Learning Set Embeddings for Fashion Compatibility Recommendation[*]

Indra Firmansyah
University of Hildesheim
Hildesheim, Germany
indra.frms15@gmail.com

Randolf Scholz
University of Hildesheim
Hildesheim, Germany
rscholz@ismll.de

Adrian Nahmendorff
University of Hildesheim
Hildesheim, Germany
nahmendorff@uni-hildesheim.de

Ngoc Son Le
University of Hildesheim
Hildesheim, Germany
sle@ismll.de

Shereen Elsayed
University of Hildesheim
Hildesheim, Germany
elsayed@ismll.de

Lars Schmidt-Thieme
University of Hildesheim
Hildesheim, Germany
schmidt-thieme@ismll.de

ABSTRACT

In recent years, learning outfit compatibility patterns from human-generated fashion outfits has gained attention from both academia and industry due to its importance to the generation of recommendations on fashion e-Commerce platforms. The researches in this area mainly tackle two relevant tasks: Fashion Fill-in-the-blank (Fashion FITB) and Outfit Complementary Item Retrieval (Outfit CIR). This work presents *Sliced-Wasserstein Fashion Compatibility Network* or *Slay-Net* to tackle these tasks. In the proposed approach, fashion outfits are modeled as set-structured data to capture the complex relationship between an item and the rest of the items within an outfit. *Slay-Net* includes an innovative approach to learn to generate the set embedding of a fashion outfit by using an attention-based set encoding and Pooling by Sliced-Wasserstein Embedding (PSWE). Furthermore, the training of *Slay-Net* follows a curriculum learning framework that includes simultaneous training for binary classification and contrastive learning through multi-task learning. Among recent related works, *Slay-Net* is able to achieve the best performance in the Outfit CIR task, as measured by the Recall@top-k metric. Experiments conducted on a real-world dataset shows that Slay-Net improves the performance in Outfit CIR task by up to 27.16%. The implementation of Slay-Net is made publicly available at https://github.com/1503-firmansyah-indra/slay-net.

CCS CONCEPTS

• **Information System → Recommender System**.

KEYWORDS

Recommendation System, Fashion Outfit Compatibility, Set-structured Data, Contrastive learning

1 INTRODUCTION

Fashion e-Commerce is an internet business that sells fashion items online. Each fashion item has its own internet page, where pictures and various details of the item are displayed. This page is commonly referred to as the *product page*. On the product page of an item on most fashion e-Commerce platforms, there is a section to show other items that customers may also be interested in, and this section is normally called *recommendation section*. Different methods can be used to generate these recommendations, and these methods are normally data driven.

There are two main types of data that are used to power the recommendations generation methods; *user-item* and *item-item* interaction data. User-item interaction data capture information such as the items which a user has interacted with, the type of interactions and a measure of the interaction. In fashion e-Commerce platforms, customers can interact with an item in different ways: for example, a customer can view the product page of an item, put an item in their wishlist, or rate an item that they have purchased before. However, the item-item interaction contains information about the interaction between items. For example, two items that are purchased or viewed together can be said to have interacted.

This work focuses on the use of item-item interaction data for curating items for the recommendation section of a product page. The specific item-item interaction data is the one derived from human generated fashion outfits. A fashion outfit, or simply an outfit, is a set of items that look visually compatible when worn by a person. The number of items in an outfit is not fixed, and this feature allows us to model outfits as set-structured data. The model proposed in this work aims to capture the patterns in the relationships of items in compatible outfit. The captured patterns can then be utilized to suggest a fashion item that can transform an incomplete outfit set into a compatible outfit. This is beneficial to fashion e-Commerce because customers tend buy an additional item that can be compatibly worn together with an item that they are already planning to buy.

Recent works that focus on learning patterns in compatible outfits focus on two task: Fashion Fill-in-the-blank (Fashion FITB) and Outfit Complementary Item Retrieval (Outfit CIR). Fashion FITB is a task where given a partially complete outfit, a model has to choose one item from a pool of four potential candidates to add into the outfit, so that the resultant outfit is a compatible one. Outfit CIR is similar to Fashion FITB, but the task is modified such that information retrieval setup is established. In Outfit CIR, the model has to retrieve k-many items from large pool of items, and these k items are those that have the high probabilities to transform the partially complete outfit into a compatible outfit. In this work, the Fashion FITB and Outfit CIR tasks will collectively be called Outfit Compatibility Learning.

Presented at the SURE workshop held in conjunction with the 18th ACM Conference on Recommender Systems (RecSys), 2024, in Bari, Italy.

Prior works [12, 13] adopt a pairwise item-level approach for the Fashion FITB task. However, these works are not specifically designed to perform the Outfit CIR task. [7] proposes a model to tackle the Fashion FITB and Outfit CIR tasks, while also adopting a pairwise item-level approach. Afterwards, [10] and [15] each proposes a model to tackle Outfit Compatibility Learning, but each outfit is modeled as a set. Modeling an outfit as a set introduces a new challenge; since the size of the outfit set is variable, there has to be a mechanism to derive a fixed-dimensional set embedding from the outfit set features, which have variable dimension. Both [10] and [15] adopt an attention-based set encoding method that includes the addition of a trainable token to the input to the set encoding process. However, this approach does not utilize all the encoded set features for generating the set embedding. That is why this paper presents an innovative approach to generate the fixed-dimensional set embedding using both set encoding and set pooling. Our approach enables the exploitation of all information from the encoded set features to generate the set embedding. The set encoding method is based based on Set Attention Block (SAB) [6]. The specific set pooling method used is Pooling by Sliced-Wasserstein Embedding (PSWE) by [8].

