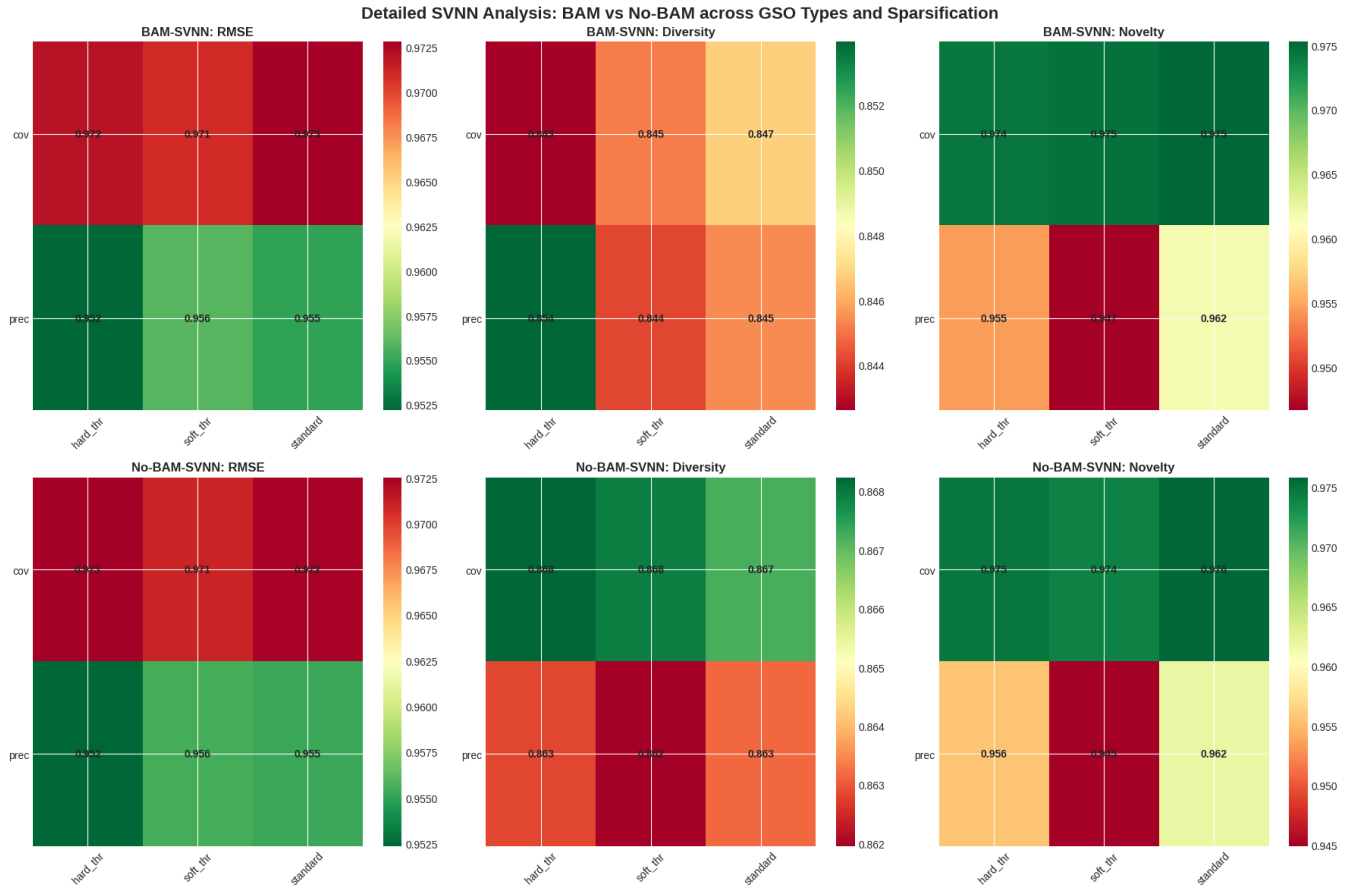

Figure 6: The individual SVNN within-configuration scores on RMSE, Diversity, and Novelty. It is interesting to notice that most of the time, only one row (so either covariance or precision matrix) performs significantly better irregardless of the sparsification.

