POLITECNICO DI MILANO
Master of Science in Computer Science and Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

New Recommendation Methods for Outfit Generation

Supervisor: Prof. Paolo Cremonesi

M.Sc. Thesis by:
Federico Sallemi, 896772
Gabriele Prato, 899662

Academic Year 2019-2020

Optionally, here goes the dedication.

Abstract

Traditional Recommender Systems rely on finding similarities between users and/or between items. In its broadest definition, a Recommender System tries to predict the preference a user would give to an item. While Content-Based approaches try to discover similarities between items and then predict a user's preference based on its past interactions with the items, Collaborative-Filtering approaches try to find similarities between users and recommend to an user what similar users already bought.
In the fashion domain, though, the user may not buy an item per se, but rather because it would fit in an ideal outfit that the user may want to wear. This behaviour implies that the content similarity between the items already bought by the user is not enough to make accurate predictions. Thus, it would be more reasonable to suggest the purchase of *compatible* clothes, rather than *similar* ones.
The problem of scoring compatibility of different outfits and learning a concept of style has already been tackled in research community by application of different machine learning techniques; however, tasks and datasets used to evaluate state-of-the-art models make some unrealistic assumptions that would not hold in a real-case scenario. This thesis introduces a novel algorithm to tackle the problem of learning outfit styles, in order to classify ensemble of clothes as fashionable outfits and complete them in a fashionable manner. Moreover, this work presents a proper comparison with the state-of-the-art models on the most used public academic datasets in this domain and on a real industrial dataset, provided by H&M. Additionally, a novel evaluation task, that releases some of the constraints existing in the tasks presented in literature, is introduced in order to assess the potentials of the different algorithms when dealing with problems more similar to those faced in real-case scenarios. Finally, this thesis attempts to move the problem of outfit completion from a general classification problem, into the Recommender Systems domain and evaluates the performances of these algorithms using some of the typical metrics used in Information Retrieval.

Sommario

I Recommender System tradizionali si basano sul computare similarità tra utenti e/o tra item. Nella sua definizione più comune, un Recommender System punta a predire la preferenza che un utente assegnerebbe ad un item. Mentre gli approcci di tipo Content-Based tentano di imparare le similarità tra items per poi predire le preferenze di un utente sulla base delle sue passate interazioni con gli item, gli approcci di tipo Collaborative-Filtering tentano di trovare similarità tra utenti per poi raccomandare ciò che utenti simili hanno già comprato.
Nell'ambito del fashion, in contrasto con quanto detto, l'utente potrebbe non essere interessato ad un item in sé, ma piuttosto perché starebbe in un outfit ideale che interesserebbe l'utente. Questo tipo di comportamento implica che la similarità di contenuto con gli item già comprati dall'utente non è sufficiente per fornire predizioni accurate; di conseguenza sarebbe più ragionevole suggerire l'acquisto di vestiti *compatibili* piuttosto che semplicemente *simili*.
Il problema di calcolare la compatibilità di outfit diversi ed imparare un concetto di stile è già stato affrontato dalla comunità dei ricercatori applicando diverse tecniche di Machine Learning; tuttavia, i tasks e i datasets usati nella valutazione dei modelli dello stato dell'arte si poggiano su alcune assunzioni poco realistiche che non reggerebbero in un caso d'uso reale.
Questo lavoro di tesi presenta un nuovo algoritmo per affrontare il problema dell'apprendimento di stili legati ad outfit, in modo da poter classificare insiemi di vestiti come outfit "alla moda" e completarli rispettandone lo stile.
Inoltre, questo lavoro di tesi introduce una comparazione con gli algoritmi dello stato dell'arte sui dataset accademici più utilizzati nella ricerca e su un dataset di tipo industriale, fornito dal nostro partner H&M. In aggiunta a ciò, viene introdotto un nuovo task di valutazione, che consente di superare alcune delle limitazioni esistenti nei precedenti task utilizzati, in modo da poter constatare il potenziale dei diversi algoritmi nell'affrontare problemi più simili a quelli incontrati in reali casi d'uso. Infine, con questa tesi, si

cerca di spostare il problema del completamento degli outfit da un generico problema di classificazione nel dominio dei Recommender System, e di valutare le performance degli algoritmi usando le metriche più utilizzate in Information Retrieval.

Acknowledgements

We would like to thank all the organizations that supported us and made this research work possible. We would like to extend a special thank you to our industrial partner H&M for making their private data available, contributing to generating new knowledge in the domain; we would also like to especially thank the Boston Consulting Group and ContentWise for supporting us during this whole work.

