StyleHub - Assessing Contrastive Learning for Perosnalized Fashion Recommendations

Rida Khan
Northeastern University
410 W Georgia St 1400, Vancouver, BC V6B 1Z3, Canada
khan.rida@northeastern.edu

Abstract

Recommendation systems face difficulties in the fashion business due to its dynamic trends and wide range of consumer tastes, especially in cold-start situations. In order to improve individualized fashion recommendations, this study investigates contrastive learning as a self-supervised method. Two approaches are assessed: one that uses product descriptions and embeddings, and the other that is based on transactional data. The latter fared better than the transactional technique, successfully differentiating between similar and dissimilar objects utilizing BERT embeddings and FAISS for similarity search. The findings show that contrastive learning provides a scalable method for adaptive fashion suggestions by collecting subtle visual and semantic cues in order to handle data sparsity. This study emphasizes how it can improve e-commerce customization.

1. Introduction

Artificial intelligence (AI) and computer vision are driving major technical improvements in the quickly changing fashion retail industry, changing how consumers receive fashion advice. The emergence of AI-powered recommendation systems has significantly changed the fashion sector, which is renowned for its ever-changing trends and wide range of consumer tastes. These solutions, which include automated outfit creation and individualized style recommendations, are becoming vital resources for improving client experiences and increasing operational effectiveness. AI models that integrate transactional data and visual content are proven to be effective innovation accelerators as fashion shops depend more and more on these sources to comprehend customer behavior.

Despite their widespread use, traditional recommendation models—which are based on user interactions and transactional history—face difficulties such data sparsity, model degradation, and an inability to keep up with the rapidly changing fashion trends. However, the increasing accessibility of fashion pictures offers a special chance to get over these restrictions. By using visual data for item-item recommendations, this research investigates how contrastive learning, a formidable machine learning technique, might be used to overcome these issues. In particular, we look into how contrastive learning may be used to make more scalable and flexible personalized suggestions for fashion goods based on image similarity.

1.1. Motivation for Study

The fashion business has special challenges for recommendation systems because of its rapidly changing trends, wide range of styles, and visually rich content. Conventional techniques, such content-based approaches and collaborative filtering, frequently find it difficult to keep up with these constant changes. For instance, content-based filtering tends to rely too much on static attributes, which results in recommendations that are less varied and adaptive, whereas collaborative filtering suffers from data sparsity, particularly for new users (cold-start problem). Due to the extremely visual nature of the fashion industry, these models are unable to effectively utilize the abundance of visual information found in fashion photos, which is frequently essential for determining a user's preferences.

There is a rising chance to reconsider the way recommendations are created because e-commerce platforms are becoming more and more accessible with high-quality photographs. In fields where picture data is essential, contrastive learning—a machine learning technique that compares pairs of similar and dissimilar objects to learn representations—has shown a lot of promise. Its use in fashion suggestion algorithms is still not well understood, though. The need to comprehend how contrastive learning can be used in the context of fashion recommendations and what particular issues it can more successfully handle than conventional techniques is what propelled this study.

1.2. Traditional Approaches to Fashion Recommender Systems

In the fashion industry, recommender systems have historically used either content-based filtering (CBF), collaborative filtering (CF), or hybrid approaches that combine the two. These methods have been the cornerstones of suggestion generation on e-commerce platforms.

1.2.1 Collaborative Filtering (CF)

To forecast customer preferences, CF uses information about user-item interactions, such as past purchases or product ratings. It works on the premise that consumers who have exhibited similar habits in the past would also likely exhibit similar preferences going forward. To find latent components that describe user and item preferences, popular methods such as Matrix Factorization (MF) have been widely applied.

Challenges

- **Data Sparsity:** Fashion datasets are inherently sparse, with users interacting with only a small fraction of the catalog. This limits CF's ability to generalize across users and items.
- **Cold Start Problem:** New users or items without sufficient interaction history cannot be effectively integrated into CF models.

1.2.2 Content-Based Filtering (CBF)

To suggest products that are similar to those a user has engaged with, CBF uses item qualities (such as color, material, and category). When transactional data is scarce, it performs exceptionally well in producing recommendations for specialized or infrequent consumers.

Challenges

- **Limited Item Representation:** Traditional item attributes often fail to capture rich visual or stylistic nuances critical in fashion.
- **Narrow Recommendations:** CBF tends to overfit to user preferences, offering recommendations too similar to past choices, which can reduce diversity.

1.2.3 Hybrid Methods

The goal of combining CF and CBF is to minimize each technique's drawbacks while maximizing their respective advantages. For example, sparse CF data can be supplemented with visual cues taken from product photos.

Challenges

- **Integration Complexity:** Merging CF and CBF effectively requires robust models and computational resources.
- **Personalization Gaps:** Hybrid methods may still struggle with evolving fashion trends and subjective user preferences.

[2]

1.3. Proposed Technique: Contrastive Learning for Fashion Recommendation

This study examines contrastive learning's potential as a workable strategy for tailored fashion recommendations with the goal of assessing how well it performs in comparison to more conventional collaborative filtering techniques. A self-supervised learning method called contrastive learning uses unlabeled data to create meaningful representations. The method improves the representation of similar pairs while reducing the similarity of dissimilar ones by teaching the model to discriminate between similar and dissimilar items.

Investigating whether contrastive learning may be used successfully in the context of tailored fashion advice is the aim of this study. In order to do this, the study suggests using two different methods to solve the tailored suggestion problem:

1. **Pair Generation Based on Past Transactional Data:** Leveraging historical transactional data to create positive and negative pairs for training the contrastive learning model, enabling the system to learn user preferences directly from purchase behavior.
2. **Item Embeddings and User Preferences:** Solely relying on item embeddings and inferred user preferences to determine similarities between items, aiming to generate recommendations without requiring extensive transactional pair data.

The purpose of this study is to compare these two methods in order to ascertain how flexible and effective contrastive learning is in capturing a range of consumer preferences and enhancing suggestion accuracy. The research's empirical findings will clarify if incorporating contrastive learning into contemporary recommendation systems is feasible, providing a way forward for more individualized and contextually aware fashion industry solutions.

