Outfit2Vec: Incorporating Clothing Hierarchical MetaData into Outfits' Recommendation

Shatha Jaradat
KTH Royal Institute of Technology
Stockholm, Sweden
shatha@kth.se

Nima Dokoohaki
KTH Royal Institute of Technology
Stockholm, Sweden
nimad@kth.se

Mihhail Matskin
KTH Royal Institute of Technology
Stockholm, Sweden
misha@kth.se

ABSTRACT

Fashion Personalisation is emerging as a major service that online retailers and brands are competing to provide. They aim to deliver more tailored recommendations to increase revenues and satisfy customers by providing them options of similar items according to their purchase history. However, many online retailers still struggle with turning customers' data into actionable and intelligent recommendations that reflect their personalised and preferred taste of style. On the other hand due to the ever increasing use of social media, fashion brands invest in influencers' marketing to advertise their brands to reach a larger segment of customers who strongly trust their influencers' choices. In this context the textual and visual analysis of social media can be used to extract semantic knowledge about customers' preferences that can be further applied in generating tailored online shopping recommendations. As style lies in the details of outfits, recommendation models should leverage the fashion metadata ranging from clothing categories and subcategories to attributes such as materials and patterns to overall style description in order to generate fine-grained recommendations. Recently, several recommendation algorithms suggested to model the latent representations of items and users with neural word embeddings approaches which showed improved results. Inspired by Paragraph Vector neural embeddings model, we present **Outfit2Vec** in which we leverage the complex relationship between user's fashion metadata while generating outfits embeddings. In this paper, we also describe a methodology to generate representative vectors of hierarchically-composed fashion outfits. We evaluate our model on an extensively-annotated Instagram dataset on recommendation and multi-class style classification tasks where our models achieve better results specially in whole outfits' ranking evaluations with an average of 22% increase.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*;

KEYWORDS

Neural Recommendation, Paragraph2Vec, Word2Vec, Clothing Metadata, User Behavior, Fashion, Style, Instagram, Personalisation

recsysXfashion'19, September 20, 2019, Copenhagen, Denmark

ACM ISBN .
https://doi.org/

ACM Reference format:
Shatha Jaradat, Nima Dokoohaki, and Mihhail Matskin. 2019. Outfit2Vec: Incorporating Clothing Hierarchical MetaData into Outfits' Recommendation. In *Proceedings of Workshop on Recommender Systems in Fashion, 13th ACM Conference on Recommender Systems, Copenhagen, Denmark, September 20, 2019 (recsysXfashion'19),* 8 pages.
https://doi.org/

1 INTRODUCTION

Personalisation is becoming a leading factor for the current and future success of E-commerce sector. According to a recent study [17], 43% of online purchases are influenced by personalised recommendations, 75% of customers prefer personalised brands messaging, 94% of retail companies are perceiving personalisation as a critical strategy for their success. With the E-commerce fashion industry's worldwide revenue predicted to rise to $712.9 billion by 2022 [17], it is expected that online retailers will direct more efforts into delivering the best perosnalisation services to online fashion shoppers. This is becoming more important with the increasing number of fashion brands and online retail competitors. Customers should experience more recommendation services tailored to their needs and fashion taste. This in turn could be reflected in increased revenue possibilities for retailers. Personalisation can also address one of the leading threats to online fashion which is the unsatisfied customers who frequently return items. Learning more about customers' style preferences can provide them with more satisfying options and possibly reduce the frequency of returns.

Social Media platforms and specifically Instagram is becoming a driving force in the fashion market [13]. Its real power is coming from the smart integration of the increasing popularity of fashion models and the attractive trends of fashion brands. Many followers don't view influencers as paid sales representatives, but rather real people who share their everyday experiences with them. As brands are increasingly investing using social media, influencer marketing has grown into a multibillion-dollar industry [13]. Brands believe that the benefit is beyond liking or sharing a single outfit's post of a fashionista in Instagram. Many followers browse the product and search for it or for a similar one in online shops. This enhances the fast growth of these brands. It is not just for the top influencers, some brands also target Micro-Influencers who have between 1K-10K followers. They believe that because they have a smaller number of followers, this in turn could help them to build more personal relationships with people. As a consequence, their engagement rates are higher, and they are a cost-effective solution [3]. With product tagging feature in Instagram, direct channels for buying can be facilitated. Tagging might be for one or some of the main items in the outfit. Other influencers advertise the product using Hashtags or description in the post and comments. Some

followers are curious about all the details in the outfit or the overall style of the fashionistas. This can be noticed in the comments of fashionistas' posts where many conversations start between followers about the outfit's details. Here text and image analysis and recommendation technologies could play a significant role in extracting semantic knowledge about followers' fashion preferences. Building fashion profiles for these users could enhance recommending them smarter options that are close to their real taste.