Notation

Abbreviations

Acc	Accuracy
AUC, AUROC	Area under Receiver-Operating-Curve
BiLSTM	Bi-directional Long Short-Term Memory
CE	Compatibility Estimation task
FC	Fully Connected Layer
FFN	Feed-Forward Neural Networks
FITB	Fill-in-the-blank task
LSTM	Long Short-Term Memory
MAP	Mean Average Precision
MLM	Masked Language Modelling task
NLP	Natural Language Processing
NN, DNN	(Deep) Neural Networks
NSP	Next Sentence Prediction task
RNN	Recurrent Neural Network
RR	Reciprocal Ranking
UOC	Unconstrained Outfit Completion task

Table 1: *Abbreviations*

Symbols

$\mathcal{O}$	$\triangleq$	Set of all the outfits
$\mathcal{I}$	$\triangleq$	Set of all the items
$\mathcal{C}_{O'}$	$\triangleq$	Subset of items to choose from for FITB questions to complete $O^{'}$
O	$\triangleq$	Outfit, i.e. a set of items
i	$\triangleq$	Item, i.e. a piece of clothing
${}^{i}p$	$\triangleq$	Item profile
$O^{'}$	$\triangleq$	Incomplete outfit
$i^{'}$	$\triangleq$	Missing item from $O^{'}$ and inserted in $\mathcal{C}_{O'}$
$\mathcal{P}(O)$	$\triangleq$	Powerset of set O
x_{vis}	$\triangleq$	Visual features vector of an item
x_{text}	$\triangleq$	Textual features vector of an item
x_{cat}	$\triangleq$	Categorical features vector of an item
x_f	$\triangleq$	Features vector of an item
cat_groups	$\triangleq$	Number of category groups for an item
$\|x\|$	$\triangleq$	Euclidean (L_2) norm fo vector x
r	$\triangleq$	Sampling factor, number of negatives sampled for each positive

Table 2: *Symbols*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The present research work aims to address some of the interesting challenges that stem from the exploration of the domain of Fashion Analytics.
Since it represents the area of focus for the present research work, it is useful to start by defining what the fashion domain is.
Fashion, in its modern significance, is a style that is popular at a particular time, especially in clothes, hair, make-up, etc. [Pre08]. As new digital ways of exploring and consuming fashion take the lead in the world markets, being able to propose fashionable ensemble of items that can soar in popularity becomes an ever more interesting endeavour; while up to now such an extremely relevant task has always been a prerogative of highly experienced fashion stylists, the question of whether the task can be automated and to which degrees of quality becomes a more and more relevant one for the entire fashion industry.
It is also useful to notice which the fundamental set of evaluation for research in this domain is: a fashion outfit is an ensemble of clothing items that maintains a coherent style, that can be considered pleasing in its overall composition and that is in line with the general taste of fashion of its present time frame.

Since most of the characteristics that make an outfit fashionable are rather subjective and difficult to effectively measure, working with fashionable outfits composition presents some unique challenges both in terms of problem definition and in terms of definition of the evaluation metrics.
Those fundamental challenges also translate into a very challenging technical environment in which to elaborate a possible solution.

1.1 Recommender Systems

The space of technical solution in the perimeter of this research refer to the domain of Recommender Systems, as a consequence of the fact that the ambition of this thesis is to work with the problem of automatically generating outfits for the fashion domain, and thus Recommender Systems are the closest available solution.
As a broad definition, a Recommender System is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item [RRS11]. As this definition has been used for a long time to refer to generic Recommender Systems across all domains, in some specific areas of application for that category of algorithms (such as the Fashion domain) the definition itself fails to incorporate the nuances and the peculiarities of solutions in the domain, and thus as a consequence General Purpose Recommender Systems (as in recommendation systems not specifically designed for a single application domain) reveal to be inadequate to serve the specific needs of an area like Fashion recommendations.