1.4. Contributions of this Study

This study makes several key contributions to the field of fashion recommendation systems, focusing on evaluating the potential of contrastive learning for personalized fashion recommendations:

- **Application of Contrastive Learning for Fashion Recommendations:** This paper applies contrastive learning

to the personalized fashion recommendation problem, exploring its effectiveness in leveraging visual and transactional data for meaningful item representations.

- **Dual Approach for Personalized Recommendations:** The study investigates two complementary strategies: (1) pair generation using past transactional data to infer user preferences, and (2) item embedding-based recommendations relying on user preferences and item similarities, offering insights into the strengths of each approach.
- **Enhanced Representation Learning:** By utilizing contrastive learning, the model captures subtle and rich visual characteristics of fashion items, which contributes to improved recommendation quality by focusing on the intrinsic properties of the items.
- **Addressing Data Sparsity:** This work explores how contrastive learning can overcome the challenges of sparse user interaction data by leveraging visual features, enabling meaningful recommendations even in data-scarce scenarios.

Following these contributions, the paper is structured as follows: Section 1 introduces the problem and outlines the study's objectives. Section 2 reviews relevant literature on recommendation systems and contrastive learning. Section 3 describes the methodology, detailing the contrastive learning framework and data preparation. Section 4 presents the experimental results, evaluating the performance of the proposed approaches. Section 5 provides a discussion of findings, implications, and limitations. Section 6 concludes with potential future directions for research.

2. Related Work

Innovations in recommendation systems have mostly focused on collaborative filtering and deep learning methods. While collaborative filtering effectively utilizes user-item interactions, it encounters difficulties with new and rare items, known as the cold-start problem. Self-supervised approaches such as contrastive learning have recently emerged, exhibiting potential in fashion recommendation by capturing relevant item representations in the absence of large historical data. This section explores these techniques, emphasizing their applications and constraints in the context of fashion recommendations.

2.1. Fashion Recommendation System Using Collaborative Filtering

Syiam et al. (2023) developed a collaborative filtering-based fashion recommendation system that uses adjusted cosine similarity to account for individual user biases, improving the reliability of item suggestions without requiring large amounts of labeled data [6]. This approach, applied to data from the fashion rental platform "Rent the Runway," allows the model to capture nuanced item relationships, making it suitable for the fast-paced fashion industry.

Key contributions include the use of adjusted cosine similarity over standard cosine similarity, which better addresses individual user biases by calculating similarity based on deviations from users' average ratings rather than absolute ratings. The adjusted similarity between items i_p and i_q is calculated as follows:

$$\text{sim}(i_p, i_q) = \frac{\sum_{k=1}^{m}(R_{k,p} - \bar{R}_k) \cdot (R_{k,q} - \bar{R}_k)}{\sqrt{\sum_{k=1}^{m}(R_{k,p} - \bar{R}_k)^2} \cdot \sqrt{\sum_{k=1}^{m}(R_{k,q} - \bar{R}_k)^2}}$$

where $R_{k,p}$ and $R_{k,q}$ represent user k's ratings for items p and q, and $\bar{R}_k$ is user k's average rating. This method allows the model to detect subtle item similarities by reducing the impact of individual rating biases.

To predict ratings for items a user hasn't rated, the system uses a weighted sum of similar item ratings:

$$P_{u,i} = \frac{\sum_{\text{all similar items, } N}(S_{i,N} \cdot R_{u,N})}{\sum_{\text{all similar items, } N} |S_{i,N}|}$$

where $P_{u,i}$ is the predicted rating by user u for item i, $S_{i,N}$ is the similarity between items i and N, and $R_{u,N}$ is the rating user u gave to item N.

The system demonstrated improved recommendation accuracy, achieving a Mean Absolute Error (MAE) of 0.4424 and a Normalized Discounted Cumulative Gain (NDCG) of 0.9988, surpassing conventional cosine similarity approaches. Larger k-fold cross-validations and higher Top-n values further enhanced its performance, showing that adjusted cosine similarity provides balanced, high-quality recommendations tailored for subjective fashion choices. This approach's adaptability for subjective visual and stylistic preferences makes it effective for recommending fashion items even when user interaction data is limited.

2.2. A Simple Framework for Contrastive Learning of Visual Representations

Chen et al. (2020) developed SimCLR, a contrastive learning framework that simplifies self-supervised learning by removing complex mechanisms like memory banks and specialized structures [1]. Unlike prior approaches that relied on architectural alterations (e.g., MoCo, CPCv2), SimCLR optimizes agreement between different augmented perspectives of the same image, achieving state-of-the-art performance without labeled data. This adaptability is relevant to diverse applications, including our personalized fashion recommendation system.

Key contributions of SimCLR include:

- **Simplified Contrastive Learning:** SimCLR removes memory banks by using high batch sizes, inherently including many negative samples. This simplification improves scalability and efficiency.

- **Data Augmentation:** Random cropping, color jittering, Gaussian blur, and horizontal flipping create varied but related views of an image, crucial for contrastive learning. Stronger augmentations improve feature learning by pushing the model towards more invariant features.
- **Nonlinear Projection Head:** A nonlinear MLP projection head between the encoder and contrastive loss improves learned representations by mapping encoder outputs to a contrastive learning-specific latent space.
- **Large Batch Sizes and Training Duration:** SimCLR uses large batch sizes (up to 8192 images) and longer training times, ensuring diverse negative samples in each batch and refining representations by more thoroughly exploring the data manifold.

SimCLR's methodology is simple yet effective. Each image undergoes a series of augmentations to create two distinct views serving as positive pairs. A ResNet encoder extracts high-dimensional features from these views. An MLP projection head then maps the encoder's output to a lower-dimensional latent space. The contrastive loss maximizes agreement between positive pairs and minimizes it between other pairs in the batch, calculated using cosine similarity and a temperature-scaled softmax function:

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)}$$

where:

- $\mathbf{z}_i$ and $\mathbf{z}_j$ are embeddings of the positive pair,
- $\text{sim}(\mathbf{z}_i, \mathbf{z}_j)$ is the cosine similarity, $\frac{\mathbf{z}_i^\top \mathbf{z}_j}{\|\mathbf{z}_i\| \|\mathbf{z}_j\|}$,
- τ is a temperature parameter controlling distribution sharpness.

