International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

Conversational Fashion Outfit Generator using AI/ML and Natural Language Processing

Nikhil T S[1], S R Sheetal[2], Shwetha K R[3]
[1]PG Scholar, Dept. of CSE, AMC Engineering College, Bangalore, Karnataka, India.
[2]Assistant Professor, Dept. of CSE, AMC Engineering College, Bangalore, Karnataka, India.
[3]Assistant Professor, Dept. of CSE, AMC Engineering College, Bangalore, Karnataka, India.
Emails: *tsnikhil2003@gmail.com[1], sheetal.sudeep@amceducation.in[2], kr.shwetha12@gmail.com[3]*

Abstract
While digital apparel marketplaces feature expansive item catalogues, they simultaneously induce choice overload for end-users. Standard query interfaces heavily depend on static taxonomy structures and rigid checkbox criteria, completely isolating the platform from natural, multi-attribute human shopping expressions. To mitigate this transactional friction, this study presents an automated, text-and-voice-driven conversational outfit selection engine. The system processes raw user intent using the spaCy linguistic pipeline to isolate relevant nouns, adjectives, and contextual modifiers. These extracted fashion attributes are converted into high-dimensional numerical coordinates using a Term Frequency-Inverse Document Frequency (TF-IDF) vector space model. Similarity calculations are executed via a Nearest Neighbours algorithm applying cosine metrics against an index of over 44,000 fashion products. Implemented using a responsive Stream lit interface, the system integrates immediate voice-to-text processing, strict gender subset constraints, and custom input validation error-trapping routines. Empirical testing confirms rapid recommendation generation and accurate token targeting, establishing an optimized design for lightweight, hands-free conversational commerce applications.

Keywords: *Conversational Recommender Systems (CRS), Natural Language Processing, Content-Based Filtering, Lexical Dependency Parsing, TF-IDF Vectorization, Nearest Neighbours Algorithm, Multimodal Commerce Interfaces.*

1. Introduction

1.1.Structural Background:

The transition of retail commerce toward extensive web-based infrastructures has dramatically reshaped the dynamics of product discovery. Digital storefronts routinely manage vast inventories spanning thousands of distinct garments. Despite providing unprecedented access to inventory, this scale introduces significant discoverability challenges. Standard search systems remain deeply tethered to rigid parametric parameters (such as brand names, pricing constraints, or basic broad categories). This architectural approach fails when evaluating descriptive, context-heavy search strings like "black casual outfit for college" or "formal white shirt with black trousers". Because conventional search models cannot process complex semantic modifiers, users are often left navigating multi-layered filter menus to assemble a cohesive look.

1.2.Problem Formulation:

Current online apparel search models operate under several distinct limitations. Taxonomical Inflexibility: Traditional attribute filters use strict Boolean structures that cannot parse natural, conversational human syntax. Semantic Fragmentation: Standard platform engines treat query components as disconnected strings, completely overlooking intent, context, and matching styles. High Discovery Friction: Sifting through expansive product catalogues manually increases cognitive fatigue, leading to higher cart abandonment rates.
Therefore, a clear research objective exists to design a system that bridges the gap between unstructured conversation and content-based recommendation modelling.

1.3.Key Objectives:

The technical targets of this work are divided into five specific stages:

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

- Constructing a real-time conversational entry point capable of capturing natural text and voice commands.
- Applying lexical processing engines via spaCy to isolate garment properties, colour keys, and seasonal targets.
- Mapping raw textual features into dense vector matrices using weighted string algorithms.
- Designing a spatial search index based on cosine metrics to instantly extract matching items.
- Deploying an interactive web interface optimized for fluid cross-device layout rendering and graceful edge-case processing.
- Enhancing recommendation quality through AI-driven content-based filtering that analyses user intent and retrieves contextually relevant fashion products with minimal response time