1.1.1 Why a domain specific Recommender System?

As from the definitions previously stated, it is possible to realize the significant distance that this specific area of application bears to the more "classical" application domains for Recommendation Systems, on which most of the state of the art is focused.
For example, a typical real-case scenario case can be represented by an user that looks for items in an e-commerce website that sells clothes. While filling the e-cart, a traditional recommender system would start suggesting items similar to those inserted in the cart or items bought by similar users. It is likely that the user has an outfit in mind, or would be interested in recommendations that complete an outfit containing those few items that he/she already selected. For example, if the users chose some shoes and trousers, he/she would like to be suggested clothes (e.g. shirts, blouse, sweaters) that fit with them and not clothes that are similar to them.
In order to make those macroscopic distances more clear, they can be summarized in the following list:

- In most of the domains in which Recommender Systems are deployed (movies, e-commerce, etc.), similarity between items has been used by researchers as a good proxy to evaluate which item to recommend. In opposition to that, in the Fashion domain similarity is not a usable proxy for recommendation, and compatibility needs to be evaluated instead (Section 2.2)

- For most domains, as above mentioned, the usage of metadata belonging to products is sufficient to produce acceptable results, while in the Fashion domain raw visual features belonging to products representations contribute for the most part of the performances of the algorithm [Vas+18]
- While the focus of General Purpose Recommender Systems insists on users as the base granularity on which to provide recommendation, in the case of the present research sets of items represent the unit to which recommendations need to be provided

As a consequence of all the above stated reasons, research in the field of Recommender Systems for the Fashion industry steered towards the investigation of solutions tailored to the needs of the specific industry, and managed to propose a number of diverse approaches in order to meet the needs of such a domain.

1.2 Tasks and Challenges

Exploring the specific domain of Recommender Systems for Fashion, it is possible to realize that two main tasks have been used in literature until now as the main benchmarks for performances of the proposed algorithms.
The first of the above mentioned task has been named Compatibility Estimation (Section 2.1.1), and consists in a typical machine learning classification task in which an outfit is given a binary label; the label assigned represent the outfit being either style coherent or not.
The second main benchmark task has been called Fill-In-The-Blank (Section 2.1.2), and consists in a selection task where one of the items composing the given outfit is removed and placed in a "subset of choice" with three additional items randomly sampled from the same category (such as shirts, shoes etc..); at this point, given the incomplete outfit and the subset, the algorithm is asked to select the previously removed item.

1.2.1 Limitations of Current Tasks

Both of the above mentioned tasks, though, present significant limitations in terms of their proximity to real-world recommendations tasks, which can be summarized as follows:

- Compatibility Estimation is useful only to test a discriminator, but gives few information on capabilities of a generator
- In Fill-In-The-Blank the set for selection doesn't take into account the entire possible catalogue

1.2.2 Novel Task

In an effort to overcome the above stated limitations that are holding back research in the field, part of the contribution for this thesis consists in proposing a novel benchmark task, named Unconstrained Outfit Completion, which consists in removing from a given outfit n items and for each of those, recommend k items selected from the whole available catalogue.

1.3 Previous Approaches

The solutions that have been proposed over time in order to meet the specific needs that arise from the study of recommendations specific to the Fashion industry have been, as previously stated, relying on a number of different approaches.
It is possible to identify a path of development in the field itself, starting from seminal approaches that would focus solely on the classification of single items inside its own category such as [Bos+12] and [Yam+12] and then going through the first applications that would switch the main intent of the research to the goal of generalizing and understanding a notion of "Style" that could be proper to a specific fashion dataset [Kia+14]; after that, it is possible to majorly focus on the most recent works that experimented with a number of different techniques, from Bidirectional LSTMs [Han+17] to Type-Aware Embeddings [Vas+18], with the goal of improving performances in now well defined tasks specific to this sub domain.
The definition of those above mentioned tasks, such as those given in Section 2.1, can be identified as one of the most relevant milestones that contributed in propelling the field forward towards its maturity with regard to other fields of Information Retrieval techniques, and most specifically of Recommender Systems.

1.4 Problem Statement and Knowledge Gap

Stemming from the considerations and observations above reported, this research work started investigating more thoroughly the available literature for the current state of the art solutions, and found that in the field there are multiple areas which are yet to be explored, and a number of gaps in knowledge that can be filled with additional research and future works in the domain. One of the most relevant gaps that was possible to verify during this thesis work, and for which a solution has been proposed in Section 2.1.3, has been the absence of a well defined benchmark task for the field that could have the following two characteristics:

- Allowed to apply evaluation metrics that are typical of the Recommender Systems domain and of Information Retrieval more generally, in order to have a fair and direct comparison with available state of the art in other research domains
- Could be used as a close enough proxy of the actual tasks that a Recommender Systems would be carrying out in an industrial setting

Another major gap that needed to be filled in the domain, that is being presently analyzed, was the lack of an architecture that could be able to scale its performances enough (in terms of both number if users served and time per inference) while still being able to perform at state of the art level. In order to make this concept clearer, it is possible to imagine a generic user of any e-commerce website from a major Fashion company (as, for example, our industrial partner H&M) trying to have a personalized shopping experience; with currently available solutions, it would have been impossible to provide him/her, and all other users, with a personalized recommendation on how to complete an outfit based on his/her personal taste, and the proposed solution for this research works aims at filling also this gap.
Lastly, up to now, no on-line study with professional stylists from the Fashion industry had been carried out, so that if a researcher had been interested in drawing a correlation between metrical, offline measured performances on well known benchmark tasks with actual performances in an industrial setting he would have been unable to do so; in this thesis work, an evaluation that aims at filling this gap is also proposed.