SimCLR uses the LARS optimizer, which stabilizes training with large batches by updating each layer proportionally. This method allows SimCLR to scale successfully to large datasets like ImageNet.

SimCLR achieved impressive results, outperforming prior approaches with a top-1 accuracy of 76.5% on ImageNet and a top-5 accuracy of 85.8% with only 1% labeled data. It generalizes well across CIFAR-10, CIFAR-100, and STL-10, demonstrating robustness and scalability.

In a fashion recommendation context, SimCLR's data augmentations can simulate multiple views of the same item, such as different angles and lighting conditions, allowing our model to learn richer, more discriminative features. The nonlinear projection head can improve the learned representations of fashion items, and the contrastive loss enables clustering of visually similar items in latent space, which is ideal for our recommendation objectives.

2.3. Contrastive Learning for Recommender System

Liu et al. (2021) propose a paradigm for recommender systems that leverages contrastive learning to address biases in user-item interaction data [3]. Their framework enhances the quality of user and item representations using Graph Neural Networks (GNN) alongside a debiased contrastive learning module.

Key contributions of this work include a *Graph Contrastive Learning Module*, a self-supervised technique that improves GNN-based recommendation models by reducing selection bias through graph perturbations. Another major contribution is their *Debiased Contrastive Loss*, which rectifies suboptimal sampling and sampling biases in collaborative filtering models, thereby improving recommendation accuracy. Lastly, they offer a *Comprehensive Evaluation* across multiple benchmarks, such as the Yelp2018, Amazon-Book, and Steam datasets, demonstrating the effectiveness of their approach.

The method begins by constructing a bipartite user-item graph, where user interactions are encoded using a GNN. This model learns embeddings by propagating information across the graph structure. To support contrastive learning, Liu et al. employ **graph perturbations** by randomly removing a subset of edges to generate two independent views for each node. This serves as the foundation for contrastive learning.

The graph contrastive loss (GCL) for user-item pairs in a batch B is computed as follows:

$$L_{GCL} = \frac{1}{B} \sum_{u \in B} l(u)$$

where $l(u)$ represents the normalized cross-entropy loss across two views for each user node. Additionally, a debiased contrastive loss (DCL) is introduced to address poor sampling, defined as:

$$L_{DCL} = -\frac{1}{|D|} \sum_{(u,i) \in D} \log \frac{e^{\phi(u,i)}}{e^{\phi(u,i)} + \sum_{l=1}^{L} e^{\phi(u,i_l)}}$$

where $\phi(u, i)$ represents the cosine similarity between user and item embeddings, and L denotes the number of negative samples. This loss penalizes ineffective sampling, ensuring that the model learns from valuable contrasts.

Results indicate that the framework significantly enhances *recall@K* and *NDCG@K* metrics across datasets like Yelp2018, Amazon-Book, and Steam, outperforming baseline models such as MF and LightGCN. These improvements highlight the model's ability to capture complex user-item relationships, which is crucial for effective recommendations.

The GNN-based architecture is particularly applicable to our research on personalized fashion recommendations, as

it aligns with our objectives of leveraging user-item interaction data, especially for users with limited purchase history. The debiased contrastive learning component can potentially enhance model performance by accurately capturing user preferences and item relationships, both vital for creating tailored recommendations.

2.4. Contrastive Learning for Interactive Recommendation in Fashion

Sevegnani et al. (2022) introduced *WhisperLite*, a pioneering recommendation system for online fashion retail that combines search and recommendation functionality through contrastive learning [5]. WhisperLite is designed to address real-time challenges in online fashion, particularly for handling complex, evolving user queries. This system is highly effective in overcoming cold-start issues typical of recommendation systems, where limited or no historical data exists for new users or products. By leveraging contrastive learning, WhisperLite significantly enhances user interaction quality, matching challenging user queries with relevant fashion products. This dual-functionality as a search and recommendation system makes it versatile and highly applicable to e-commerce.

WhisperLite's architecture seamlessly integrates with user inputs to deliver personalized recommendations based on both textual and product data. This capability is particularly important in fashion retail, where consumer preferences are diverse and dynamic. The system focuses not only on recommendation accuracy but also on user engagement, enhancing the shopping experience.

Key contributions of WhisperLite include the *integration of search and recommendation systems* to allow users to conduct dynamic fashion searches and receive real-time recommendations, adaptable to changing preferences, such as "formal black dresses for evening events." This hybrid model provides a more interactive experience compared to static recommendation systems, which often rely solely on past behavior.

Another core advancement is the use of *CLIP Embeddings*, a model trained on a large dataset of image-text pairs. These embeddings enable WhisperLite to accurately map user text queries to product images, capturing user intent for improved recommendations. CLIP's dual representation facilitates precise matching of user descriptions with product images, enhancing the system's ability to provide contextually relevant recommendations, such as accurately pairing a query for a "casual blue summer dress" with suitable products.

WhisperLite also addresses the *cold-start problem* using contrastive learning, which enables generalization in cases where minimal data exists. By training the model to correlate text and images via CLIP embeddings, WhisperLite is capable of making accurate predictions even for new users or items, which is particularly valuable in fashion e-commerce where trends change quickly.

Lastly, WhisperLite demonstrates improved retrieval metrics, such as precision@k, recall@k, and NDCG, which are key for assessing recommendation relevance. WhisperLite outperforms traditional recommendation systems, especially for complex queries that combine multiple product attributes, such as "red heels with a comfortable fit for office wear."

The methodology underlying WhisperLite combines text and image embeddings to create more accurate and customized recommendations. WhisperLite uses CLIP embeddings to project both user queries and product images into a shared latent space. This embedding model allows the system to match complex natural language descriptions, like "elegant silk blouse," to relevant images, improving recommendation quality.

WhisperLite uses *cosine similarity* to determine the relevance of product matches to user queries by calculating the angular distance between embedding vectors, ranking items by relevance:

$$\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$$

where $\mathbf{u}$ and $\mathbf{v}$ represent text and image embedding vectors. Higher similarity scores indicate stronger matches, enabling the system to prioritize relevant items.