In the last couple of years, a promising new class of neural recommendation models that can generate user and item embeddings by modeling the nonlinearity in **Matrix Factorization** (MF) latent factor models has emerged and has shown improving results. **Word2Vec** model [15] has inspired multiple neural recommendation models such as Prod2Vec [5] and Item2Vec [1]. In the context of Word2Vec [15], the model is able to learn word representations in a low-dimensional continuous vector space using a surrounding context of the word in a sentence. Then as a result, semantically similar words will be close to each other in the resulting embedding space. Researchers have realized that just like word embeddings can be trained by treating a sequence of words in a sentence as context, the same can be done for training embeddings of user's actions (purchases for example) by treating such sequence of user actions as context [14]. This gives us inspiration that a training data that consists of user's outfits which consist of a combination from different clothing categories and subcategories along with their attributes (material/pattern), style and brands can be treated as sequences and learnt by the model. The vocabulary in the model can then be the sequence of outfit details rather than just clothing categories. However, we still deal here with the hierarchical nature of data which is a more complex scenario than the previously described scenarios in literature (where they just consider products or items). In this paper we describe a methodology to generate representative vectors of such hierarchically-composed items such as outfits. These representations are then used in our **Outfit2Vec** and **PartialOutfit2Vec** models in which we handle the complex relationship between fashion metadata in order to generate outfits embeddings. We evaluate our models on an extensively- annotated Instagram dataset on recommendation and multi-class style classification tasks.

2 RELATED WORKS

Some of the recent neural recommendation models have focused on user-to-item recommendations and others brought into focus the value of item-to-item recommendations to avoid problems such as cold start and sparsity of data. Neural Network-based Collaborative Filtering **(NCF)** [8] is an example of a model in which a neural architecture is used to replace the traditional inner product on the latent features of users and items in matrix factorization. In their architecture, they use a multi-layer perceptron (MLP) to learn the user-item interaction function. Their assumption is coming from the fact that the inner product in matrix factorization that simply combines the multiplications of latent features linearly, may not be sufficient to capture the complex structure of user interaction data. A very similar model to NCF is Neural Tensor Factorization **(NTF)** [9] with the addition of including a long-short term memory architecture to model the dynamicity in data that represent how the users' preferences change over time. **Content2Vec** [16] is a recent model where image and text embeddings are joined together for product recommendation.

Inspired by the famous word embedding model Word2Vec [15], **Item2Vec** was proposed in [1] as a neural item embedding to learn latent representations of items in collaborative filtering scenarios. The main goal of Item2Vec is to learn item-item relations even when the user's information is absent for the purpose of inferring items' similarity. Learning items' similarity is done by embedding items in a low-dimensional space. Another similar model is **Prod2vec** [5], which generates product embeddings from sequences of purchases and performs recommendation based on most similar products in the embedding space. Prod2vec embeds products into real-valued, low-dimensional vector space using a neural language model applied to a time series of user purchases. As a result, products with similar contexts (their surrounding purchases) are mapped to vectors that are nearby in the embedding space. To be able to make meaningful and diverse suggestions about the next product to be purchased, they further cluster the product vectors and model transition probabilities between clusters to use the closest products in the embedding space. **Meta-Prod2Vec** [18] enhances Prod2Vec by injecting the items' metadata (items' attributes and users' past interactions) as side information while computing the low-dimensional embeddings of items. The assumption behind their model is that the item representations that merges the items metadata can lead to better performance on recommendation tasks. **Search2Vec** [4] is one example where multiple embeddings are considered in the learning process. In Search2Vec, the authors have addressed the idea of learning low-dimensional representations of "different" inputs in a common vector space. In their model, search queries and sponsored advertisements are examined together to decide the best matching and/or relevant advertisements to be presented to the user while performing search.

An increasing number of research papers started to focus on applying embeddings approaches for learning outfits' compatibility and similarity for the purpose of recommendation. In [7], Bidirectional LSTMs are used to train sequences of outfits to predict the next item within an outfit that is described from top to bottom. Multimodal representations of images and their descriptions are learnt to enhance the training of the model by minimising the loss between the visual and semantic information to learn the compatibility of items. In [2], multimodal representations are learnt from fashion images and their text, and then fed to a neural network that combines the embeddings for all items within an outfit and outputs a score of compatibility based on the relationship between the embeddings of the different items. All this work depends on the idea that compatible items will be closer to each other in the embedding space. In [19] they use Siamese CNNs to learn features of images, and then with Nearest Neighbour retrievals compatible items can be generated. Attenion-based techniques were applied in [11] where their primary goal is to complete an outfit based on the context which is the scene in the image (e.g. sea).

3 METHODOLOGY

It can be noticed that all the previously described neural recommendation models have focused on one type of inputs such as "product" or "item" for finding low-dimensional word representations. However, considering more than one type of input is a more challenging task. In our work, we have sequences of outfits' descriptions such that each outfit is composed of multiple clothing items. Furthermore, each item consists of clothing category, subcategory, with materials and patterns details. Style labels are attached to the whole outfit. Moreover, brand names and hashtags are also available at the outfit's level and they can be used as additional contexts to enhance the outfits' recommendations. We care about having the materials and patterns details describing the items at the instance level rather than the whole sequence. This becomes more important while generating partial recommendations, as the recommendation should describe an instance of clothing including the materials and patterns, and should not be just the category of the clothing item or the material as an example. The clothing metadata was generated from our framework described in [6] by analysing the text and comments of Instagram fashion images using an unsupervised embedding approach. We followed a fashion taxonomy that we defined in a similar way as Zalando online shop[1].