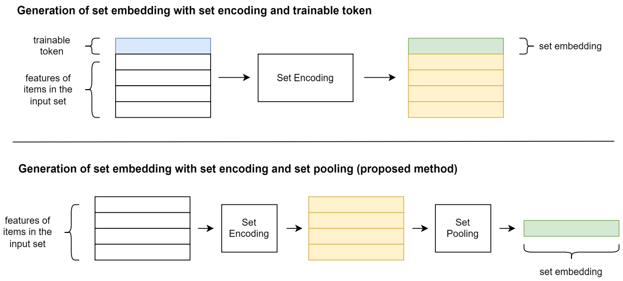
Generation of set embedding with set encoding and trainable token
trainable token
features of items in the input set
Set Encoding
set embedding
Generation of set embedding with set encoding and set pooling (proposed method)
features of items in the input set
Set Encoding
Set Pooling
set embedding

Figure 1: *top*: Illustration of set embedding generation using set encoding and trainable token. *bottom*: Illustration of set embedding generation using set encoding and set pooling. *all*: In both illustrations, the left most part (features of items in the input set) is the output of feature extraction process. The generated set embedding will then be processed further in Slay-Net to calculate the losses.

Furthermore, [10, 15] use a form of curriculum learning, in which the model is trained with binary classification and contrastive learning losses consecutively in separate phases. This work proposes a curriculum learning framework which includes a simultaneous training of binary classification and contrastive learning heads via multi-task learning. This is to allow the model to capture patterns that can only be discovered through simultaneous exposure to binary classification and contrastive learning dataset.

The model proposed in this paper is called *Sliced-Wasserstein Fashion Compatibility Network*, or shortly *Slay-Net*. Based on the empirical evidence collected using the Polyvore dataset [13], Slay-Net achieves the best performance metrics in the Outfit CIR task as compared to existing related works.

In summary, the main contributions introduced in this paper are the following.

- We introduce a new model, *Slay-Net*, which uses both an attention-based set encoding method and the set pooling method PSWE for generating the fixed-dimensional set embedding of a set-structured fashion outfit for the Fashion FITB and Outfit CIR tasks.
- Slay-Net is trained using a curriculum learning framework that includes the simultaneous training for binary classification and contrastive learning via multi-task learning.
- We evaluate the performance of Slay-Net on a real-world dataset, and we find that Slay-Net consistently improves the performance on Outfit CIR task over the best published baselines by up to 27.16%.

2 RELATED WORK

2.1 Outfit Compatibility Learning

One of the first prior works in Outfit Compatibility Learning models items in an outfit as a sequence and proposes a bidirectional LSTM model [4]. The same work also introduces the Fashion FITB task. In [13], each item in the outfits is projected into type-respecting latent spaces and the compatibility between items is learned using contrastive learning in a pairwise manner. However, the drawback of this approach is that is that it may not perform well on the item type that has not been seen during training. Hence, [12] develops a contrastive learning framework that does not require an explicit definition of latent spaces.

Other relevant prior works model items and the item-item relations as captured in fashion outfits as graph. In this approach, each item is modeled as a node, and if two items co-occur in an outfit, an edge is created between these two nodes. The models proposed by [3] and [11] use the graph autoencoder method. The models consist of an encoder and a decoder; the encoder generates embedding for each node based on the edges it has, while the decoder predicts if there is a potential edge between two nodes. If the decoder is confident that there is an edge between two nodes, there is a high probability that the items represented by the two nodes are visually compatible when worn together by a person.

To simulate a large-scale retrieval setup, which is a closer resemblance of an actual e-Commerce setup, [7] proposes a new task called Outfit Complementary Item Retrieval or shortly Outfit CIR. This is a task where given a partially complete outfit, the compatible item has to be efficiently retrieved from a large number of candidate items. A category sub-space attention network is proposed in [7] to tackle the Outfit CIR task. A loss called the Outfit Ranking Loss is used to perform contrastive learning by considering all item pairs in an outfit, with respect to a single reference item.

A modified version of the Outfit Ranking Loss is used to perform contrastive learning in [10], and the work also model outfits as set structured data. The modified loss is called Set-wise Outfit Ranking Loss. The complex relationship between items within an outfit is learned using self-attention mechanism. In [15], an outfit is also modeled as a set, but a cross-attention mechanism is utilized to learn the relationship between items in an outfit. Furthermore, the work adopts FashionCLIP [2] as the image feature extractor, and SentenceBERT [9] as the text feature extractor. The two feature extractor models are used with their weights frozen, and this results in a more efficient model training compared to previous related works.

2.2 Deep Learning on Set-structured Data

A different strategy is required when learning from set-structured data, as compared to learning from fixed-dimensional data. For example, the output of the model has to be the same regardless of how the items in the set are being permutated. Furthermore, the model has to be able to accommodate the varying size of the input set. An example of a set-structured data is fashion outfit.

[16] proposes a deep learning architecture for set-structured data via fully connected layers and aggregation of the transformed elements in the input set. While this architecture is able to achieve permutation equivariance, the intermediate encoding of the set elements is done in isolation. Hence, it cannot capture the interaction between elements within a set. To address this, [6] proposes an attention-based framework for learning from set-structured data, and it is called SetTransformer. SetTransformer consists of set encoding and set pooling steps.