The model employs *contrastive loss* to refine alignment between text queries and product images, aiming to reduce the distance between related embeddings while increasing the distance between unrelated ones. This ensures that relevant items are recommended and irrelevant ones excluded:

$$\mathcal{L} = -\log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)}$$

where $\mathbf{z}_i$ and $\mathbf{z}_j$ are normalized embeddings of paired text and image inputs, and τ is a temperature parameter. This dual-loss strategy improves matching by allowing the model to learn from both positive and negative samples.

WhisperLite underwent real-user testing via Amazon Mechanical Turk, which confirmed the model's practical effectiveness and showed improvements in both technical performance and user satisfaction in real-world e-commerce scenarios. User feedback helps ensure that WhisperLite continually adapts to meet consumer preferences.

Evaluated on real-world datasets (e.g., retail and open-source datasets from sectors like restaurants and movies), WhisperLite consistently outperformed baseline recommendation models in terms of precision, recall, and NDCG, especially in cold-start scenarios where user data is limited. WhisperLite's use of contrastive learning and CLIP embeddings facilitates accurate query-product alignment, achieving better results than traditional systems.

The techniques implemented in WhisperLite offer valuable insights for developing our own contrastive learning-based recommendation system. WhisperLite's methodology, while focusing on text-to-image matching, can be adapted to our project's needs, specifically for image-to-image matching. Its approach to handling cold-start scenarios is particularly relevant for our fashion recommendation environment, as users may lack substantial browsing histories. By using contrastive learning, our model will be able to generalize well with minimal data, providing accurate recommendations for both new users and products. This feature is essential for creating a robust and user-friendly recommendation system in fashion.

2.5. Semi-Supervised Visual Representation Learning for Fashion Compatibility

Revanur et al. (2021) present a semi-supervised approach to learning visual representations for fashion compatibility, which minimizes reliance on large labeled datasets by leveraging both labeled and unlabeled data [4]. This approach is particularly advantageous for fashion recommendations, where labeling vast amounts of data is often impractical or costly. The method utilizes pseudo-labeling and consistency regularization, allowing the model to learn effectively even with limited labeled samples, making it scalable for real-world fashion e-commerce applications.

The primary contributions include a novel *Pseudo-Labeled Data Augmentation* technique, which dynamically creates pseudo-positive and pseudo-negative triplets during each training cycle, thereby optimizing unlabeled data usage. Additionally, *Self-Supervised Consistency Regularization* is applied to enforce consistent attribute learning by requiring similarity between augmented and original views of each image. Finally, the method's performance is extensively evaluated on datasets such as Polyvore, Polyvore-Disjoint, and a proprietary Fashion Outfits dataset, achieving results comparable to fully-supervised models.

The methodology leverages a ResNet-18 backbone trained with a triplet margin loss:

$$L = \max(0, d(\phi_A, \phi_P) - d(\phi_A, \phi_N) + m)$$

where ϕ_A, ϕ_P, ϕ_N are the anchor, positive, and negative embeddings, respectively, and m is the margin. Pseudo-labeled triplets are generated, where an anchor-positive pair is accompanied by an unlabeled but visually similar negative sample. Consistency regularization, applied to shape and color transformations, further strengthens attribute learning. The overall loss function integrates labeled, pseudo-labeled, and self-supervised losses:

$$L = L_l + \lambda_{ss} L_{ss} + \lambda_{pseudo} L_{pseudo}$$

where L_{ss} is the self-supervised consistency loss, and $\lambda_{ss}, \lambda_{pseudo}$ are hyperparameters that adjust the contribution of each loss component.

The proposed model performs comparably to fully-supervised models in tasks like compatibility prediction and fill-in-the-blank, despite using only a fraction of the labeled data. Consistency regularization aids in learning meaningful visual features, enhancing generalization across varied fashion combinations and scenarios.

This semi-supervised technique aligns with our project's goal of minimizing labeled data dependency in fashion recommendations. The method's focus on visual compatibility and attribute learning, such as color, shape, and pattern, directly supports our objective of delivering personalized fashion recommendations. The self-supervised approach for data-scarce scenarios offers a promising enhancement for style and compatibility assessments in our model.

3. Methodology

This study's technique is focused on assessing how contrastive learning can be used to provide tailored fashion recommendations. By using contrastive learning to increase recommendation accuracy and adaptability, the suggested framework seeks to close the gap between conventional collaborative filtering methods and contemporary machine learning techniques. The dataset and two different approaches used—one based on historical transactional data and the other using picture embeddings—are explained in detail in this section.

3.1. Dataset Selection

This research uses the **H&M Personalized Fashion Recommendations** dataset, which includes consumer purchase histories, article metadata, and additional customer information. This dataset is ideal for researching the suggestion of rare and new products since it includes users' time-stamped purchase behavior over an extended period, providing insights into both popular and less common purchases. The dataset consists of the following essential parts:

- **Images:** A directory containing photos for each article, organized by the first three digits of each `article_id`. Note that not all `article_id` values have associated photos, resulting in variability in visual data availability.
- **articles.csv:** A metadata file for each article, including attributes such as product type, color, and seasonality. This information is valuable for understanding product features and supporting item similarity comparisons.
- **customers.csv:** A file containing metadata for each customer, including demographic and behavioral data, which can be used for personalization.
- **transactions_train.csv:** The main training dataset, recording each customer purchase. Duplicate values represent multiple purchases of the same item on a given day.

This dataset, which contains detailed records of transactions throughout time, allows us to investigate both typical and uncommon purchase behaviors. By analyzing item purchase frequency, we can identify and assess recommendations for goods with few interactions (rare items) and those freshly added to the catalog (new things), providing an effective testbed for our model's effectiveness in real-world scenarios.

3.2. Contrastive Learning Method

To train the recommendation model for this study, we use a contrastive learning methodology based on SimCLR. A self-supervised learning approach called SimCLR (Simple Contrastive Learning of Representations) compares positive pairings (similar items) with negative pairs (dissimilar items) to learn feature representations. Since it has been demonstrated to be successful in contrastive learning tasks, we employ the normalized temperature-scaled cross-entropy (NT-Xent) loss function.