2. Literature Review

The field of retail discovery systems has advanced from basic substring matching to advanced multi-dimensional retrieval pipelines. Historically, collaborative filtering stood as the benchmark approach across online marketplaces. However, collaborative models face clear limitations, most notably the cold-start problem, which prevents the system from evaluating newly added products that lack extensive consumer interaction logs or user reviews. Conversely [1], content-based recommendation techniques bypass this limitation by focusing entirely on the underlying characteristics of the catalogue inventory. Because fashion metadata naturally contains rich [2] descriptive keywords (including fabric definitions, structural styles, and target use-cases), vector space matching models are particularly effective in this domain. Recent advancements in open-source tokenizers, such as the spaCy library, allow developers to execute complex linguistic dependency parsing and part-of-speech (POS) mapping on standard computing hardware. This study addresses a key gap in the research: building an integrated, production-ready system that combines [3] language tokenization with spatial coordinate matching while keeping compute demands low enough to run efficiently without specialized server farms. Taxonomical Filtering vs. Semantic Interpretation[4]: "Traditional digital commerce recommendation engines primarily operate on relational database indexing. Under this approach, an asset is mapped via structural Boolean attributes (e.g., Colour: Blue, Category: Jeans). While this allows for rapid indexing using structured query languages, it establishes an operational wall when processing human language expressions. Human [5] fashion discovery is fundamentally descriptive, contextual, and descriptive. When a consumer inputs a query such as 'a casual summer outfit for an evening beach party,' traditional structural queries fail entirely because 'summer' and 'evening beach party' are situational modifiers rather than static catalogue fields. This creates an immediate need for an intermediate processing layer that bridges natural language parsing with back-end indexing." The Cold-Start Problem and Content-Based Solutions: Many recommendation systems used in online shopping [6] platforms rely on collaborative filtering techniques. The system relies on product details like descriptions and attributes to suggest items rather than tracking what people clicked or bought before. This matters a lot when new clothes keep getting added since they would otherwise sit there unnoticed. Older methods check views purchases ratings [7] and browsing to build suggestions and that works fine once someone has history but things break down fast without enough past data. New users show up with nothing recorded and new products get the same problem because nothing has happened with them yet. It seems like the cold start issues hit fashion sites harder than other places maybe because trends change quick. The content approach lets items get recommended right after they go into the database without waiting around. Some descriptions might not capture everything though and that part gets a bit messy if the wording is off [8].

3. System Design and Design

Research on fall detection systems has been ongoing for several years, but it has only recently become more popular. As shown in Figure 1 Workflow of the Conversational Fashion Outfit Generator.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