Depending on the task that a model is tackling, the model architecture may include set encoding and set pooling to learn from set-structured data. There are prior works that focus on the set pooling step. [17] proposes a set pooling framework called Feature Sort Pooling (FSPool) that includes feature sorting across the set elements and weighted sum of the sorted set features. A method to generate a fixed-dimensional set embedding in a Euclidian space based on the generalized sliced Wasserstein distance [5] is proposed in [8]. This method is called Pooling by Sliced-Wassertstein Embedding (PSWE).

3 PROBLEM SETTING

This work is motivated by the problem of generating recommendation for fashion e-Commerce platforms using item-item interaction data. The specific item-item interaction data is one that is derived from human generated fashion outfit data. Two items that occur in at least one outfit are said to have interacted. A fashion outfit, or simply an outfit, is a set of items that look visually compatible when worn by a person. The items in an outfit are of different types. For example, an outfit can consist of a blouse, skirt, pair of shoes, and bag. Furthermore, the number of items in an outfit is variable. Some aspects of this recommendation generation problem are captured in researches in Outfit Compatibility Learning. Recent works in this area focus on two tasks: Fashion FITB and Outfit CIR. These two tasks are also the focus of this work.

In these two tasks, given a partially complete outfit $O_{partial} = \{I_i\}_{i=1}^{n_o} \setminus I_p$, the set embedding of the outfit f_o is to be generated. In this context, I_i is an item in the outfit, I_p an item removed from the outfit and n_o the number of items in the outfit, which can be different form outfit to outfit. To achieve a good performance in the two tasks, the model has to be trained so that the generated set embedding f_o is close to the embedding of the removed item f_p on a Euclidean space.

3.1 Fashion FITB

Fashion FITB is a problem in which an item has to be chosen from a selection of four potential items to transform a partially complete outfit into a compatible outfit. One can think of this problem as one where a blank from a partially complete outfit has to be correctly filled, and hence, it is named as Fashion Fill-in-the-blank. Performing Fashion FITB manifests in answering "FITB questions", and each question is a multiple choice question, where choosing the correct answer will result in compatible outfit. A better model will correctly answers more FITB questions and hence achieves higher FITB accuracy.

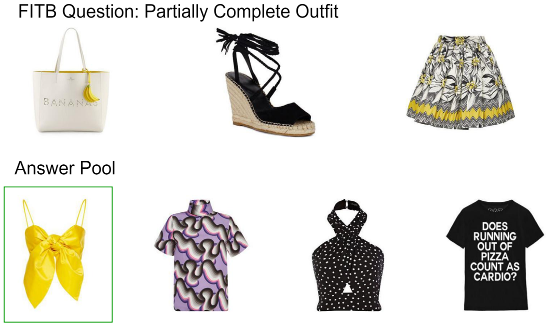
FITB Question: Partially Complete Outfit
BANANA
Answer Pool
DOES RUNNING OUT OF PIZZA COUNT AS CARDIO?

Figure 2: Illustration of dataset for FITB task by [13]. The first row shows an example of FITB question, and the second row shows the answer pool. The item with green outline is the correct answer.

Fashion FITB task can be formally defined as a multiple choice problem, where the goal is to complete $O_{partial}$ by selecting the most compatible item from a set of candidate items $\{I^i_{option}\}^4_{i=1}$, where $I_p \in \{I^i_{option}\}^4_{i=1}$. The correct answer to each FITB question is I_p, the item removed from the complete outfit. Let there be a parameterized embedding function g that takes in either a set of items or a single item, such that $g(O_{partial}) = f_o$ and $g(I_p) = f_p$. This work proposes Slay-Net as g. The prediction of the answer to each FITB question is obtained through a series of steps. Firstly, given a set of candidate items $\{I^i_{option}\}^4_{i=1}$, the Euclidean distance between f_o and the embedding of each candidate item $g(I^i_{option})$, where $i \in \{1, 2, 3, 4\}$, is calculated. The Euclidean distance is formally defined as follows:

$$d(f_o, g(I^i_{option})) = \|f_o - g(I^i_{option})\|_2 \tag{1}$$

The item, which embedding's Euclidean distance to f_o is shortest, is set as the prediction to the answer to the FITB question I_{pred}.

$$I_{pred} = \arg\min_{I_{option}} d(f_o, g(I_{option})) \tag{2}$$

where $I_{option} \in \{I^i_{option}\}^4_{i=1}$. The FITB question is correctly answered when I_{pred} is I_p.

FITB training and evaluation dataset is first introduced by [4], but is later improved by [13]. In the original FITB dataset, the pool of potential answers to an FITB question contains items of different types. For example, if the item type of the correct answer to an FITB question is shoes, the answers contain other item types, such as top or bottom. [13] modify the answer pool for each question so that it only contains items of the same type as the correct answer.

3.2 Outfit CIR

Although the Fashion FITB task described in the previous section does measure important aspects of the performance of models that are proposed in Outfit Compatibility Learning works, the small answer pool of size four is not representative of an actual e-Commerce system. Most e-Commerce operators have a large item catalog, and generating recommendation is a retrieval problem. For establishing a setup that is of closer resemblance to actual e-Commerce operations, the number of candidate answers for each FITB question has to be greatly increased. The modified Fashion FITB task with retrieval setting is the Outfit CIR task. In Outfit CIR task, the retrieval setting is manifested through the use of 3000 candidate items for each FITB question. As will be explained later, there is a hierarchy of item types and this hierarchy consists of two levels of item types. The candidate answers in Outfit CIR have the same item type as the correct answer across the two levels of item types.