The overall approach consists of the following key components:

1. **Feature Extractor**
   The feature extractor is a deep neural network designed to transform the input data into meaningful representations. In our case, the feature extractor takes in the preprocessed dataframe, which already contains extracted embeddings. It retains only the necessary features to feed into the model for further processing. This feature vector encapsulates the essential information about the input item, enabling the model to learn discriminative representations in a high-dimensional space.
2. **Contrastive Dataset Construction**
   Given the pairwise nature of contrastive learning, a dataset of positive and negative item pairs is constructed. Positive pairs consist of items that are deemed similar (for instance, items purchased together by a user, or this could be done based on textual embeddings of items that are similar), while negative pairs are randomly selected items from the dataset that are assumed to be dissimilar. These pairs are used as inputs to the contrastive learning model.
3. **Model Architecture**
   The model takes in the feature vector from the feature extractor and processes it through two hidden layers, which reduce its dimensionality. The architecture is designed as follows:
   - **Input Layer:** The model receives a feature vector of dimension d_{input}.
   - **Hidden Layers:** The feature vector is passed through two fully connected hidden layers, each with ReLU activations, to reduce the dimensionality of the input. The output from the second hidden layer has a reduced dimension d_{hidden}.

     $$h_1 = \text{ReLU}(W_1 x + b_1) \quad \text{(first hidden layer)}$$

     $$h_2 = \text{ReLU}(W_2 h_1 + b_2) \quad \text{(second hidden layer)}$$

   - **Projection Head:** After dimensionality reduction, the resulting representation is passed through a projection head. This projection head further reduces the dimensionality to a lower space d_{proj}, making the final output suitable for contrastive learning. The projection head is typically a small multi-layer perceptron (MLP) with a final output layer that produces the representation in this lower-dimensional space.

     $$z = W_3 h_2 + b_3 \quad \text{(projection head output)}$$

     Where W_3 and b_3 are the weight and bias parameters for the projection head.
4. **NT-Xent Loss Function**
   The key component of contrastive learning is the loss function, which measures how well the model distinguishes between positive and negative pairs. We use the normalized temperature-scaled cross-entropy (NT-Xent) loss, which encourages the model to pull positive pairs closer together in the feature space while pushing negative pairs apart. The NT-Xent loss function is given by:

   $$L(i,j) = -\log\left(\frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{N}\exp(\text{sim}(z_i, z_k)/\tau)}\right)$$

   Here:
   - z_i and z_j are the representations of the positive pair (item i and item j).
   - $\text{sim}(z_i, z_j)$ is the cosine similarity between the representations z_i and z_j:

     $$\text{sim}(z_i, z_j) = \frac{z_i^\top z_j}{\|z_i\|\|z_j\|}$$

   - τ is the temperature parameter, which controls the smoothness of the similarity distribution.
   - The sum in the denominator runs over all items in the batch, including both positive and negative pairs.

   The model minimizes this loss during training to learn robust representations that are discriminative for similar and dissimilar items.

3.2.1 Approach Using Past Transactional Data

This method creates training pairs for the contrastive learning model by utilizing past transactional data. Items commonly co-purchased by the same user are used to create positive pairings, which represent innate relationships derived from user behavior. To provide diversity in the training data, negative pairs, on the other hand, are randomly

selected from goods that were not bought together. The goal of training the contrastive learning model is to maximize the distance between negative pair embeddings and reduce the distance between positive pair embeddings. This approach enhances the system's capacity to provide tailored recommendations by enabling it to directly understand significant correlations between items from user behavior. The efficiency of the learnt embeddings in forecasting user preferences and producing precise suggestions is assessed after training.

3.2.2 Approach Using Image Embeddings

Using the dataset's textual description columns, this method concentrates on textual similarity. In order to capture the semantic substance of the descriptions, stopwords are eliminated from these columns and BERT embeddings are created. Cosine similarity is then computed using these embeddings, which serve as the foundation for producing positive and negative item pairs. objects with comparable textual descriptions make up positive pairs, whilst objects with divergent descriptions make up negative pairs. Even without a lot of transactional data, the model may employ feature-extracted visual embeddings to suggest similar things when a user provides an image of a desired item. This ensures that the suggestions are based on the written descriptions of the goods.

During evaluation, we track the cosine similarity of both positive and negative pairs. For each pair, the cosine similarity is calculated as:

$$\text{cosine\_similarity}(z_i, z_j) = \frac{z_i^T z_j}{\|z_i\| \|z_j\|}$$

where z_i and z_j represent the embeddings of items i and j, respectively.

The model's performance is assessed by comparing the average cosine similarity for positive pairs ($\text{sim}_{\text{positive}}$) and negative pairs ($\text{sim}_{\text{negative}}$). The goal is to ensure that the average similarity of positive pairs is high, while the average similarity of negative pairs is low. The evaluation metric is considered successful if the model increases the distance between positive and negative pairs, indicated by a larger difference between the average positive similarity and the average negative similarity.

Thus, model performance is gauged by how well the embeddings are separated in the feature space, with higher positive similarities and lower negative similarities indicating better performance in distinguishing similar and dissimilar items.

4. Experiments and Results

In this section, we describe the model architecture and training process, followed by two distinct approaches used in our experiments. We will evaluate the effectiveness of each approach based on the metrics described earlier, particularly the cosine similarity between positive and negative pairs.

4.1. Model Architecture and Training

This section provides an overview of the contrastive learning model, its components, and the associated loss functions used to tackle both approaches.

4.1.1 Model Overview

The proposed model utilizes a contrastive learning framework to learn meaningful representations of fashion items, enabling personalized recommendations based on textual and visual features. The architecture comprises several key components: a feature extractor, a projection head, and the contrastive loss function. These components work together to learn a shared embedding space where similar items (based on either textual descriptions or visual appearance) are placed close together, while dissimilar items are pushed farther apart.

The model is trained using a contrastive loss that encourages the network to learn embeddings that are discriminative for the task of item similarity. The system is capable of handling both textual and image inputs, with a particular emphasis on textual data in the form of item descriptions.

4.1.2 Model Components

Feature Extractor The Feature Extractor is a neural network module responsible for transforming raw features (either text-based or image-based) into a high-dimensional feature vector. In our implementation, the FeatureExtractor class is an MLP (Multi-Layer Perceptron) that consists of several fully connected layers, ReLU activations, batch normalization, and dropout for regularization.

The input features, typically extracted from item descriptions (text) or images, are passed through this network to produce an embedding vector that captures relevant information about the item. The final output dimension of the feature extractor is 512, which is configurable.