A more detailed description of the problem is given as follows: given a set of outfit choices obtained from a defined number of users, we define the following components of each outfit:

- List of clothing categories that exist in the outfit. For example: an outfit that consists of a dress, a coat, shoes and a bag.
- List of clothing subcategories as an additional level of details. For example: the dress is a cocktail dress, the coat is a trench coat, the shoes is a Stiletto heel pumps and the bag is a handbag.
- List of materials from which the clothing items are made. For example: the dress is made from lace as a main component, the coat is made from polyester, the pumps and the handbag are made from leather.
- List of patterns that describe the clothing items. For example: the dress is floral, the coat is plain, the pumps is plain, and the handbag is camouflage.
- List of styles that describe the overall outfit. For example: classic chic style.
- List of brands of the clothing items. For example: the dress and the coat are from Zara, the bag is from Gucci and the pump is from Tamaris.

Our objectives is to find a D-dimensional representation $v_o \in R^D$ of each outfit o such that similar outfits based on their vectors' similarity can be recommended to the user. To address the challenges of dealing with the hierarchical levels in each outfit's item description, we propose a methodology for generating outfits' representative vectors in the coming subsection. Then, we describe our outfit2vec model.

[1] https://zalando.com

3.1 Methodology for Generating Outfits' Representative Vectors

In this section we describe a methodology that can be followed to generate representative vectors of hierarchically-composed items such as outfits. As each outfit is composed of multiple clothing categories, and each of which has different attributes such as materials and patterns, a strategy of projecting these details into vectors representing the whole outfit should be decided. The strategy described here is a combination of rule-based and embedding-based approaches. The main steps in our methodology are defined as follows:

- Mapping of item details into clothing entities
- Projecting the entities into outfit vectors

3.1.1 Mapping of Item Details into Clothing Entities. As each outfit has a number of clothing categories, subcategories, materials, and patterns, the first step is to map the clothing items' details to instances describing the whole item hierarchically based on the fashion taxonomy that we follow. For example: a Skinny Jeans in an outfit (Jeans is the clothing category and Skinny is the subcategory) with its material's label (Denim) and pattern's label (Plain) should be mapped to a single clothing entity. A semantic description of the clothing item within an outfit would not just include the clothing category but rather the attributes of materials and patterns. This in turn makes the item unique and differentiated from other Jeans instances. These entity instances are then converted to the vector space using embedding-based approach. We experimented with different candidate vectors describing clothing items within an outfit. Candidate vectors vary based on the details they contain such as: clothing categories, subcategories, materials, patterns, styles, brands, and hashtags. They also vary based on the arrangement of clothing details. For example, we experimented with sequences of outfits with clothing categories, followed by the subcategories, then followed by the attributes. Another candidate vector describes the outfit by having the structure: pattern material subcategory category for each item. In other candidate vectors we tried adding the brands and hashtags information in addition to the clothing details in the same sequence. The representative vector that we chose for each clothing entity was mapped from the following structure: *pattern-material-subcategory-category*. We found that combining the clothing details into one word which later results into the model projecting the combined words into a single word vector had the best results in whole outfits prediction. We have also experimented building entities from the structure *pattern material subcategory category* by considering each word individually, and changing the prediction to be for a group of words from the outfit's sequence rather than single word in the partial outfits prediction. The **objective from the arrangement** (pattern material subcategory category) is to facilitate having partial recommendations as we split sequences of outfits and generate recommendations based on some parts of the clothing entities. So, for the predictions they result as four words each from different type (pattern material subcategory category) and describing a single clothing entity. **Figure** 1 shows an illustrative example of the procedure of combining the metadata of each clothing item to form an entity. In the experiments section, we show the results of experimenting with different

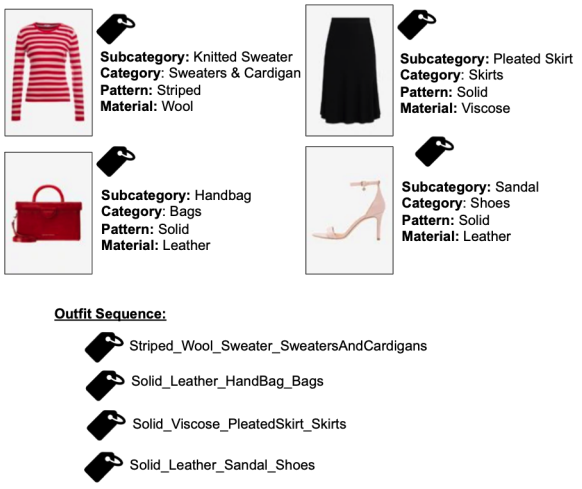

Figure 1: An illustration of combining clothing metadata into clothing entity descriptions

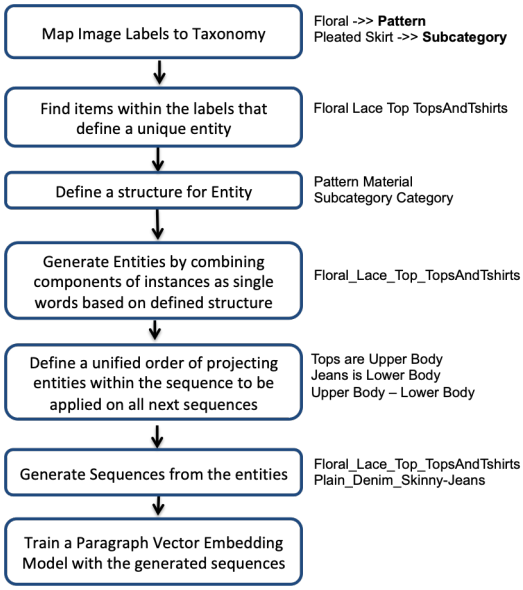

Figure 2: General process to follow to generate embeddings from Hierarchically-Composed Clothing Instances

candidate vectors while deciding the structure of the representative vector.