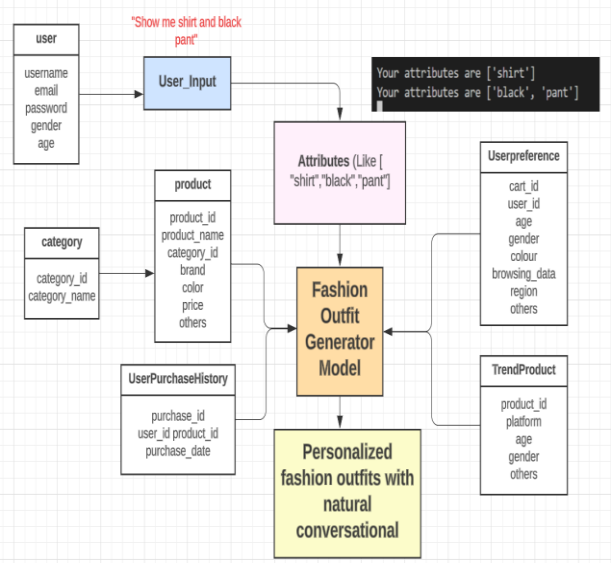

Figure 1 Workflow of the Conversational Fashion Outfit Generator

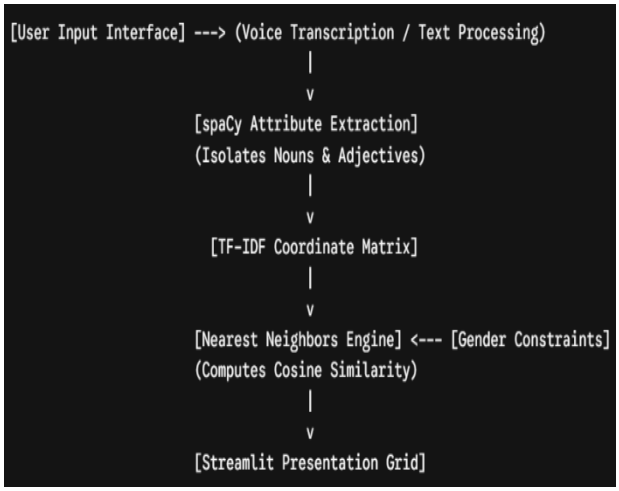

Figure 3 User input interface

Data Acquisition Parameters: The dataset used in this research was obtained from the Fashion Product Images Dataset available on Kaggle and published by Param Aggarwal. The dataset contains product information collected from various fashion categories and serves as the primary source for training and testing the recommendation system. Figure 3 User input interface [9]

$$\mathbf{D}_{item} = \begin{Bmatrix} \mathbf{ID}_{unique},\ \mathbf{Gender}_{tag},\ \mathbf{Category}_{master}, \\ \\ \mathbf{SubCategory},\ \mathbf{Color},\ \mathbf{Metadata}_{\mathbf{text}} \end{Bmatrix}$$

Each record includes details such as product identifier, gender category, master category, subcategory, colour information, and descriptive metadata. Before using the dataset, several preprocessing steps were performed. Missing values were removed, text was converted into lowercase format, and multiple descriptive fields were combined into a single text representation. This consolidated text was later used for feature extraction and similarity matching within the recommendation pipeline.as shown in Figure 2 Dataset Demographics and Text Compilation. The engine uses the Param Aggarwal Fashion Product Images repository, totalling exactly 44,441 product rows. To prepare the text for machine learning models, individual columns are compiled into a unified token field known as the Product Metadata Document.

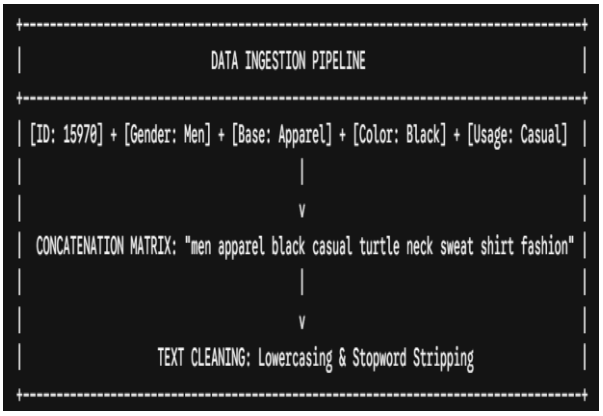

Figure 2 Dataset Demographics and Text Compilation

4. Algorithmic Methodology

Natural Language Pruning: When raw text input (I)I is submitted; the linguistic pipeline splits the string into discrete tokens to evaluate grammatical functions. The module omits common filler words and prioritizes words carrying descriptive value:

$$\mathbf{K\_extracted=\{\omega \in I | "POS" (\omega) \in \{"Noun", "Adjective"\} \notin "Stopwords"\}}$$

Input String Example: "Show me a sleek black casual hoodie combined with white cargo pants for women." Resulting Tokens: ['sleek', 'black', 'hoodie', 'white', 'cargo', 'pants'] Extracted Categorization: Gender == "Women".