Mathematically, the output of the feature extractor $\mathbf{z}_i$ for an input $\mathbf{x}_i$ is computed as:

$$\mathbf{z}_i = f_{\text{feat}}(\mathbf{x}_i)$$

where $f_{\text{feat}}(\cdot)$ denotes the mapping learned by the Feature Extractor, and $\mathbf{z}_i$ is the output embedding for item i.

Projection Head The Projection Head is another MLP that maps the output of the feature extractor into a lower-dimensional space, suitable for contrastive loss computation. The projection head projects the embeddings from the

feature extractor into a space where contrastive learning can be applied effectively.

The output projection $\mathbf{p}_i$ for item i is computed as:

$$\mathbf{p}_i = f_{\text{proj}}(\mathbf{z}_i)$$

where $f_{\text{proj}}(\cdot)$ is the projection head, and $\mathbf{p}_i$ is the projected representation of item i.

SimCLR Model The *SimCLR* class combines the feature extractor and the projection head into a single model. Given an input $\mathbf{x}_i$ (either a textual description or an image), the model first extracts features using the feature extractor, then applies the projection head to obtain the final projection. The model is trained to minimize the contrastive loss between pairs of items.

$$\mathbf{p}_i = f_{\text{proj}}(f_{\text{feat}}(\mathbf{x}_i))$$

Contrastive Loss: NT-Xent Loss The model uses the *NT-Xent Loss* (Normalized Temperature-scaled Cross Entropy Loss) to encourage the network to learn similar embeddings for similar items and dissimilar embeddings for different items. The NT-Xent loss is computed between pairs of embeddings $\mathbf{p}_i$ and $\mathbf{p}_j$, where the goal is to pull similar items together (positive pairs) and push dissimilar items apart (negative pairs).

The loss function is defined as:

$$L_{\text{NT-Xent}}(\mathbf{p}_i, \mathbf{p}_j) = -\log\left(\frac{\exp(\text{sim}(\mathbf{p}_i, \mathbf{p}_j)/\tau)}{\sum_{k=1}^{N}\exp(\text{sim}(\mathbf{p}_i, \mathbf{p}_k)/\tau)}\right)$$

where:

- $\text{sim}(\mathbf{p}_i, \mathbf{p}_j)$ is the cosine similarity between the projections of items i and j,
- τ is a temperature scaling parameter that controls the softness of the distribution,
- N is the total number of items in the batch, and the denominator is a sum over the similarity scores of the anchor item i and all other items in the batch.
- The positive pair (items i and j) is associated with the correct label, while the negative pairs are samples from the rest of the batch.

The cosine similarity between two vectors $\mathbf{p}_i$ and $\mathbf{p}_j$ is computed as:

$$\text{sim}(\mathbf{p}_i, \mathbf{p}_j) = \frac{\mathbf{p}_i \cdot \mathbf{p}_j}{\|\mathbf{p}_i\|\|\mathbf{p}_j\|}$$

This loss function encourages the model to project similar items (positive pairs) closer together in the embedding space while pushing dissimilar items (negative pairs) further apart.

Contrastive Dataset and Training Pairs A ContrastiveDataset is used to manage the pairs of items for contrastive training. The dataset is constructed by forming positive and negative pairs based on item similarity. Positive pairs consist of items that are similar (based on textual description or visual similarity), and negative pairs are randomly sampled from items that are dissimilar.

- Positive Pairs: Items that are similar to each other based on precomputed features (e.g., embeddings or nearest neighbors),
- Negative Pairs: Items that are dissimilar to each other, sampled randomly from the dataset.

The training dataset consists of pairs (x_1, x_2) where x_1 and x_2 are the inputs (either textual descriptions or images) for the positive or negative pairs. These pairs are fed into the model during training to compute the contrastive loss.

4.2. Approach 1: Using Prior Transaction History for Pair Generation

In this approach, we leverage the transaction history data of individual customers to generate positive and negative pairs for training. The goal is to create pairs of items based on the assumption that items purchased together or within the same transaction have a higher likelihood of being similar in the customer's preferences.

4.2.1 Pair Generation Process

We evaluate each customer's prior transaction history and create positive pairs from items they have previously purchased. For instance, if a customer has bought items A and B in the same transaction or within close timeframes, we consider them as a positive pair, assuming a level of similarity or relevance in the customer's preferences.

On the other hand, negative pairs are generated from items that the customer has never purchased or interacted with, based on their transaction history. These items are assumed to be dissimilar, providing a contrast for the model to learn how to distinguish between relevant and irrelevant items.

4.2.2 Model Training with Transactional Data

These positive and negative pairs are fed into the model, as described in the previous section. The model, using the contrastive loss function, aims to minimize the distance between positive pairs (similar items) and maximize the distance between negative pairs (dissimilar items). The training process follows the standard procedure of feature extraction, projection, and loss calculation, as outlined earlier.

4.2.3 Challenges and Observations

However, we observed a significant issue after training for 50 epochs: the cosine distance between both positive and negative pairs was approaching 1, indicating that the model was unable to effectively differentiate between the pairs. This resulted in the model's performance decaying, as it was unable to distinguish between the positive and negative pairs effectively.

We hypothesize that the transactional data alone, while helpful for pairing items based on purchase behavior, may not capture the full spectrum of item similarity necessary for a robust recommendation model. The transactional data did not provide enough variation in the types of negative pairs, leading to a lack of sufficient contrast for the model to learn meaningful embeddings.

4.2.4 Conclusion and Transition to Approach 2

Given these challenges, we concluded that using transactional data in its raw form was not the most effective way to generate meaningful positive and negative pairs for contrastive learning. As a result, we transitioned to an alternative approach, which is described in the next section, where we use item data for generating pairs. This approach allowed us to incorporate richer item-related features, enabling the model to better differentiate between similar and dissimilar items.

4.3. Approach 2: Using Item Data for Pair Generation

In this approach, we shift our focus from transactional data to item-related features for generating positive and negative pairs. This method allows us to capture richer item representations by leveraging descriptive data such as product group name, product type name, color group, garment group, and detailed descriptions.