3.1.2 Projecting Clothing Entities into Outfit Vectors. As mentioned in the previous subsection, multiple candidate vectors were calculated for each clothing entity to decide the structure of the representative vectors. Then a unified rule-based approach was followed to decide some order of the entities to be projected as outfits' sequences. The order was followed to provide a consistent way of describing outfits. Depending on the assumption that we usually describe an outfit starting from the Jacket and dress and then the shoes and the bags, we have defined some rules which could be stated as the following: (1) *Add Jacket or Coat Entity if Exists* (2) *If Upper Body and Lower Body Exists:* a. *Add Upper Body Entity*, b. *Add Lower Body Entity*, (3) *If Upper Body doesn't Exist and a Dress Exists: Add Dress Entity*, (4) *Add Tights and Socks Entity if Exists*, (5) *Add Shoes Entity if Exists*, (6) *Add Bags Entity if Exists*, (7) *Add Accessories Entity if Exists*. Upper body entities consist of the following categories: (1) *Blouses and Tunics*, (2)*Tops and TShirts*, (3) *Jumpers and Cardigans*. Lower body categories include: (1) *Skirts*, (2) *Jeans*, (3) *Trousers and Shorts*. The procedure can be generalised as we illustrate in **Figure** 2 to other hierarchically-composed structures. As shown in the figure, the first step is to map the raw text to a defined taxonomy from which instances can be defined. A structure of the instance's description should be decided as in our case it was: pattern-material-subcategory-category. Entities are generated by combining components of instances to be projected as single words to the model. A unified order of the sequences can provide a consistent way of describing the predictions. Finally, the generated sequences are used to train the model.

3.2 Outfit2Vec and PartialOutfit2Vec Models

To address the challenges of dealing with the hierarchical levels in each outfit's item description and motivated by the success of distributed language models such as Paragraph Vector (Paragraph2Vec) [12], we present Outfit2Vec and PartialOutfit2Vec. Paragraph2Vec which is an unsupervised model for constructing representations of input sequences of variable length such as paragraphs has two models. In the distributed memory model of Paragrpah Vector (PV-DM), every word is mapped to a unique vector, and every paragraph is also mapped to a unique vector to act as a memory that remembers what is missing from the context of the paragraph. The paragraph vector is then concatenated with several word vectors from a paragraph to predict the following word in the given context. This model addresses the weaknesses of bag-of-words models as it considers the order of words which is important in some tasks. We propose an outfit embedding model that learns user's outfits representations using the PV-DM model but rather than providing the outfit details as separate words to the model, we apply the described methodology in the previous section to present clothing entities as words to the model, and the sequence of clothing entities as paragraphs. In this case, a clothing entity denoted as **i** with the structure [category-subcategory-material-pattern] is mapped to a unique word vector rather than providing the separate parts of the entity as words. The objective of our model is to evaluate the effect of projecting each clothing entity as a single word vector on the ability of the model to provide related fine-grained recommendations at the whole and partial outfit levels. We train our model for two purposes: (a) performing a prediction of a clothing entity within a sequence of an outfit which we refer to as **(PartialOutfit2Vec)**, and (b) performing a prediction of a whole new outfit to the user which we refer to as **(Outfit2Vec)**. For the first purpose, the task is to find the most similar clothing entity to complete a partial sequence of an outfit. The second task is to find the most similar outfits in the training set from the inferred outfit's vector

A formal description is as follows: given a sequence of outfits o_1, o_2, o_3, ... o_N, (N is the total number of outfits), where each outfit o_n is composed of a sequence of clothing entities $i_1, i_2, i_3, ... i_T$, (T is the

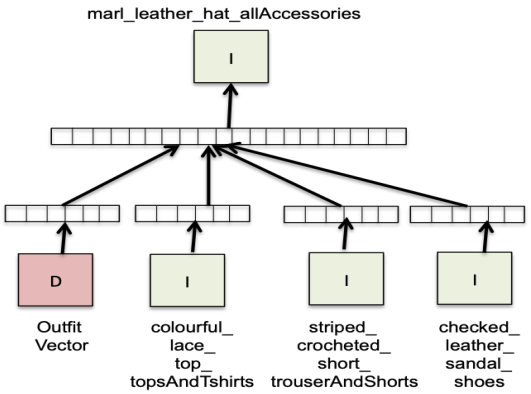

Figure 3: An illustration of learning outfit vector Outfit2Vec(PV-DM). The clothing entities are presented as words and the sequence of clothing entities form the outfit's vector.