The performance of the model in the Outfit CIR task is measured using the *Recall@top-k* metric. Given a large catalog $C = \{I^i_{cand}\}_{i=1}^{3000}$ containing 3000 candidate items, the Euclidean distance $d(f_o, g(I^i_{cand}))$ between f_o and the embedding of each candidate item is calculated according to equation (1). *Recall@top-k* measures the percentage of FITB questions where the correct answer I_p is in the top k candidate items with the shortest Euclidean distances.

4 SLAY-NET

In this section we present our proposed architecture the Sliced-Wasserstein Fashion Compatibility Network, or shortly Slay-Net. After a brief overview of the architecture, the individual components will be explained. The section concludes with a description of the curriculum learning framework.

4.1 Architecture Overview

The Slay-Net is trained by following a curriculum learning. In the first phase, binary classification and contrastive learning heads are simultaneously trained, while the second phase focuses on contrastive learning. Showing the two different flows (the binary classification and contrastive learning flow) gives a good overview of the architecture of Slay-Net.

Figure 3 shows the binary classification flow of Slay-Net, and this is only used in the first phase as a pre-training. The input to the model is a complete outfit $O_{full} = \{I_i\}_{i=1}^{n_o}$, where I_i is an item in the outfit and n_o the number of items in the outfit. No item is removed from a complete outfit. Each item in the outfit manifests as an image and text description. Feature extraction of the image and text are done using FashionCLIP [2] and SentenceBERT [9], respectively. These extracted features are then concatenated to give E_{set}, which is used as input for the set operation block. The first component of the set operation block is set encoding, and this is based on the Set Attention Block (SAB) [6]. The second component is set encoding via Pooling by Sliced-Wasserstein Embedding (PWSE) [8], and this outputs a fixed-dimensional outfit embedding x_o. The last component is the learning head specifically used for the binary classification pre-training. The dataset fed into the binary classification flow includes compatible and incompatible outfits. The binary classification training process trains Slay-Net to predict whether an input outfit is compatible or not.

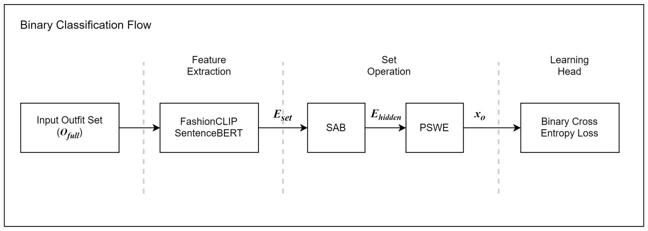

Figure 3: Binary Classification Pretraining Flow of Slay-Net

Figure 4 shows the contrastive learning flow of Slay-Net. It is used in both phases of the curriculum learning, and it has a more direct impact to the performance improvement of Slay-Net in the Fashion FITB and Outfit CIR tasks. Each instance of Contrastive Learning datasets consists of an anchor $O_{partial}$, a positive I_p and negative I_n. The anchor is a partial outfit, an outfit in which one of the items is removed. The positive is the item removed from the outfit. The negative is one or more items of the same general or fine-grained item type as the positive. Whether the negative is of the same general or fine-grained item type is decided based on which curriculum phase the learning is in. The anchor goes through both feature extraction and set operation step, while the positive and negative only go through feature extraction step. The set embedding of anchor x_o, embedding of positive x_p and negative x_n go through CSA-Net [7] to generate the final embedding of anchor f_o, positive f_p and negative f_n. The final embedding is also called CSA embedding. Afterwards, Set-wise Outfit Ranking Loss [10] is calculated.

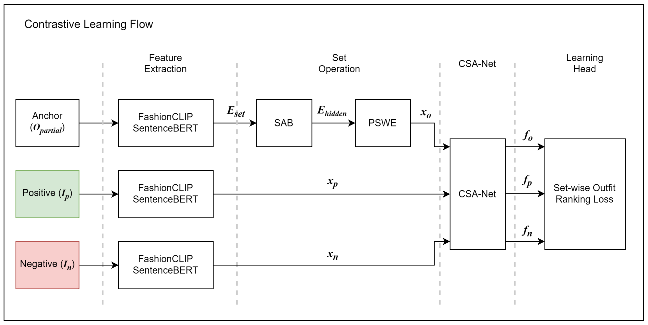

Figure 4: Contrastive Learning Flow of Slay-Net

4.2 Feature Extraction

The main input to Slay-Net is set-structured outfit data. Each outfit consists of a number of items, and each item manifests as an image and a text. Image and text are unstructured data, and since the latter part of the model expects the data to be structured, feature extraction needs to be performed to convert the unstructured data into structured data or embeddings. Hence, for each item, an image embedding $E_{img} \in \mathbb{R}^{d_{img}}$ and a text embedding $E_{text} \in \mathbb{R}^{d_{text}}$ are generated, where $d_{img} = d_{text} = 64$.