4.3.1 Data Preprocessing and Feature Engineering

We began by selecting a set of descriptive columns that represent the items. These columns include:

- Product group name
- Product type name
- Color group
- Garment group
- Detailed description

These features were combined to form a comprehensive textual representation of each item. After combining the features, we proceeded with data cleaning. This involved lowering the case of all text, removing non-alphanumeric characters, and tokenizing the text. Additionally, we removed repetitive words and stopwords to ensure the data was suitable for vectorization.

4.3.2 Vectorization and Embedding Generation

Next, we used the TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer to transform the preprocessed text data into feature vectors, with a maximum of 500 words per item. These vectors served as a numerical representation of the item descriptions, capturing the most significant words that describe each item.

To further refine these representations, we utilized BERT (Bidirectional Encoder Representations from Transformers), a pre-trained model known for generating high-quality embeddings. The TF-IDF vectors were passed through BERT to generate embeddings that encapsulate semantic meaning, providing a richer and more nuanced representation of each item.

4.3.3 Pair Generation Using FAISS

Once we had the BERT embeddings for each item, we employed FAISS (Facebook AI Similarity Search) to find the 5 most similar items for each item in the dataset. FAISS efficiently computes similarities between high-dimensional vectors, allowing us to identify the closest matches based on the item embeddings. These similar items became our positive pairs.

Negative pairs were generated by randomly selecting items that were not part of the 5 most similar ones, ensuring a contrast between positive and negative examples for training the model.

4.3.4 Dataset and Training Setup

This approach resulted in a total of 246,226 pairs, which we split into an 80-20 training and testing set. The training set consisted of 196,981 pairs, while the test set included 49,245 pairs. With this dataset, we applied the contrastive learning model, as described in previous sections, to train the model and learn effective item embeddings.

4.3.5 Model Training

The positive and negative pairs generated through FAISS and random selection were then fed into the contrastive learning model. The model was trained to minimize the distance between positive pairs (similar items) and maximize the distance between negative pairs (dissimilar items), enabling it to learn useful representations of items based on their descriptive features.

This approach, by using item descriptions and semantic embeddings, addressed the limitations of transactional data and improved the model's ability to generate meaningful recommendations based on item similarity.

4.4. Training Procedure

The model is trained using stochastic gradient descent (SGD) or Adam optimizer to minimize the NT-Xent loss. The training process involves the following steps:

1. **Input**: A batch of pairs (x_1, x_2), representing either textual descriptions or image embeddings, is passed through the model,
2. **Feature Extraction**: The feature extractor generates embeddings for each item in the pair,
3. **Projection**: The projection head transforms the embeddings into a space suitable for contrastive learning,
4. **Loss Calculation**: The NT-Xent loss is computed based on the cosine similarity between the projections of the pair,
5. **Backpropagation**: Gradients are computed and used to update the model's weights via backpropagation,
6. **Optimization**: The optimizer updates the model parameters using the gradients.

The optimizer used is Adam with a learning rate of 0.001, and the learning rate schedule is adjusted using CosineAnnealingLR. The model is trained for a total of 50 epochs, with a batch size of 256.

4.5. Mathematical Formulation of Training

Given a batch of N pairs (x_1, x_2), the model learns embeddings $\mathbf{p}_1$ and $\mathbf{p}_2$ for each pair. The contrastive loss is computed for each pair (x_1, x_2) as:

$$L_{\text{batch}} = \frac{1}{N} \sum_{i=1}^{N} L_{\text{NT-Xent}}(\mathbf{p}_i, \mathbf{p}_j)$$

where $L_{\text{NT-Xent}}(\cdot, \cdot)$ is the NT-Xent loss function. The loss is backpropagated to update the parameters of the feature extractor and projection head, minimizing the embedding space distance between similar items while maximizing the distance between dissimilar items.

4.6. Key Findings / Results

To evaluate the performance of the contrastive learning model, we calculated the average cosine distances between positive and negative pairs, which were generated using the item data-based approach described in Approach 2. This evaluation helps assess whether the model was able to effectively learn to distinguish between similar (positive) and dissimilar (negative) pairs.

4.6.1 Pair Classification

We began by classifying the test pairs into positive and negative pairs. The positive pairs were those items that were identified as similar based on their embeddings, and the negative pairs were randomly selected items that were dissimilar. The following code snippet shows the classification process:

```
# Initialize empty lists for positive
and negative test pairs
positive_test_pairs = []
negative_test_pairs = []
# Loop through test pairs and classify
them for pair in test_pairs:
    if pair in positive_pairs:
        positive_test_pairs.append(pair)
    elif pair in negative_pairs:
        negative_test_pairs.append(pair)

# Output
print(f"Positive Test Pairs:
{len(positive_test_pairs)}")
print(f"Negative Test Pairs:
{len(negative_test_pairs)}")
```

This resulted in the following number of test pairs:

Positive Test Pairs: $24,524$ Negative Test Pairs: $24,722$

4.6.2 Cosine Distance Calculation

To further evaluate the model, we calculated the cosine distances between the embeddings of the positive and negative pairs. The embeddings were generated through the BERT-based approach described in Approach 2, and we used PyTorch's cosine_similarity function to compute the cosine similarity. The cosine similarity was then converted to cosine distance using the formula:

$$\text{Cosine Distance} = 1 - \text{Cosine Similarity}$$

The cosine distances for both positive and negative pairs were calculated in batches to optimize the evaluation process. The results are summarized below:

```
# Positive Cosine Distances
positive_cosine_distances =
(1 - cosine_sim).tolist()

positive_average_cosine_distance =
sum(positive_cosine_distances) /
len(positive_cosine_distances)

print(f"\nAverage Cosine Distance
for all positive pairs:
{positive_average_cosine_distance:.4f}")
```

After evaluating all the positive test pairs, the average cosine distance was found to be:

Average Cosine Distance for Positive Pairs: 0.2286

Similarly, the average cosine distance for the negative pairs was calculated using the same method:

```
# Negative Cosine Distances
negative_cosine_distances =
(1 - cosine_sim).tolist()

negative_average_cosine_distance =
sum(negative_cosine_distances) /
len(negative_cosine_distances)

print(f"\nAverage Cosine Distance
for all negative pairs:
{negative_average_cosine_distance:.4f}")
```

The average cosine distance for the negative pairs was found to be:

$$\text{Average Cosine Distance for Negative Pairs: } 0.5849$$

4.6.3 Training Results

After 50 epochs of training, the contrastive learning model showed a significant increase in the distance between positive and negative pairs. Specifically, the average cosine distance for positive pairs (0.2286) was much smaller compared to that for negative pairs (0.5849), indicating that the model successfully learned to differentiate between similar and dissimilar items.