total number of entities), the objective of the **PartialOutfit2Vec** model is to maximize the average log probability of predicting the entity i_t given its context of entities within the outfit and is defined as follows:

$$\frac{1}{T}\sum_{t=k}^{T-k} logp(i_t|i_{t-k},...i_{t+k}) \tag{1}$$

The prediction task is obtained through a softmax function over vocabulary I which is extracted from the clothing entities:

$$P(i_t|i_{t-k},...i_{t+k}) = \frac{e^{y_{it}}}{\sum_i e^{y_n}} \tag{2}$$

Where y is computed as follows: $y = b + Uh(i_{t-k},...,i_{t+k};I)$ Where U,b are the softmax parameters, h is constructed by the concatenation of both clothing entities and outfit paragraph vectors. For larger scale classifications, hierarchical softmax can be used. For **Outfit2Vec** the prediction task becomes at the level of a whole outfit sequence o_t rather than the clothing entity i_t.

The model can be tweaked by providing tags to each paragraph, where the tag vector can be concatenated along with the paragraph and word vectors. In our model, we provide the outfit details as tags while training the sequences. **Figure** 3 illustrates learning outfit and clothing entities vectors framework.
In PV-DM model, the total number of parameters is computed as: N * p + M * q where N is the total number of outfit sequences in a given corpus, M is the number of clothing entities in the vocabulary, each outfit is mapped to p dimensions and each clothing entity is mapped to q dimensions. As the number of parameters can get larger in a larger scale coprus, we have also experimented using Paragaraph Vectors Distributed Bag of Words model (PV-DBOW) where the model is supposed to store less data as it doesn't store the word (clothing entities) vectors. So, the context of words in the input is ignored, and the prediction is performed from samples of the paragraphs. A detailed comparison of the behavior of our models in both ways (Outfit2Vec-(PV-DM), Outfit2Vec-(PV-DBOW)) is shown in the experiments section.

4 EXPERIMENTAL PIPELINE

In this section, we present our evaluation experiments on top-k recommendations for whole and partial outfits. Then, we present our evaluation on multi-class style classifications. A detailed description of our dataset is also provided.

4.1 Datasets

The experiments were performed on an extensively-annotated Instagram dataset that we gathered in our work [10] from a community of fashion models (RewardStyle) [2]. The data is in the form of images, image captions, user comments, and hashtags associated with each post. We applied the text analysis procedure which was described in [6]. This resulted in annotating each image with a list of categories, sub-categories, materials, patterns and styles with percentages according to the importance. We employed a subset consisting of around **50,000** posts from the collected Instagram dataset, each with the described annotations. Some statistics about the dataset are as follows: the number of unique outfits is 14,573, the number of clothing entities is 72,479, and the number of unique clothing entities is 4790. The number of outfits vary per user as it is based on the amount of posts collected from each user.

4.2 Whole Outfits Recommendation (Outfit2Vec)

In our experiments, the models are trained with sequences of whole outfits with 80% split for training and 20% for testing. An example of an outfit sequence is "colourful lace top topsAndTshirts - striped crocheted short trouserAndShorts -checked leather sandal shoes". The difference between whole and partial outfits' evaluation is in the input provided to the model during testing. In whole outfits experiments, a complete sequence is provided to the model for inference of its representation. We consider the next sequence in this case to the be the ground truth as our objective is to evaluate the model's ability to predict the next outfit based on the history of outfits in the corpus. The most similar top k paragraph vectors to the inferred one are then retrieved. To evaluate the top k results, we use the following measures: Normalized Discounted Cumulative Gain (NDCG), Mean Reciporcal Rank (MRR) and Mean Average Precision (MAP). We compare the model in two cases: by modeling the clothing entities as single words *Outfit2Vec(PV-DM)-SE* and by providing the clothing details as separate words in the outfit's sequence *Outfit2Vec(PV-DM)-SW*. SE stands for *Structured Entities* and SW stands for *Structured Words*. An example of a clothing entity within a structured entity sequence is: floral-lace-top-topsAndTshirts (one word), and an example of a clothing entity within a structured word sequence is: floral lace top topsAndTshirts. So, the difference between SE and SW models are in projecting the details into single or separate words but both of them reflect the same structure. The same approach is applied for Outfit2Vec (PV-DBOW) models where we donate the models under evaluation as *Outfit2Vec(PV-DBOW)-SE* and *Outfit2Vec(PV-DBOW)-SW*. We also run our experiments on randomly arranged sequences which we denote here as PV-DBOW-Random and PV-DM-Random. We experimented different values for the vector size used for ParagraphVector training, and

[2] https://www.rewardstyle.com/

we selected 200. We train the models under evaluation for 30 epochs.