1.5 Solution and Contribution

In an effort to provide an answer to the gaps previously individuated and exposed, this research proposes a new architecture solution to improve the state of the art for the domain and help direct further research on the still consistent challenges ahead.
The basis for the newly proposed solution can be found in the Attention

Mechanism [Vas+17], which allows to learn the correlations between the content of two sequences/sets while at the same time be agnostic to the specific position that this content occupies.
The implementation of that mechanism that has been selected to serve as basis for the solution is the Transformer Network, and more specifically the implementation that was originally developed for the Natural Language Processing domain by Liu et al. [Liu+19]; that specific implementation has been used as the second stage for an end-to-end trainable architecture, comprising a first stage which is tasked with extracting representative features for all available items and embed them in a common representation space, where a notion of style is maintained.
In order to tailor the solution for the fashion domain, the utilization of the special tokens handled by this kind of transformer network have been specialised for them to handle specifically the benchmark tasks typical of this research field.
Then, it is possible to highlight the main contributions that this newly proposed solution brings to the field in exam:

- Increased performance w.r.t state of the art for both main benchmark tasks
- Capability to treat outfits as unordered sets thanks to the Transformer architecture
- Capability to scale in an Industrial environment, bringing down complexity for inference from a quadratic to a linear complexity

On top of the contributions delivered by the new architecture, this research works also delivers:

- First available analysis of an Industrial dataset, providing a summary of differences with what is publicly available in Section 3.2
- A new benchmark task which is closer to real-world implementation scenarios with baseline metrics and performances for future work in Section 2.1.3

1.6 Research Question and Takeaways

Summarizing what said until now, the main research questions for this work are as follows:

- ***Is it possible to improve the state of the art performances in fashion outfit composition, while allowing for both representation of outfits as sets and evaluation of an entire catalogue?***
- ***Is it possible, on top of that, to tackle the problem of unconstrained outfit generation while performing at least on-par with human expert stylists?***

Following this, the main goal for this work has been to answer positively to all the main inquiries of the research question, while at the same time being able to propose a solution that would be able to scale to the needs of a large industrial partner such as the one (H&M) that supported this work.
It is possible to say that this research work achieves the goals that had been set for it, by delivering a 5,91% increase in performance for the Compatibility Estimation task (Section 2.1) and a 11,21% increase on Fill In The Blank task (Section 2.1), as is more thoroughly explained in Chapter 6.
A lot is still to be done even in the perimeter of this same research work, like for example explaining in a rigorous scientific way the only case in which the proposed solution ends up performing worse then available state of the art solutions, case for which we provide a number of hypothesis in Section 6.3.1.

1.7 Methodology Summary

In carrying out the present research, the process followed in order to be able to present sound and reliable results, as well as reproducible insights for the domain, has been articulated into three main steps.
In the first phase of the work, an extensive analysis of the available state of the art techniques and benchmark tasks has been carried out, validating each potential baseline by reproducing all relevant results reported in literature; at the same time, non deep-learning related approaches have been tested in order to produce evidence of the need to resort to more complex solutions in order to obtain satisfactory results in the fashion domain.
Secondly, the most relevant limitations and drawbacks present in the past state of the art research has been highlighted and evaluated with regard to its impact in preventing a successful industrial application to be produced, and insight from this analysis has been used to inform the direction of the research in order to provide significant improvements to state of the art approaches in the domain at hand.
Lastly a novel architecture, aimed at solving the pressing limitations that a Recommender System specifically built for fashion poses, has been implemented and tested, comparing performances with previous state of the art

techniques both on well known benchmark tasks and on a novel task here defined and presented.