The substantial difference in cosine distances between the positive and negative pairs confirms that the contrastive learning approach effectively minimized the distance for similar items while maximizing it for dissimilar items, achieving the desired behavior of the model after 50 epochs.

5. Discussion and Conclusion

5.1. Summary of Findings

This study explored the effectiveness of contrastive learning for fashion item recommendation using transactional and item data. The goal was to assess the model's ability to distinguish between similar and dissimilar items based on embeddings. Two approaches were examined:

- **Transactional Data Approach**: Initially, we used transaction history to generate positive pairs (items purchased together) and negative pairs (items not purchased together). However, this approach showed limitations as the cosine distance between positive and negative pairs converged to a similar value (close to 1) after 50 epochs, indicating model decay and insufficient differentiation capability when relying solely on transactional data.
- **Item Data Approach**: We then used item data, including product descriptions, group names, colors, and other textual features. After tokenizing the text and generating embeddings with BERT, we employed Faiss for similarity-based pair generation. Positive pairs were selected by finding the top 5 most similar items for each product, while negative pairs were randomly selected. This approach resulted in improved model performance, with clear differentiation between positive and negative pairs (cosine distances: 0.2286 for positives, 0.5849 for negatives), confirming the effectiveness of contrastive learning in this context.

5.2. Challenges

While the second approach yielded promising results, several challenges were encountered:

- **Transactional Data Limitations**: The transactional data approach led to model decay, where the cosine distance between positive and negative pairs became indistinguishable. This suggested that transaction history alone does not provide sufficient features for effective contrastive learning.
- **Textual Data Alignment Issues**: Although the second approach with textual embeddings showed improvement, it was not without issues:
  - Textual descriptions sometimes did not accurately reflect the visual characteristics of items, leading to the incorrect generation of positive and negative pairs. This misalignment between text and image data impacted the training process.
  - One potential solution to this issue is the incorporation of multi-modal models, such as **CLIP (Contrastive Language-Image Pretraining)**, which can align both textual and visual features, ensuring better pair generation.
- **Computational Cost**: Generating and processing embeddings for the large dataset (over 246,000 pairs) was computationally expensive. While Faiss helped mitigate this issue by improving the efficiency of similarity search, handling such a large dataset still required considerable resources.
- **Negative Pair Selection and Dataset Balance**: Balancing the dataset and managing the random selection of negative pairs introduced additional complexity. An imbalance in the dataset could potentially bias model training and affect performance.

5.3. Implications

The findings from this study have several important implications:

- **Leveraging Product Features**: The success of the second approach, which used item descriptions and textual

features, highlights the importance of incorporating detailed product information in recommendation models. Expanding this approach to include attributes such as material, size, and style could further improve recommendation accuracy.

- **Contrastive Learning in Fashion Recommendations**: This study demonstrates that contrastive learning can effectively identify item similarities. Using advanced embedding techniques like BERT, combined with Faiss for similarity-based pair generation, can yield high-quality recommendations without relying heavily on user-specific data. This is particularly useful in scenarios where user data is sparse or unavailable.
- **Hybrid Models**: The results suggest that combining transactional and item data could leverage the strengths of both approaches. Future work should explore hybrid models to capitalize on this potential.
- **Future Research Directions**: Future research should focus on:
  - Incorporating multi-modal models (e.g., CLIP) to better align textual and visual features.
  - Addressing challenges related to negative pair selection and dataset balancing.
  - Expanding the scope of item data to include more product attributes, which could further enhance recommendation accuracy.

Overall, the findings contribute to the growing body of research on contrastive learning-based recommendation systems and demonstrate the potential of using product feature data and advanced embedding techniques to improve fashion recommendation systems.

5.3.1 Conclusion

This study explored contrastive learning for fashion item recommendation using transactional and item data. While transactional data provided some insights into user behavior, it was insufficient for effective training, leading to model decay. The second approach, which utilized item descriptions and textual embeddings, showed more promise by generating accurate and diverse positive pairs. However, misalignment between textual descriptions and item visuals presented a challenge, leading to incorrect pair generation.

Despite these challenges, the second approach demonstrated the potential of contrastive learning in fashion recommendations, outperforming the transactional data approach. Future work could incorporate multi-modal models like **CLIP** to align textual and visual features, improving the generation of positive and negative pairs. By overcoming these challenges, contrastive learning could become a powerful tool for generating personalized and visually relevant fashion recommendations.

This paper contributes to the research on contrastive learning for fashion recommendation systems and offers valuable insights into using transaction history and item metadata for this task. The findings provide a foundation for further research aimed at enhancing the accuracy and scalability of recommendation systems in the fashion industry.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607. PMLR, 2020. 3

[2] Yashar Deldjoo, Fatemeh Nazary, Arnau Ramisa, Julian McAuley, Giovanni Pellegrini, Alejandro Bellogín, and T. D. Noia. A review of modern fashion recommender systems. *ACM Computing Surveys*, 56:1 – 37, 2022. 2

[3] Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong. Contrastive learning for recommender system. In *Proceedings of the Conference on Recommender Systems*, 2021. 4

[4] Ambareesh Revanur, Vijay Kumar, and Deepthi Sharma. Semi-supervised visual representation learning for fashion compatibility. In *ACM Conference on Recommender Systems (RecSys)*, 2021. 6

[5] Karin Sevegnani, Arjun Seshadri, Tian Wang, Anurag Beniwal, Julian McAuley, Alan Lu, and Gerard Medioni. Contrastive learning for interactive recommendation in fashion. In *ACM SIGIR Workshop on eCommerce*, 2022. 5

[6] Muhammad Khiyarus Syiam, Agung Toto Wibowo, and Erwin Budi Setiawan. Fashion recommendation system using collaborative filtering. *Building of Informatics, Technology and Science (BITS)*, 5(2):376–385, 2023. 3