A ranking measures that we chose in our evaluation is Normalized Discounted Cumulative Gain (NDCG) which is one of the mostly used measures of effectiveness in information retrieval algorithms. The usefulness of a retrieved document is decided using a graded relevance scale based on the document's position in the result list. The gain of the result list is accumulated from the top to the bottom of the list with the gain of each result discounted at lower ranks. With that, higher relevant documents are more useful when appearing earlier in a search engine result list (having higher ranks). This is a very relevant measure for the outfits' prediction evaluation as the users expect to find the most relevant suggestions at the top of the recommendations list. For a prediction, the normalised discounted cumulative gain nDCG is computed as: $nDCG_p = \frac{DCG_p}{IDCG_p}$ where DCG_p is the discounted cumulative gain and $IDCG_p$ is the ideal DCG which reflects the maximum possible DCG through the position p. They are calculated using the following formulas respectively: $DCG_p = \sum_{i=1}^{p} \frac{rel_i}{log_2(i+1)}, IDCG_p = \sum_{i=1}^{|Rel_p|} \frac{2^{rel_i}-1}{log_2(i+1)}$ where rel_i is the graded relevance of the result at position i. In our context i is the relevance of the prediction result. In **Table** 1 we show the average NDCG for all retrieved results for k=30 and 40 where k is the number of retrieved predictions. We chose the values of k based on the assumption that it is a suitable number of retrieved recommendations for an online shopper to browse from. We have also computed the results for k=10,20 and k=50 which is shown in **Figure** 4. As can be seen from the statistics in **Table** 1, *Outfit2Vec(PV-DM)-SE* outperformed *Outfit2Vec(PV-DM)-SW* with an average of increase of 19% for the measured k values. For Outfit2Vec(PV-DBOW) models, it can be also seen that using the SE methodology resulted in an average increase of 25%. The average increase of *Outfit2Vec(PV-DBOW)-SE* when compared to *Outfit2Vec(PV-DBOW)-SW* was higher than the average increase of *Outfit2Vec(PV-DM)-SE* compared to *Outfit2Vec(PV-DM)-SW*. This can be explained by the ignorance of order in the PV-DBOW models. That's why the introduced structured approach achieves more significant improvement in PV-DBOW model when the details are projected as single words. Both models have outperformed *PV-DBOW-Random* and *PV-DM-Random* which are composed by random sequences of outfits without applying our methodology. **Figure** 4 shows a graphical comparison of the increase in NDCG values for the models under evaluation.

Another statistical measure that we compute is the Mean Reciprocal Rank (MRR). The mean reciprocal rank is the average of the reciprocal ranks of outfits retrievals for a sample of predictions Q and is computed as follows: $MRR = \frac{1}{Q} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$ Where $rank_i$ refers to the rank position of the first relevant document for the i_{th} query. The sequences that we handle are composed of clothing categories, subcategories, materials and patterns. To calculate the matching between the ground truth and the predictions, we calculate if all the items within the goal are available in the prediction sequence to be counted as a correct result. As can be seen in **Table** 1, for *Outfit2Vec-SE* and *Outfit2Vec-SW*, the models performed similarily in MRR. However, a positive increase was noticed for *PV-DBOW-SW* when compared to *PV-DBOW-SE*.

Mean Average Precision (MAP) is a classical evaluation metric in binary classification algorithms and information retrieval tasks. We are interested in calculating the precision at top k results. But since precision is a binary metric used to evaluate the models with binary output and in our case we have a multiclass output. Thus, we need to translate our output to a binary output (relevant and not relevant outfits). To do the translation, we assume that a relevant outfit intersects with a number of items above a certain threshold from the number of items in the ground truth. The threshold we have used is 0.7. A relevant outfit for a specific user means that it contains relevant materials and patterns and categories that are similar to what the user prefers in the ground truth. So, we compute precision and recall at K where k is a user defined integer that reflects the number of recommended outfits to the user. Based on the assumption that a recommended item should be a relevant item, Precision at k is the proportion of recommended items in the top-k set that are relevant and is calculated as follows: $P(k) = \frac{RR@k}{R@k}$ Where RR is the number of recommended items that are relevant @K, and R is the number of recommended items @k. $AP(k) = \frac{1}{m} \sum_{k=1}^{N} (P(k)$ if kth item was relevant) where m is the number of outfits. Then we calculate the MAP which is the average of the average precision metric over all the results' list. What is noticed from the results is that the MAP values for *Outfit2Vec(PV-DM)-SE* and *Outfit2Vec(PV-DBOW)-SW* were similar. However, *Outfit2Vec(PV-DBOW)-SE* had increased values when compared to *Outfit2Vec(PV-DBOW)-SW*. It is also expected that they both outperform the models that were trained with random sequences. **Figure** 5 illustrates the MAP performance for all the models. A clear decrease is noticed towards the models that were not trained with the structured methodology. As described in the methodology section, we have compared different candidate vectors before choosing the structure we use in our experiments. **Figure** 7 illustrates the results of evaluating the candidate vectors with the metrics NDCG, MRR, precision and recall @k = 1. The chosen representative vector had the highest values for the evaluated measures.