The feature extraction strategy of this work is mostly influenced by [15]. Figure 5 shows the *Image Encoder Block*, which is the network block within the feature extraction component that generates

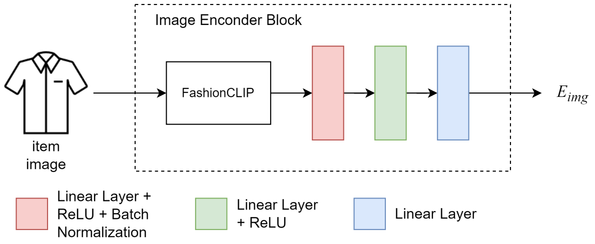

Figure 5: Illustration of *Image Encoder Block*

an embedding E_{img} from an image. The first component of the Image Encoder Block is the pre-trained FashionCLIP [2]. During training, the FashionCLIP weights are frozen. This is to expedite the training process as the number of parameters of FashionCLIP is very large. FashionCLIP generates a vector of dimension 512. The next components are three linear layers with slightly different architectures. The first consists of a linear layer, ReLU activation function, and Batch Normalization. The second consists of a linear layer and ReLU activation function. Finally, the third consists of only one linear layer.

The second block within the feature extraction component is the *Text Encoder Block*, and this generates an embedding for the text description of an item. The architecture is similar to that of the Image Encoder Block, but SentenceBERT [9] is used instead of FashionCLIP. Our model uses the pre-trained multilingual variant "distiluse-base-multilingual-cased-v2" of SentenceBERT that generates a vector of dimension 512. FashionCLIP can generate embedding for both text and image, but the FashionCLIP embeddings of an image and its text description are expected to be close to each other in the latent space. That is why a different model is used to generate the text embedding so that no redundant information will be used. Similar to FashionCLIP in the *Image Encoder Block*, the weights of SentenceBERT are frozen to accelerate model training.

Once the image and text embedding of each item in the input outfit set have been generated, Slay-Net constructs the outfit set feature matrix $E_{set} \in \mathbb{R}^{n_o \times d_o}$. The outfit set feature matrix is formally defined as:

$$E_{set} = \{E_{img,i} || E_{text,i}\}_{i=1}^{n_o} \tag{3}$$

where || is concatenation operation, n_o the number of items in the input outfit and d_o the sum of d_{img} and d_{text}.

4.3 Modeling Outfit as Set-structured Data

In prior works ([13], [12], [7]) where contrastive learning is used to perform outfit compatibility learning, the learning is done in a pairwise manner. This work aims to perform the contrastive learning in a set-wise manner. Each contrastive learning dataset instance consists of an anchor, a positive and negative. The goal of contrastive learning is to learn model weights such that the embeddings of anchor and positive are close to each other while making sure that the embeddings of anchor and negative are further away from each other. In a pairwise learning setup, the anchor and positive are two items that co-occur in an outfit. On the other hand, in a set-wise learning setup, the anchor is a partially complete outfit with one item taken out and the positive is the item taken out.

Prior works, such as [10] and [15], adopt a set-wise learning in outfit compatibility learning. In their works, they introduce a trainable target item token x_{target} in the set encoding step. The features of the elements of the input set and x_{target} are concatenated and passed into set encoding block, which is based on attention mechanism. The encoded x_{target} is then used as the fixed-dimensional set embedding. However, Slay-Net uses both set encoding and set pooling steps to generate the fixed-dimensional set embedding. The set encoding step is based on the Set Attention Block (SAB) [6] and the set pooling step is based on Pooling by Sliced-Wasserstein Embedding (PWSE) [8].

4.3.1 Set Encoding. The set encoding used in Slay-Net is permutation-equivariant, and this means that if the elements in the input set are shuffled, the outputs are shuffled correspondingly in the same way. SAB [6] is used as the building block of the set encoding in Slay-Net. SAB is an attention-based parameterized function that can capture the interaction between elements within an input set. The input to the set encoding is the outfit set feature matrix E_{set}. Given input E_{set}, SAB is formally defined as:

$$SAB(E_{set}) = LayerNorm(H + rFF(H)) \tag{4}$$

where H is defined as:

$$H = LayerNorm(E_{set} + Multihead(E_{set}, E_{set}, E_{set}; w_{SAB})) \tag{5}$$

The rFF used in the above equation is any row-wise feed forward layer, and $LayerNorm$ is layer normalization mechanism as per [1]. $Multihead$ is multi-head attention mechanism introduced in [14], and w_{SAB} is the set of parameters correspond to it. The output of set encoding step is the encoded outfit set feature matrix $E_{hidden} \in \mathbb{R}^{n_o \times d_{hidden}}$, and this will be the input of the set pooling step.

4.3.2 Set Pooling. Given E_{hidden}, the set pooling step generates a fixed-dimensional set embedding $x_o \in \mathbb{R}^{d_o}$. PSWE [8] is used in the set pooling step. PSWE treats the elements from the input set as samples from a probability distribution. Hence, using this modeling, comparing two sets is analogous to comparing two probability distribution. Furthermore, this allows the use of concepts and tools from Optimal Transport (OT) for comparing two sets. These concepts and tools are normally used to compare two probability distributions. In PWSE, generalized sliced Wasserstein distance [5] is used to define distance measure between two sets. PSWE embeds an input set into a Euclidian space, in which the weighted Euclidian distance of two sets is equal to the generalized sliced Wasserstein distance. Once PSWE has finished the embedding process, the generated fixed-dimensional set embedding x_o is passed to the learning heads.