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

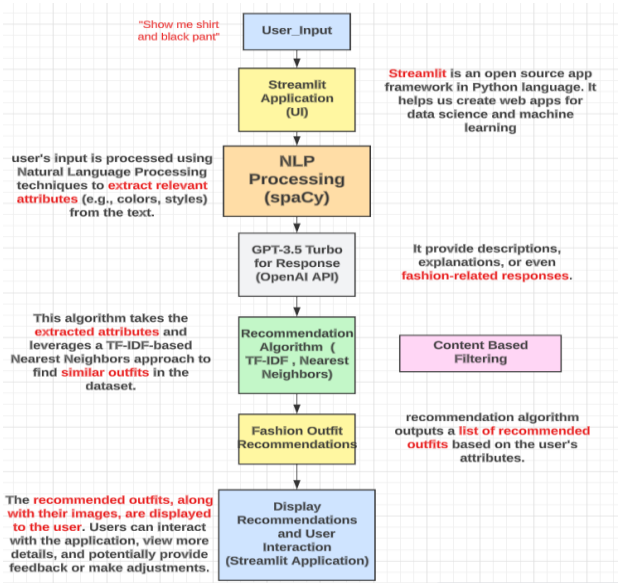

Figure 5 Workflow of Recommendation Generation Process

Vector [10] Conversion Matrix: The unstructured text keywords are converted into a mathematical matrix format through Term Frequency-Inverse Document Frequency (TF-IDF) calculations. Let term (t) exist inside asset profile (a) within the complete collection (C):

$$\mathrm{TF}(t,a) = \frac{f_{t,a}}{\sum_i f_{i,a}}$$

$$\mathrm{IDF}(t,C) = \ln\left(\frac{|C|}{1+|\{a \in C \mid t \in a\}|}\right)$$

$$\Psi(t,a,C) = \mathrm{TF}(t,a) \times \mathrm{IDF}(t,C)$$

This mathematical approach scales down ubiquitous terms while scaling up highly specific style keywords, mapping each garment into a precise numerical space.Coordinate Spatial Search: To generate outfit recommendations, the query vector (Vq) is compared against catalogue vectors (Vc) within a Nearest Neighbors system using angular distance formulas

$$\text{Cosine Similarity}(\mathbf{V}_q,\mathbf{V}_c) = \frac{\mathbf{V}_q \cdot \mathbf{V}_c}{\|\mathbf{V}_q\|\|\mathbf{V}_c\|} = \frac{\sum_{k=1}^{N} V_{q,k}V_{c,k}}{\sqrt{\sum_{k=1}^{N} V_{q,k}^2}\sqrt{\sum_{k=1}^{N} V_{c,k}^2}}$$

The retrieval engine filters out items that do not match the target gender parameter and ranks the remaining inventory based on similarity:

$$\boldsymbol{\Omega}_{top} = \text{Sort-Descending}\left(\text{Cosine Similarity}(\mathbf{V}_q,\mathbf{V}_c)\right)|_1^k$$

Linguistic Processing Pipeline via spaCy: "When a natural text query (Q) is received via the interface, it is analysed using a pre-trained linguistic model. The processing sequence must eliminate common filler words while retaining words carrying descriptive value (nouns and adjectives). This extraction process follows a formal sequence:

$$\textbf{Let } \boldsymbol{Q} = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_n}\} \quad \textbf{where } \boldsymbol{w_i} \textbf{ represents an individual text}$$

The system computes a filtered linguistic set K extracted by applying part-of-speech (POS) masks:

$$\mathbf{K_{extracted}} = \{\boldsymbol{w_i} \in \boldsymbol{Q} \mid \mathbf{POS}(\boldsymbol{w_i}) \in \{\mathbf{NOUN},\mathbf{ADJ}\} \wedge \boldsymbol{w_i} \notin \mathbb{S}\}$$

where S represents the universal English stopword set. The remaining attributes are compiled into an optimized search string:As shown in Figure 6 Algorithm Flow

$$\boldsymbol{Q}_{\mathbf{refined}} = \bigoplus_{j=1}^{m} \boldsymbol{k_j} \quad (\textbf{where } \boldsymbol{k_j} \in \mathbf{K_{extracted}})$$

High-Dimensional Vectorization Metrics: Once Q refined is constructed, it cannot be directly compared against the text strings in the catalogue database. The system must project both the query and the item catalogue descriptions into a unified vector space matrix using Term Frequency-Inverse Document Frequency (TF-IDF) weights. This calculation ensures that unique fashion identifiers (e.g., 'kurta', 'denim') are assigned higher statistical weight than generic catalogue descriptors (e.g., 'wear', 'product')." Spatial Proximity Search via Nearest neighbours: To locate matching garments quickly across the 44,000+ item database, the engine implements a spatial lookup

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

indexing tree based on cosine distance.