1.8 Outline

The rest of the thesis is organized as follows:

- Chapter 2 defines the tasks already faced in literature and the new task we will introduce, together with the respective evaluation methods;
- in Chapter 3, the datasets used to train the proposed algorithm and test its performances will be described, highlighting similarities, differences, and the necessary operations to clean them;
- Chapter 4 outlines the solutions presented by other authors to tackle the task outlined above, and the description of the supportive technologies adopted in the solution proposed in this thesis;
- Chapter 5 contains the description of the proposed solution, its architecture and implementation details, its potentiality, and compares it with the state-of-the-art algorithms;
- Chapter 6 presents the results obtained in the different tasks and compares them with the state-of-the-art;
- Chapter 7 summarises the work, the contributions and the achievements of this thesis and proposes some future research lines.

Chapter 2

Background

The generation and completion of an outfit in the fashion domain is one of the basis on which fashion is founded. Being able to work with outfits is a challenging problem and it encapsulates a number of different tasks that may end up being relevant in different applications even outside of the specific field. Since generating an outfit that is both relevant and coherent in terms of style is a very complex problem, it can be analyzed as constituted by a number of different sub-tasks, each playing a specific role in shaping the algorithms that have the ambition to tackle the issue.
As a consequence of the application of Machine Learning techniques in the fashion domain being fairly recent (the first relevant papers discussing visual analytic and outfit generation have been published around 2017 [Li+17; Han+17; Vas+18]), even tough facing a significant increase in term of effort from the scientific community in the latest years, it is not possible yet to find a comprehensive literature review drawing the evolution of state-of-the-art methods in outfit compatibility estimation and outfit generation nor finding a unique standard in the definition of the tasks.
Despite the above mentioned lack of a comprehensive work carried out in outlining the state-of-the-art landscape, in this thesis work a brief description of the evolution of techniques in the field is going to be given in Section 2.1 in parallel with the description of the main tasks guiding the research in the fashion domain; furthermore, a more in-depth analysis of the most relevant present state-of-the-art techniques is given in Section 4.1.

2.1 Tasks in Fashion

The previously mentioned evolution of state-of-the-art techniques in this specific research domain can be described by looking at both the technologies involved in the architectures proposed by researchers and the tasks tackled in those proposed works. The first solutions in the space of fashion visual analysis can be identified in the works presented by Bossard et al. [Bos+12] and Yamaguchi et al. [Yam+12], where the task that captured the focus of researchers was limited to the classification of fashion items into their proper categories, exploiting as underlying technology simple RGB direct embedding or the use of either Random Forest approaches or Human Pose Estimation. Soon after the previously discussed seminal approaches were published, the largely influential work from Kiapour et al. [Kia+14] was released, tackling mostly the task of finding compatibility as a mean of classification, but paving the way for most of the future literature by introducing two main innovations in the modalities to approach research in the domain:

- They curated a new, fashion specific, dataset with information about the "perceived style" of a given outfit collected from users
- They used as target of their classification effort not the "catalogue" categories (shirts, tops, trousers etc.) but their own definition of "category" according to the style classification of a given item.

As a consequence of this new perspective in approaching the fashion domain, working with the objective of understanding, classifying and utilizing "style" for recommendations and suggestions for the industry rapidly became the new paradigm guiding researchers efforts in the subsequent years.
From 2016, research in the field had a significant impulse and a number of new works were published; the main innovation presented in this time span was the use of Convolutional Neural Networks (CNN) as the primary way to extract visual features from item's images as in Bracher, Heinz, and Vollgraf [BHV16]; this method of extracting visual features has become, in a wide variety of different fields of application, the de facto standard for working with visual features and in the same way became the standard for research in the fashion domain, too, giving researchers the chance to possibly identify complex and non-linear relationship between the visual characteristics of items.
From that moment on, it is possible to identify two main tasks of interest in the domain of fashion recommendation, that have driven forward the research and informed the choice of the architectures deployed to provide solutions in that space:

- Outfit Compatibility Estimation (CE)
- Fill-in-the-Blank Outfit completion (FITB)

In the development of this thesis, a new task is presented:

- Unconstrained Outfit Completion (UOC)

All these tasks are evaluated on a sequence of questions. Each question is an outfit (eventually incomplete in FITB and UOC) of the test sets, and the answer to provide is a binary label for the classification task (CE), or the choice of an item from a small set (FITB), or the choice of one or more items from a whole catalogue/dataset (UOC).