4.4 Learning Heads

Slay-Net consists of two learning head; binary classification and contrastive learning head. A curriculum learning is used to train Slay-Net. The binary classification head will only be used in the first phase of the curriculum learning, while contrastive learning head is used in the first and second phase of the curriculum learning.

4.4.1 Binary Classification Head. The Polyvore outfits dataset [13] contains a dataset for Outfit Compatibility Prediction task. Outfit Compatibility Prediction task is one in which a model has to predict whether a set of items is a compatible outfit. The binary classification head performs a fine-tuning of the weights of Slay-Net through backpropagation with respect to Binary Cross Entropy loss L_{BCE}. The L_{BCE} is defined as:

$$L_{BCE} = \sum_{i=1}^{N_{bc}} y_i \log \sigma(f_{bc,i}) + (i - y_i) \log(1 - \sigma(f_{bc,i})) \tag{6}$$

where $f_{bc,i}$ is defined as:

$$f_{bc,i} = Linear_{bc}(x_{o,i}; \theta_{bc}) \tag{7}$$

$Linear_{bc}(.;)$ is a set of linear layers in binary classification head, $x_{o,i}$ the set embedding of instance i, y_i the ground truth for instance i and N_{bc} the number of instances in the Outfit Compatibility Prediction dataset.

4.4.2 Contrastive Learning Head. The Contrastive learning head is the main learning head as it enable the inference process for the Fashion FITB and Outfit CIR tasks. The embedding of the anchor x_o, positive x_p and negative x_n pass through a modified version of the Category-based Subspace Attention Network (CSA-Net) [7]. Afterwards, the CSA embedding of the anchor f_o, positive f_p and negative f_n are generated. The generation of CSA embedding consists of two steps; first, k subspace embeddings are generated, and then the weighted sum of the subspace embeddings are computed. The CSA embedding is formally defined as:

$$f = \sum_{i=1}^{k} w_i (x_{input} \odot m_i) \tag{8}$$

where m_i is the mask for subspace i, $w = \{w_i\}_{i=1}^{k}$ the weight of subspace embeddings and $x_{input} \in \{x_o, x_p, x_n\}$. In addition, w is a function of the one hot encoding of the general item type of the positive.

Set-wise Outfit Ranking Loss [10] is used in the contrastive learning head. The use of Set-wise Outfit Ranking Loss aims to optimize the Euclidean distance of f_o to f_p and f_o to f_n, such that f_o is relatively closer to f_p and further away from f_n. There are two variants of it; the mean $L_{SOR,mean}$ and hard $L_{SOR,hard}$ Set-wise Outfit Ranking Loss. $L_{SOR,mean}$ is used in the first phase of curriculum learning while the sum of $L_{SOR,mean}$ and $L_{SOR,hard}$ is used in the second phase of the curriculum learning. These losses are defined as follows;

$$L_{SOR,mean} = \frac{1}{|F_n|} \sum_{j=1}^{|F_n|} [d(f_o, f_p) - d(f_o, f_{n,j}) + m]_+ \tag{9}$$

$$L_{SOR,hard} = [d(f_o, f_p) - \min_{j \in \{1,\dots |F_n|\}} d(f_o, f_{n,j}) + m]_+ \tag{10}$$

where $[.]_+$ is a hinge loss, m the margin , $F_n = \{f_{n,j}\}_{j=1}^{n_{negative}}$ and $n_{negative}$ the number of negatives sampled in each instance. Furthermore, $d(x_a, x_b)$ denotes the Euclidian distance between the vector x_a and x_b. In this work, hyperparameter optimization is conducted to determine the optimal value of the margin m for each dataset. The optimization process involves training the model across a range of margin m values and recording the performance using the FITB accuracy. The value of the margin m that achieves the highest FITB accuracy is selected as the optimal value for the corresponding dataset.

4.5 Curriculum Learning

Slay-Net is trained using a curriculum learning framework, which consists of two phases. The first phases involves training the model in a multi-task learning setup, where both the binary classification and contrastive learning heads are used. In the multi-task learning setup, Slay-Net is exposed to the binary classification and contrastive learning dataset simultaneously. The aim of this is to allow Slay-Net to capture patterns which can only be discovered through simultaneous exposure to binary classification and contrastive learning dataset. The loss used in the first phase of the curriculum learning L_{phase1} is as follows:

$$L_{phase1} = \lambda_{cl} L_{SOR,mean} + \lambda_{bc} L_{BCE} \tag{11}$$

where λ_{cl} and λ_{bc} are the weight for the contrastive learning loss and binary classification loss, respectively. Furthermore, in the first phase, negatives with the same general item type as the positive are randomly sampled for each contrastive learning dataset instance.

The second phase of the curriculum learning focuses on the contrastive learning, and the loss used L_{phase2} is as follows:

$$L_{phase2} = L_{SOR,mean} + L_{SOR,hard} \tag{12}$$

The strategy adopted to sample negatives is different in the second phase. During the second phase, negatives with the same fine-grained item type as the positive are randomly sampled. This makes the second phase contrastive learning more difficult because items of the same fine-grained type share more similarities and Slay-Net has to capture the more nuanced outfit compatibility patterns.