4.3 Partial Outfits Recommendation

As previously described, Outfit2Vec represents the whole-outfits recommendation task, and PartialOutfit2Vec represents the recommendation of a clothing entity within an outfit's sequence. In partial outfits experiments, we split the outfit sequences in a way that includes a group of clothing entities and infers the remaining entity. For example: in the sequence "colourful lace top topsAndTshirts - striped crocheted short trouserAndShorts -checked leather sandal shoes", the first two entities are provided to the model for inference of its representation where the top k results of the most similar completing part of sequence are retrieved and compared to the ground truth which is the last entity in the sequence. The same approach of structured entities and structured words was applied in this experiment, so the models under evaluation are: PartialOutfit2Vec(PV-DM)-SE, PartialOutfit2Vec(PV-DM)-SW, PartialOutfit2Vec(PV-DBOW)-SE and PartialOutfit2Vec(PV-DBOW)-SW. PartialOutfit2Vec was compared with Word2Vec (Skip-gram) for the task of predicting a clothing entity from a sequence of an outfit. We have also compared against the Word2Vec (CBOW) model but the results were significantly lower so we focused on

Table 1: Evaluation Results of Avg. NDCG, MAP and Avg. MRR for k = 30 and 40 for all the models under comparison in whole outfits prediction task

Model	NDCG@30	NDCG@40	MAP@30	MAP@40	MRR@30	MRR@40
Outfit2Vec(PV-DM)-SE	**0.22**	**0.33**	0.37	0.41	0.06	0.06
Outfit2Vec(PV-DM)-SW	0.08	0.09	0.39	0.44	0.06	0.05
Outfit2Vec(PV-DBOW)-SE	**0.30**	**0.38**	0.37	0.41	0.07	0.07
Outfit2Vec(PV-DBOW)-SW	0.08	0.10	0.21	0.23	0.04	0.04
PV-DBOW-Random	0.08	0.09	0.13	0.14	0.03	0.03
PV-DM-Random	0.07	0.07	0.23	0.23	0.04	0.03

Table 2: Evaluation Results of Avg. NDCG, MAP and Avg. MRR for k = 30 and 40 for all the models under comparison in partial outfits prediction task

Model	NDCG@30	NDCG@40	MAP@30	MAP@40	MRR@30	MRR@40
PartialOutfit2Vec(PV-DM)-SE	0.43	0.60	0.26	0.34	0.19	0.26
PartialOutfit2Vec(PV-DM)-SW	**0.77**	**0.86**	**0.65**	**0.67**	**0.59**	**0.58**
PartialOutfit2Vec(PV-DBOW)-SE	0.54	0.67	0.34	0.38	0.28	0.31
PartialOutfit2Vec(PV-DBOW)-SW	**0.77**	**0.79**	**0.82**	**0.81**	**0.74**	**0.75**
Word2Vec-SkipGram	0.07	0.08	0.19	0.29	0.05	0.05

skip-gram model. The hyperparameters used for Word2Vec are: window size: 10 and embedding size 200. The values were chosen based on the average length of the outfit's sequence details. As can be noticed from Table 2, both *PartialOutfit2Vec(PV-DM)* and *PartialOutfit2Vec(PV-DBOW)* have outperformed Word2Vec. This can be explained by the ability of ParagraphVector-based models to capture the relationship between components of a paragraph more than word2vec models. Interestingly, the **structured words** trained models have performed significantly better than the **structured entities** models for the partial outfit prediction task. We explain this improved performance by the length of predictions as it is shorter than the whole outfits predictions. While at the whole outfits prediction, the whole sequence is to be predicted and the structured entities have shown more improvements in that scenario. For MRR evaluations, it was noticed that both PartialOutfit2Vec models have outperformed Word2Vec.

We conclude that the structured approach has achieved improvements in prediction generally, but the structured entities showed additional value at the whole-outfits prediction task.

4.4 MultiClass Classification Evaluation

As one of our objectives is to classify outfits into their relevant styles, we have compared the effect of the suggested methodology on the performance of the following baselines in a multiclass style classification experiment: (a) Convolutional Neural Network (CNN), (b) PV-DBOW, (c) Word2Vec, (d) TF-IDF for transforming text and Multinomial variant of Naive Classifier, (e) TF-IDF for transforming text and Logistic Regression classification, (f) TF-IDF for transforming text and Support Vector Machines (SVM) classification. **Figure** 6 shows that the models performed similarly when provided with input after applying our methodology. Average accuracy, precision, recall and F1 score were calculated. As the figure shows, the performance of the classification was similar. We conclude from this experiment that our methodology affects the recommendation

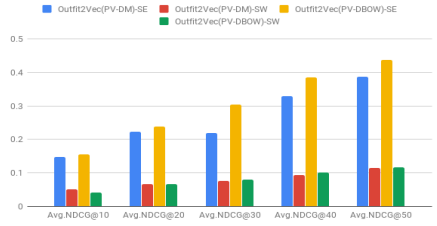

Figure 4: Illustration of the improved NDCG results achieved using the representative vectors methodology when applied to the models under study.

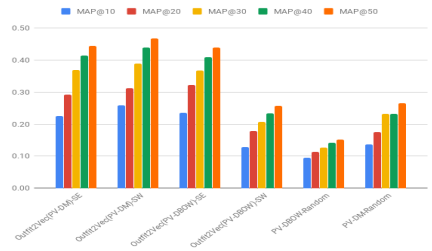

Figure 5: Illustration of the MAP performance for all the models.

evaluation task more than the classification. We explain this by the amount of details that affect the recommendation task, whereas in the classification, only the style value is predicted, which shows the value of our approach in a fine-grained recommendation evaluation tasks.