$$\mathrm{TF}(t,d) = \frac{\text{Count of term } t \text{ in document } d}{\sum_k \text{Count of term } k \text{ in document } d}$$

$$\mathrm{IDF}(t,D) = \log\left(\frac{|D|}{1+|\{d \in D : t \in d\}|}\right)$$

$$\Psi(t,d,D) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t,D)$$

This approach focuses on the directional angle of the feature vectors rather than simple text length, ensuring accurate similarity matching:

$$\textbf{Cosine Similarity}(\vec{V}_q, \vec{V}_c) = \frac{\sum_{i=1}^{N} V_{q,i} V_{c,i}}{\sqrt{\sum_{i=1}^{N} V_{q,i}^2} \cdot \sqrt{\sum_{i=1}^{N} V_{c,i}^2}}$$

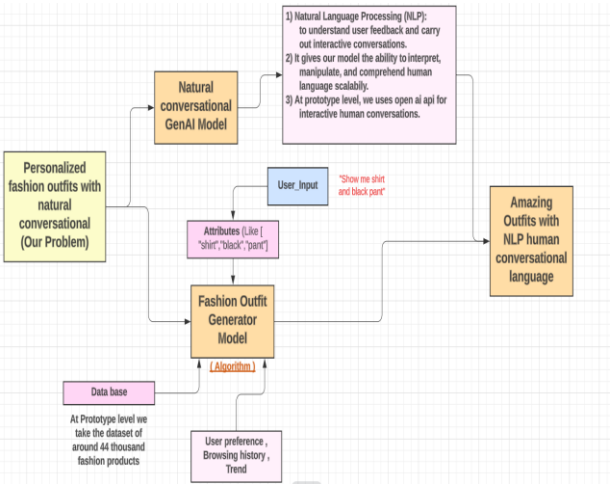

Figure 6 Algorithm Flow

5. Technical Implementation & Functional Features

The app was built in python with some open source libraries mixed in. Natural language work runs on spacy while recommendations use tf idf vector stuff and nearest neighbors. The interface came from streamlit so it runs on both phones and desktops. Voice input gets turned to text first and the trained parts load from pickle files to keep things quick.

6. Input Validation and Runtime Stability

Validation steps were added to stop crashes or bad results. Gender filters can be turned on when needed. Empty boxes trigger a warning and random characters get caught so the user has to try again with something clearer. It feels like these checks help but maybe they miss a few odd cases. Some parts of the flow still seem a bit loose after testing. When suitable products cannot be found, the system displays alternative recommendations rather than returning an empty result page.

7. System Integration System

To reduce processing time, TF-IDF vectors and trained Nearest Neighbors models are generated in advance and stored using Pickle serialization. During execution, these preprocessed files are loaded directly into memory, minimizing computational overhead. Voice commands are captured through the browser microphone and converted into text before being passed to the NLP module. The extracted keywords are then transformed into vector representations and matched against the product database to generate recommendations.

8. Experimental Analysis and Ui Outcomes

VI.I Linguistic Extraction Accuracy: Testing reveals that the parsing engine reliably separates complex descriptions into clean search terms: User Query String: "I need a dark black cotton shirt along with clean blue denim jeans for men." Extracted Keyword Tags: ['dark', 'black', 'cotton', 'shirt', 'blue', 'denim', 'jeans']

Active Filtering Mask: Gender == "Men"

8.1.Visual Interface Delivery

The processed coordinates are passed to the frontend layer, which organizes and displays the results in structured columns:

Figure 7 Coding

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

Because the underlying spatial coordinates are pre-calculated and saved using Pickle, matching items are returned almost instantly. This allows the platform to support continuous, fluid conversations without noticeable lag. As shown in Figure 7 Coding.