5 EXPERIMENT DETAILS

5.1 Dataset

Polyvore outfits dataset [13] is used in this work, and dataset consists of 68,306 outfits and 365,054 unique items retrieved from Polyvore, a now-defunct fashion social media website. This work uses both the disjoint and nondisjoint variant of the dataset. The difference between two variants is whether they allow an item to be in different dataset splits. In disjoint variant, each item can only be either in train, validation or test split, while nondisjoint variant does not have this restriction.

In the dataset, there are two hierarchies of item types; general and fine-grained item type. The general item type is labeled using a word that describes the type, such as "shoes" and "tops". However, the fine-grained type is only labeled with a number. Figure 6 illustrate the difference between general and fine-grained item types. The top half shows the case where the reference item is a pair of strappy heels. In this case, the general item type is shoes and the fine-grained item type is strappy heels. For each general and fine-grained item types, examples are given to illustrate the difference between the two item types. The bottom half shows the case where reference item is a pair of long jeans.

Table 1: Slay-Net performance comparison with prior works. Results marked with $*$ symbol are taken from [15].

Work	Disjoint				Nondisjoint			
	FITB	**R@10**	**R@30**	**R@50**	**FITB**	**R@10**	**R@30**	**R@50**
(Vasileva et al., 2018) [13]	55.65*	3.66*	8.26*	11.98*	57.83*	3.50*	8.56*	12.66*
(Tan et al., 2019) [12]	53.67*	4.41*	9.85*	13.87*	59.07*	5.10*	11.20*	15.93*
(Lin et al., 2020) [7]	59.26*	5.93*	12.31*	17.85*	63.73*	8.27*	15.67*	20.91*
(Sarkar et al., 2022) [10]	59.48*	6.53*	12.12*	16.64*	67.10*	9.58*	17.96*	21.98*
(Wang and Zhong, 2023) [15]	**63.04***	6.10*	13.24*	18.81*	65.32*	6.81*	14.46*	20.38*
Slay-Net (Ours)	61.36	**6.85**	**14.23**	**19.52**	**68.53**	**11.11**	**21.59**	**27.95**
Improvement over best published baseline (%)	-2.66	4.90	7.48	3.77	2.13	15.97	20.21	27.16

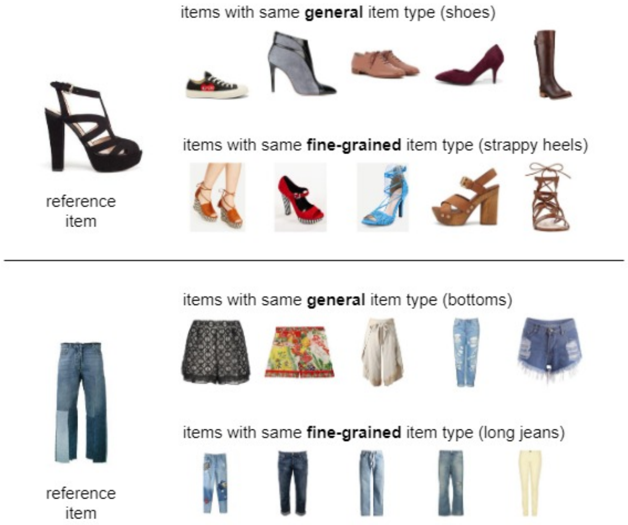
items with same **general** item type (shoes)
items with same **fine-grained** item type (strappy heels)
reference item
items with same **general** item type (bottoms)
items with same **fine-grained** item type (long jeans)
reference item

Figure 6: Illustration of general item type and fine-grained item type. *top*: the general and fine-grained item type of the reference item is shoes and strappy heels, respectively. *bottom*: the general and fine-grained item type of the reference item is bottoms and long jeans, respectively.

5.2 Evaluation

This works focuses on two tasks: Fashion FITB and Outfit CIR. The performance on Fashion FITB is measured by FITB accuracy and the relevant dataset from [13] is used to calculate the metric. The Outfit CIR task is measured using Recall@top-k, but unlike Fashion FITB, the dataset to calculate the metric is not readily available in [13]. Data pre-processing as per [7] needs to be performed, where 3000 items of the same fine-grained item type as the correct answer to the FITB question are to be selected from training and test splits. In [7], [10] and [15], Recall@top-k for three different values of k (10, 30 and 50) are calculated, and this work will calculate the same variants of Recall@top-k, too.

5.3 Implementation Details

The main loss being minimized in Slay-Net is Set-wise Outfit Ranking Loss [10]. When calculating the the Set-wise Outfit Ranking loss, CSA embedding [7] need to be generated and the number of subspace embeddings k is set to 5, the same value as in the original work [7]. The hinge loss margin used are different in the disjoint and nondisjoint dataset; 1.35 is used for disjoint dataset and 0.45 is used for nondisjoint dataset. The set encoding of Slay-Net consists of 2 layers of SAB with 4 heads in each layer. In the set pooling, the number of elements in PSWE reference set is 19 for nondisjoint dataset and 16 for disjoint dataset.

During the first phase of the curriculum learning, the weight of the contrastive learning loss λ_{cl} is set as 0.8 while the weight of the binary classification loss λ_{bc} is set as 0.2. The first phase of the curriculum learning is run for 20 epochs while the second phase is run for 50 epochs. In the second phase, the weights of Slay-Net are initialized using the weights of the best model of the first phase. At the end of each training epoch, the FITB accuracy is calculated using the validation split dataset. The model of the epoch, in which the validation FITB accuracy is the highest, is chosen as the best model. The training is carried out with batch size of 512 and Adam optimizer with learning rate of 5e-5.