B Per-metrics SVNN Heatmaps

C Full ANOVA Analysis

RMSE: F = 1214.96, p = 4.929e-96, $\eta^2 = 0.976$
MEAN_DIVERSITY F = 34.66, p = 3.242e-19, $\eta^2 = 0.536$
MEAN_NOVELTY F = 105.25, p = 2.740e-38, $\eta^2 = 0.778$

	group1	group2	meandiff	p-adj	lower	upper	reject	d
0	BAM-SVNN	Naive	0.0626	0.000	0.0507	0.0746	True	-2.879033
1	BAM-SVNN	No-BAM-SVNN	0.0000	1.000	-0.0119	0.0120	False	-0.004098
2	BAM-SVNN	PCA	-0.0322	0.001	-0.0545	-0.0099	True	3.624429
3	BAM-SVNN	Random	0.2533	0.000	0.2413	0.2652	True	-25.011759
4	Naive	No-BAM-SVNN	-0.0626	0.000	-0.0745	-0.0507	True	2.878475
5	Naive	PCA	-0.0949	0.000	-0.1172	-0.0725	True	3.453954
6	Naive	Random	0.1906	0.000	0.1787	0.2026	True	-8.640426
7	No-BAM-SVNN	PCA	-0.0323	0.001	-0.0546	-0.0099	True	3.645029
8	No-BAM-SVNN	Random	0.2532	0.000	0.2413	0.2652	True	-25.056886
9	PCA	Random	0.2855	0.000	0.2632	0.3078	True	-28.202364

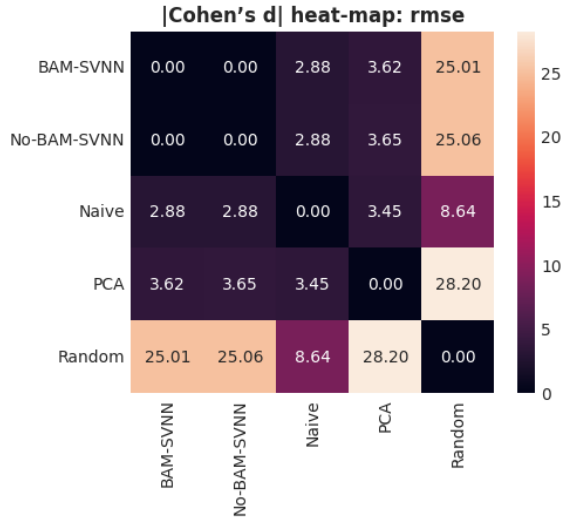

Figure 7: The RMSE analysis of the ANOVA across the configurations.

	group1	group2	meandiff	p-adj	lower	upper	reject	d
0	BAM-SVNN	Naive	0.0164	0.0000	0.0108	0.0219	True	-2.240801
1	BAM-SVNN	No-BAM-SVNN	0.0189	0.0000	0.0133	0.0244	True	-2.560978
2	BAM-SVNN	PCA	0.0234	0.0000	0.0130	0.0338	True	-3.196397
3	BAM-SVNN	Random	0.0201	0.0000	0.0146	0.0257	True	-2.290866
4	Naive	No-BAM-SVNN	0.0025	0.7276	-0.0031	0.0080	False	-0.362419
5	Naive	PCA	0.0070	0.3380	-0.0034	0.0174	False	-1.099300
6	Naive	Random	0.0038	0.3362	-0.0018	0.0093	False	-0.448786
7	No-BAM-SVNN	PCA	0.0045	0.7463	-0.0058	0.0149	False	-0.697726
8	No-BAM-SVNN	Random	0.0013	0.9691	-0.0043	0.0068	False	-0.151007
9	PCA	Random	-0.0033	0.9075	-0.0136	0.0071	False	0.358446