2.1.1 Compatibility Estimation

The first of the two main tasks that took the front stage in the fashion domain is the Compatibility Estimation of an outfit. In this kind of task, as it is defined in most of the relevant papers published in the domain ([Li+17], [Han+17], [LSL17]), a model is asked to distinguish between real outfits and ensembles of random items. It is evaluated as a classification task, asking the algorithm to provide a binary score to the outfit, distinguishing between those whose clothes fit together and those that do not. In literature, the algorithms performing outfit compatibility are compared on AUC, but other metrics for binary classification (i.e. precision, recall, accuracy) can be used as well to evaluate the performance of an algorithm. The most utilized dataset (cf. Chapter 3) for this task only includes human-generated outfits, thus reducing the ground truth to only positive samples. In almost all the papers in literature, the negative samples are generated by random sampling items to create outfits containing clothes that do not fit together. Anyway, the sampling method of the negatives and the space from which the samples are selected (both for training and testing phase) strongly affects the performances and the evaluation of the goodness of the algorithms. For example, asking an algorithm to judge outfits that may contain more than one item per category (e.g two skirts, or four shoes) is an easier task than judging only well-formed outfits (i.e. that are actually wearable by a human being). Unfortunately, only few works (e.g. [Vas+18]) have a comprehensive and structured comparison between the results obtained using different sampling spaces and methods. In contrast to all the other papers, Li et al. [Li+17] decided to use the outfit *popularity* as a proxy for its compatibility estimation. Namely, the outfits with a number of *likes* higher than the 90^{th} percentile were considered positive samples, and those under the 40^{th} percentile were considered negative samples. Unfortunately, the field *likes* or other outfit-related metadata were present only in one of the available datasets (cf. Chapter 3 for a complete description of the datasets and their

structure).
The task of classifying compatible outfits has been approached in a number of different ways by recent research works in literature, but we may identify two main approaches that lead the way in terms of performance. In the papers from Han et al. [Han+17] and Nakamura and Goto [NG18] the architecture deployed in order to compute compatibility between fashion items is based on a Bidirectional LSTM network, used to learn the outfits provided as ground truth in the form of a sequence, in order to then indicate item compatibility as the probability of an item to be next in the current sequence (representing the current outfit).
What is instead the most common approach to handle this task is the utilization in series of a Convolutional Neural Network and a vectorization approach, with the objective of projecting fashion items into a common high-dimensional space, where it is possible to find compatibility using as a proxy the distance between items in that new space given by the entirety of the features of each item. This approach as implemented in papers such as Lee, Seol, and Lee [LSL17] vastly influenced subsequent research, and was later improved in the work of Vasileva et al. [Vas+18], where instead of projecting every item in a single common space, one space for each couple of item categories was created in order to be able to maintain diverse relationship between items of different categories.
Both those major approaches, and also all other approaches so far presented in published papers, rely on the assumption that the compatibility between pairs of items or between sub-sequences of items can be a proxy of the compatibility of the whole outfit, as a consequence of the pair being seen coupled together in at least an outfit of the dataset. The authors made such assumption due to the limitations of their architectures (cf. Chapter 4), and it can hold on the public datasets used in literature (cf. Section 3.1) due to their structure, where the number of items appearing more than once on the whole dataset is negligible. On different datasets, where the same item appears in many different outfits, the transitivity of item-compatibility may not hold anymore.

2.1.2 Outfit Completion

The second task which is a relevant benchmark in literature for the usefulness of an architecture is outfit completion, which is generally referred to as a Fill-In-The-Blank task. FITB consists in finding the missing item that best completes an outfit from a subset of possible choices; it is a recommendation task, where the ground truth consists of typically one missing item that

should be selected by the algorithm. From the Recommender Systems point of view, the outfit can be considered as a user, according to the standard Recommender Systems terminology. The type of missing items, their number and the sampling modality for the list of candidates varies from paper to paper. Typically, only one item is removed from the outfit, while the subset of proposed items to choose from contains the missing item and other three clothes [Vas+18], [Han+17] (the only exception is in Li et al. [Li+17], that uses subsets of five elements, one of which is the correct one). In its original formulation, the categories of the items (i.e. jackets, t-shirts, top, skirts, bottom, etc,) in the subset need not be the same, but, as pointed out and tested in [Vas+18], the sampling method strongly affects the testing results. In order to have a fair test, it would be required to put only items that share the same clothing category, otherwise a model able to distinguish categories would easily achieve great results on the original version of this task (e.g., given an outfit missing shoes, and a subset of possible choices containing only one pair of shoes and three items of different categories) even without learning a proper concept of style, but rather learning to count categories. Most of the work in literature used the first version of the task, but this work tested and compared all the algorithms using a category-aware sampling, as done by Vasileva et al., using also the results reported in [Vas+18] for this version of the task.
This task is evaluated on accuracy in literature, assuming that random would give an accuracy of 25%, given the choice set size of 4.
From a technical point of view, the FITB score provided by a model is strictly correlated to the CE score, since the former is solved by completing the outfits with the most compatible choice among the subset provided to the algorithm. While architectures that project item features in a different space using FFN are built to tackle first the CE task and then reused to tackle the FITB task (increasing the time complexity, since each possible choice must be evaluated), BiLSTM architectures use the opposite approach, tackling first the FITB task and the using the final state of the RNN to classify the outfit.
Also in the case of this second task, as part of the contribution expressed in this thesis work, a more general approach to the task is going to be defined and tested, taking the name of Unconstrained Outfit Completion.