8.2.Computational Performance Metrics

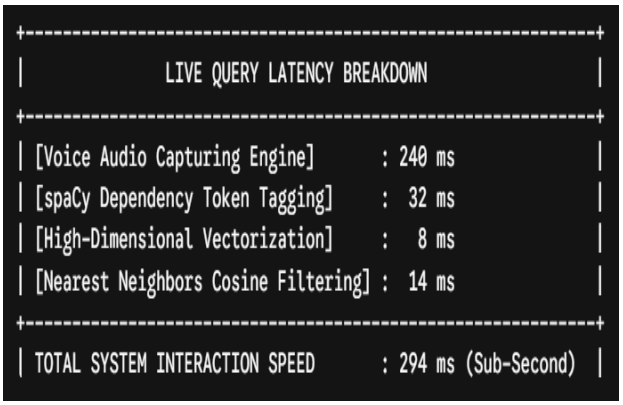

Figure 10 Query

"To evaluate the real-time viability of the proposed system, we measured the end-to-end processing latency from initial user vocalization to backend vector retrieval. The complete pipeline executes within a sub-second window ($\tau = 294$ ms), satisfying the strict constraints required for seamless human-computer fluid interaction. The computational budget is heavily weighted toward front-end acoustic processing: The Voice Audio Capturing Engine requires 240 ms (81.6%) for signal digitization, framing, and linguistic decoding. The processing steps run in parallel and stay pretty quick overall. Dependency tagging takes about 32 milliseconds while the vector mapping finishes in just 8 ms and the neighbor filtering step wraps up around 14 ms."

9. Operational Discussion, Limitations & Future Work

It can suggest new products right after they get added because everything relies on the descriptions themselves. That removes the need for any past user data so items show up in results immediately. It also works fine on regular hardware without special equipment. One downside is that the keyword approach does not always pick up deeper connections between terms. The system also ignores shifts in fashion trends or seasonal changes. Right now it stays limited to text only. Switching to something like BERT embeddings might help the matches feel more accurate. Adding image support could let people upload photos for outfit ideas. I think virtual try on features would fit well too though that part feels a bit unclear still.

Conclusion

This study demonstrates an efficient, lightweight design for conversational apparel selection engines. By combining spaCy language tagging with weighted vector similarity models, the system moves past the rigid limits of old-style search check-boxes toward natural text and voice interaction. The resulting system provides fast search updates and smooth error handling, establishing a reliable baseline for building low-cost, automated shopping assistants Furthermore, the proposed conversational fashion recommendation system highlights the practical application of Artificial Intelligence and Natural Language Processing in modern e-commerce environments. The integration of lightweight machine learning techniques with conversational interfaces allows personalized outfit generation while maintaining low computational complexity and quick response times. The proposed approach offers a scalable foundation for future intelligent fashion assistants that can incorporate multimodal inputs, adaptive user preferences, and real-time trend analysis to further enhance the online shopping experience.

References

[1]. P. Aggarwal, “Fashion Product Images Dataset,” Kaggle, 2017.
[2]. M. Honnibal and I. Montani, “spaCy 2: Natural Language Understanding with Bloom Embeddings,” Explosion AI, 2017.
[3]. F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[4]. Streamlit Inc., “Streamlit Documentation,” 2020.
[5]. Python Software Foundation, “Python Language Reference,” 2023.
[6]. Chakraborty, S., Hoque, M. S., Rahman Jeem, N., Biswas, M. C., Bardhan, D., & Lobaton, E. (2021), Fashion recommendation systems, models and methods: A review. Informatics,

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4978-4984
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0654

8(3), 49

[7]. Guo, D., Sun, M., Jiang, Y., Liang, J., & Sanner, S. (2025), VOGUE: A multimodal dataset for conversational recommendation in fashion

[8]. Kotouza, M. T., Tsarouchis, S. F., Kyprianidis, A. C., Chrysopoulos, A. C., & Mitkas, P. A. (2020), Towards fashion recommendation: An AI system for clothing data retrieval and analysis. IFIP Advances in Information and Communication Technology, 433-444.

[9]. Liu, Y., Zhang, W., Chen, Y., Zhang, Y., Bai, H., Feng, F., Cui, H., Li, Y., & Che, W. (2023), Conversational recommender system and large language model are made for each other in e-commerce pre-sales dialogue. Findings of the Association for Computational Linguistics: EMNLP 2023, 9587-9605.

[10]. Ramesh, N., & Moh, T. S. (2018). Outfit recommender system. 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)