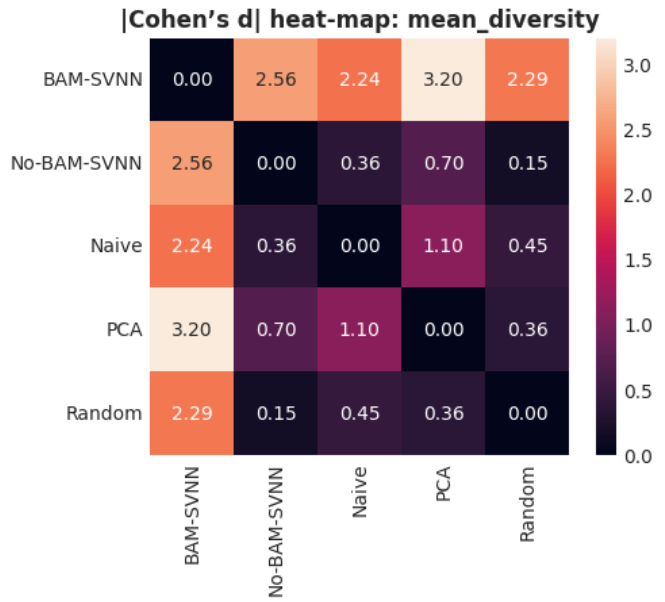

Figure 8: The diversity analysis of the ANOVA across the configurations.

	group1	group2	meandiff	p-adj	lower	upper	reject	d
0	BAM-SVNN	Naive	-0.0927	0.0000	-0.1088	-0.0767	True	5.692933
1	BAM-SVNN	No-BAM-SVNN	-0.0000	1.0000	-0.0161	0.0161	False	0.000786
2	BAM-SVNN	PCA	-0.0256	0.1345	-0.0557	0.0045	False	2.036206
3	BAM-SVNN	Random	-0.0730	0.0000	-0.0890	-0.0569	True	2.617408
4	Naive	No-BAM-SVNN	0.0927	0.0000	0.0766	0.1088	True	-5.663462
5	Naive	PCA	0.0671	0.0000	0.0370	0.0972	True	-3.790297
6	Naive	Random	0.0198	0.0078	0.0037	0.0359	True	-0.672421
7	No-BAM-SVNN	PCA	-0.0256	0.1348	-0.0557	0.0045	False	2.005418
8	No-BAM-SVNN	Random	-0.0730	0.0000	-0.0890	-0.0569	True	2.612481
9	PCA	Random	-0.0474	0.0003	-0.0774	-0.0173	True	1.359593

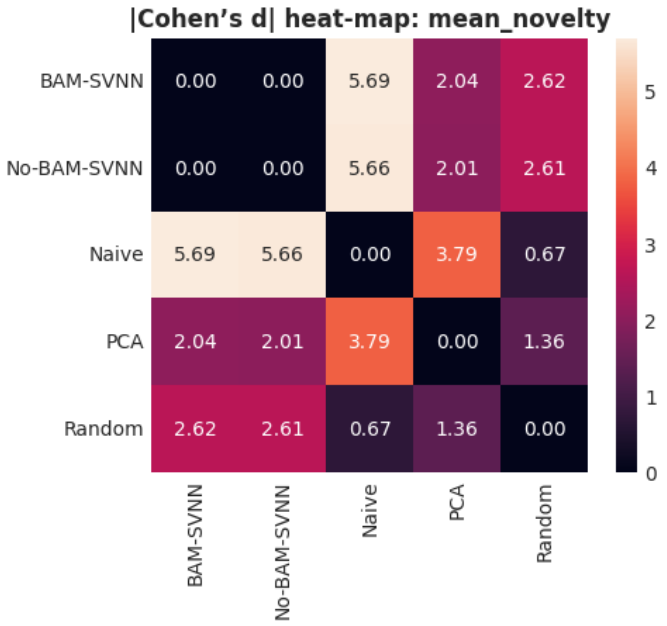

Figure 9: The novelty analysis of the ANOVA across the configurations.