2.1.3 Unconstrained Outfit Completion

This novel task, proposed here for the first time, is a generalization of the FITB task. It is the typical question that a well-structured Recommender

System is supposed to answer, i.e. recommend items that best complete a profile (in general purpose recommender systems that is the user profile, in this case an outfit). The number of relevant recommendations is then evaluated using some typical evaluation metrics used in the Information Retrieval domain, i.e precision, recall, mean average precision, accuracy, reciprocal rank. In this task, a model is given an incomplete outfit and then is asked to predict the missing items, given their categories. The search space of the items is the whole dataset, not just a small sub-sample.
It is relevant to notice that a fundamental requirement, for any solution aiming at solving this task in a way that can be useful in a real world use-case, is for the computational complexity of the algorithm to scale not more than linearly with the number of recommendations needed.
Formally: given an outfit O, $|O|-2$ incomplete outfits are created by removing items (i.e, each of the new outfits has a minimum of 2 and a maximum of $|O|-1$ items), for each of them the model is asked to predict the missing items, (a set containing from 1 to $|O|-2$ items), then the recommendations are evaluated using *precision@k*, *recall@k*, *f1@k*, *rr@k*, as defined in Section 2.2. For each missing item, the model is asked to return k recommendations.
There are some important considerations on the limitations, the meaningfulness and the computational effort of this task:

- Some outfits may share some items (and in a real-case scenario this may be frequent). If all the non-shared items end up in the missing items group (i.e. they are removed fro their respective outfits before testing), then these two outfits are completely equivalent. The selection of one of the two testing groups, would introduce a bias in the evaluation that was not present during the training time. Thus, in this case, the testing groups are merged.

- The most meaningful evaluation would require to predict all the relevant subsets from the testing group of an outfit (i.e. the set containing the missing items). Such type of test, that would really evaluate the capability of an algorithm to model the interactions between the items in an outfit, becomes computationally unfeasible due to the exponential growth of the number of generated subsets (up to $k^{|O|-2}$) and the computational cost of their comparisons with those present in the testing group. In order to create a suitable task, the outfits are created by taking those used in the test set of the CE task, and then for each outfit $|O|-2$ random items are removed, one at a time generating $|O|-2$ outfits, whose sizes are in $[2,\ |O|-1]$ and whose respective testing groups have size in $[|O|-2,\ 1]$. At this point, the goal becomes guessing the items in the testing group predicting k elements for each item category present in the testing group (Fig 2.1).

- Even tough it considers only the missing items separately (i.e. predicting the missing items, and not the subsets of the testing group that would properly complete an outfit), this task would evaluate the ability of the algorithm to model the relationship between the visible items and its ability to generalize such relationship to all the items present in a whole catalogue.
- By removing the constraint of the fixed size subset of answers, this task moves the problem of the outfit completion into the recommendation domain, closer to a real-case scenario. In a real-case scenario, for example, an user would choose some items in an e-commerce website and a suitable model shall be able to recommend the missing items to complete the outfit the users is willing to buy. In this case, the model would have to choose the items to recommend from the whole catalogue of purchasable clothes, rather than a subset of 4 elements as in the FITB task.

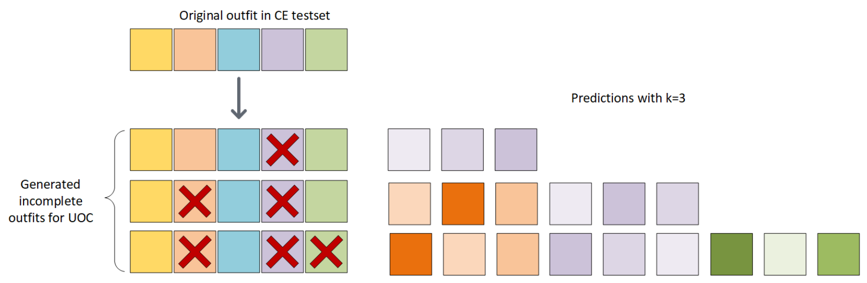
Original outfit in CE testset
Predictions with k=3
Generated incomplete outfits for UOC

Figure 2.1: *Generation of incomplete outfits and predictions for UOC task. Different colors mean different categories.*

2.2 Evaluation Methods

The metrics used to evaluate the tasks are the following:

- Compatibility Estimation task:

$$AUROC = \int_0^1 TPR(FPR^{-1}(x))dx$$

where TPR and FPR are the true positive rate and the false positive rate at different thresholds.