4.5 Discussion

As presented in the experiments section, projecting the clothing details as separate entities have improved the accuracy of retrievals

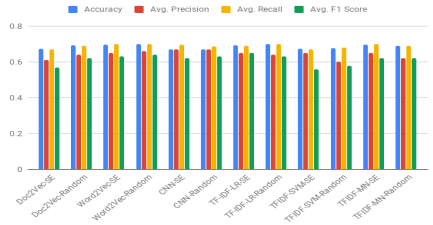

Figure 6: Comparison of Multiclass Style Classification results for different baselines using the described methodology

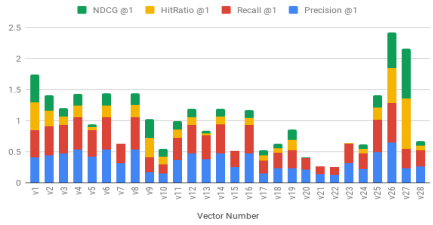

Figure 7: Illustration of representative vectors choice strategy.

in a significant way at the whole-outfits recommendation evaluation. The improvement was noticed on the rank evaluation measures which is very important for the outfits recommendation task. Both whole and partial outfits prediction tasks have been improved using the described **structured entities** and **structured words** approaches. We have also noticed that our methodology can be more important to fine-grained recommendation than to simple classification tasks. As shown in the classification task's evaluation, the models performed similarly with no noticed significant changes. This can be explained by the level of difficulty we have in the recommendation task, as the model is expected to find predictions similar to the ground truth in terms of multiple clothing details, while for the classification, the number of labels which are the style labels in this case are limited. No pre-trained models were used in our experiments for two major reasons: (a) in our methodology we create new words in the model's vocabulary by combining the clothing entity details, so having a pre-trained words model will not add a value or enhance the training in this case, (b) PV-DBOW models use the paragraph vectors for training and then for inference, so having a pre-trained words model will not add a value as well. Same applies for PV-DM based models where the training happens by concatenating the words and the paragraph vector.

5 CONCLUSIONS AND FUTURE WORK

We present Outfit2Vec and PartialOutfit2Vec models for learning clothing embeddings. In our models, we deal with a complex scenario of hierarchically-composed clothing items where we aim to recommend whole- and partial- outfits that consist of multiple clothing entities. Our objective from this work is to present a general strategy of dealing with learning representations of hierarchically-composed complex structures to learn their embeddings as unique instances within a taxonomy. We showed in our experiments that we run on an extensively-annotated Instagram dataset on recommendation and multi-class classification tasks the improvements achieved with our approaches. For our future work, we plan to experiment on larger-scale samples of data to evaluate the performance of our model in different settings.

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 1–6.

[2] Elaine M. Bettaney, Stephen R. Hardwick, Odysseas Zisimopoulos, and Benjamin Paul Chamberlain. 2019. Fashion Outfit Generation for E-commerce. *arXiv e-prints*, Article arXiv:1904.00741 (Mar 2019), arXiv:1904.00741 pages. arXiv:cs.CV/1904.00741

[3] Brandon Brown. 2018. 19 Micro-Influencer Statistics You Must Know In 2018. (2018). https://www.grin.co/blog/19-micro-influencer-statistics-you-must-know-in-2018

[4] Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, Ricardo Baeza-Yates, Andrew Feng, Erik Ordentlich, Lee Yang, and Gavin Owens. 2016. Scalable semantic matching of queries to ads in sponsored search advertising. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 375–384.

[5] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1809–1818.

[6] Kim Hammar, Shatha Jaradat, Nima Dokoohaki, and Mihhail Matskin. 2018. Deep text mining of instagram data without strong supervision. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, 158–165.

[7] 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*. ACM, 1078–1086.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[10] Shatha Jaradat, Nima Dokoohaki, Ummul Wara, Mallu Goswami, Kim Hammar, and Mihhail Matskin. 2019. TALS: A Framework For Text Analysis, Fine-Grained Annotation, Localisation and Semantic Segmentation. In *To appear in Proceedings of the 1st Workshop on Deep Analysis of Data-Driven Applications - COMPSAC 2019: Data Driven Intelligence for a Smarter World IEEE Conference*.

[11] Wang-Cheng Kang, Eric Kim, Jure Leskovec, Charles Rosenberg, and Julian McAuley. 2019. Complete the Look: Scene-based Complementary Product Recommendation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 10532–10541.

[12] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[13] Paris Martineau. 2018. Inside the pricy war to influence your Instagram Feed. (2018). https://www.wired.com/story/pricey-war-influence-your-instagram-feed/

[14] Chris McCormick. 2018. Applying word2vec to Recommenders and Advertising. (2018). http://mccormickml.com/2018/06/15/applying-word2vec-to-recommenders-and-advertising/

[15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[16] Thomas Nedelec, Elena Smirnova, and Flavian Vasile. 2017. Specializing joint representations for the task of product recommendation. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*. ACM, 10–18.

[17] Aaron Orendorff. 2019. The State of the Ecommerce Fashion Industry: Statistics, Trends Strategy. (2019). https://www.shopify.com/enterprise/ecommerce-fashion-industry

[18] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 225–232.

[19] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*. 4642–4650.