6 RESULTS

Table 1 shows the performance of Slay-Net and relevant prior works in Fashion FITB and Outfit CIR tasks. The performance in Fashion FITB task is measured using FITB accuracy, while that of Outfit CIR task is measured using Recall@top-k (on Table 1, it is abbreviated as R@k). Each metric figure of Slay-Net is an average of six figures from six training runs, where each run uses different random seed. In [15], two variants of the model is proposed; one trained with the outfit textual description and another without. Since Slay-Net does not use the outfit textual description for training and inference, the performance figures of [15] on Table 1 are the ones for the variant where outfit textual description is not used in training and inference.

Slay-Net outperforms the relevant prior works in all datasets and tasks, except the Fashion FITB task in disjoint dataset. [15] achieves the best performance in the Fashion FITB task in disjoint dataset. Although the complete implementation of [15] is not yet made publicly available at the time of writing, it can be inferred that the model proposed by [15] has smaller number of parameters as compared to Slay-Net. In the Polyvore outfits dataset [13], there are 53,306 outfits for training in the nondisjoint dataset, and 16,995 in the disjoint dataset. The number of outfits available for training in the disjoint dataset is relatively smaller. One potential explanation of the observation in the Fashion FITB performance in disjoint dataset is that the model with fewer parameters could perform better when the available training data is relatively smaller.

Slay-Net outperforms relevant prior works in Fashion FITB task in nondisjoint dataset and in Outfit CIR task in disjoint and nondisjoint datasets. This observation could be credited to the research contributions introduced in this work. In [10] and [15], only set encoding is used to generate the fixed-dimensional set embedding, but Slay-Net uses both set encoding and set pooling. The use of set pooling enables the exploitation of all encoded features of items in the input outfit set to generate the set embedding. Furthermore, when training Slay-Net, the multi-task learning in the first phase of the curriculum learning allows Slay-Net to capture patterns that can only be discovered through simultaneous exposure to the binary classification and contrastive learning dataset. This, in turn, allows Slay-Net to start the second phase of the curriculum learning with the model parameter weights that enable it to achieve better performance in Fashion FITB and Outfit CIR task by the end of the training process.

7 CONCLUSION

This work presents the model Slay-Net that leverages both set encoding and set pooling to learn to generate the embedding of the set-structured input in the Outfit Compatibility Learning. Furthermore, Slay-Net is trained using an innovative curriculum learning approach that involves simultaneous training of binary classification and contrastive learning head via multi-task learning. This work focuses on two tasks; Fashion FITB and Outfit CIR tasks. Slay-Net outperforms the state-of-the-art approaches in the Outfit CIR task as measured by Recall@top-k.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[2] Patrick John Chia, Giuseppe Attanasio, Federico Bianchi, Silvia Terragni, Ana Rita Magalhães, Diogo Goncalves, Ciro Greco, and Jacopo Tagliabue. 2022. Contrastive language and vision learning of general fashion concepts. *Scientific Reports* 12, 1 (2022), 18958.

[3] Guillem Cucurull, Perouz Taslakian, and David Vazquez. 2019. Context-aware visual compatibility prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 12617–12626.

[4] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. 2017. Learning fashion compatibility with bidirectional lstms. In *Proceedings of the 25th ACM international conference on Multimedia.* 1078–1086.

[5] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. 2019. Generalized sliced wasserstein distances. *Advances in neural information processing systems* 32 (2019).

[6] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning.* PMLR, 3744–3753.

[7] Yen-Liang Lin, Son Tran, and Larry S Davis. 2020. Fashion outfit complementary item retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 3311–3319.

[8] Navid Naderializadeh, Joseph F Comer, Reed Andrews, Heiko Hoffmann, and Soheil Kolouri. 2021. Pooling by sliced-Wasserstein embedding. *Advances in Neural Information Processing Systems* 34 (2021), 3389–3400.

[9] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[10] Rohan Sarkar, Navaneeth Bodla, Mariya Vasileva, Yen-Liang Lin, Anurag Beniwal, Alan Lu, and Gerard Medioni. 2022. Outfittransformer: Outfit representations for fashion recommendation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition.* 2263–2267.

[11] Anirudh Singhal, Ayush Chopra, Kumar Ayush, Utkarsh Patel Govind, and Balaji Krishnamurthy. 2020. Towards a unified framework for visual compatibility prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision.* 3607–3616.

[12] Reuben Tan, Mariya I Vasileva, Kate Saenko, and Bryan A Plummer. 2019. Learning similarity conditions without explicit supervision. In *Proceedings of the IEEE/CVF international conference on computer vision.* 10373–10382.

[13] Mariya I Vasileva, Bryan A Plummer, Krishna Dusad, Shreya Rajpal, Ranjitha Kumar, and David Forsyth. 2018. Learning type-aware embeddings for fashion compatibility. In *Proceedings of the European conference on computer vision (ECCV).* 390–405.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[15] Xin Wang and Yueqi Zhong. 2023. Text-Conditioned Outfit Recommendation With Hybrid Attention Layer. *IEEE Access* (2023).

[16] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. *Advances in neural information processing systems* 30 (2017).

[17] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. 2019. Fspool: Learning set representations with featurewise sort pooling. *arXiv preprint arXiv:1906.02795* (2019).