In this specific domain, a true positive is considered the correct guess of the outfit as originally present in the dataset or randomly generated.

- Fill-in-the-Blank task:

$$Accuracy = \frac{|guessed\ missing\ items|}{|questions|}$$

where guessed missing items are the items removed from the outfit and correctly selected by the algorithm from the subset of choice, and questions represents the incomplete outfit in the testset.

- Unconstrained Outfit Completion task:

$$\begin{aligned}
Precision@k &= \frac{|guessed\ items|}{k} \\
Recall@k &= \frac{|guessed\ items|}{|missing\ items|} \\
F1@k &= 2\frac{Precision@k \cdot Recall@k}{Precision@k + Recall@k} \\
RR@k &= \begin{cases} \frac{1}{rank\ guessed\ item} & if\ guessed \\ 0 & otherwise \end{cases} \\
MAP@k &= \frac{\sum_{c=1}^{k} Precision@c \cdot rel(c)}{|missing\ items|} \\
\text{where } rel(c) &= \begin{cases} 1 & if\ recommendation@c\ is\ a\ missing\ item \\ 0 & otherwise \end{cases}
\end{aligned}$$

All the metrics used in the UOC task are averaged across all removed items from an outfit and across all outfits in the testsets. Some of the metrics used for the UOC task are actually bounded by the fact that the missing item is only one per category in many of the questions to evaluate, especially in the Polyvore datasets (cf. Section 3.1), where most of the items appear only once in the whole datasets and some items are even labelled twice with different ids. So, for example, using a cut-off $k = 5$, precision will be bounded to a maximum of 0.2 in the vast majority of questions (see Tab. 3.1 for the statistics about the dataset), since the merging of the testing groups would rarely appear.

2.3 Similarities and differences with traditional recommendation domains

It is trivial to see the analogies between an user in a traditional Recommender System and an outfit in this specific domain: users are typically characterized by the items they interacted with, while outfits are characterized by the items they are composed of. Depending on the datasets, outfits and users may have other data not related to their interactions with the items, nevertheless those interactions are the key relevant data of any Recommender System. However, there are some key differences:

- the variance in the users' history/profile lengths is much wider than the outfits' lengths,
- every outfit is very similar to a new user, in terms of profile length. Thus, all predictions suffer from a quasi-cold-start problem, i.e. all outfits, once removed the items to guess, have few "interactions" (i.e. items) in their profile (typically less than 5, cf Chapter 3 for statistics),
- traditional Recommender Systems are asked to base their predictions on the concept of similarity (between items in Content-Based algorithms, and between users in Collaborative-Filtering ones), while the outfit completion task requires the notion of compatibility between the items composing an outfit.

Chapter 3

Datasets

As in most, if not all, Machine Learning related research works, what can be actually achieved in terms of both performance and generalization of the results is strongly dependent on the data that is available for the models to be trained and experiments to be conducted.
From the point of view of data alone, even tough the field of visual analysis for the fashion industry is one that caught the attention of the research community recently, there is a very limited number of publicly available datasets that allow for visual feature extraction and style inference. The most used datasets in the research community that provide also outfits-items annotations are described in the following Section 3.1. In the context of this thesis, an industrial dataset, provided from a fashion retailer, will be presented. All these datasets were used to train, validate and test the performance of the proposed model both in an academic perspective and in a real world scenario.

3.1 Public Dataset

In this section, the major public datasets used in this field of research is going to be presented. The datasets focus on metadata analysis and outfit prediction, and are currently the most famous benchmark datasets for all tasks concerning outfit classification and completion.
As more on that will be presented in section 3.2, part of the novelty of this thesis work also resides in the fact that it was possible to individuate significant differences in the organization, presentation and composition of an industrial, real-world dataset compared to the datasets routinely used in research as a performance benchmark. This work will present quantitative results that may have a direct application in a possible industrial utilization

in the tasks defined in Chapter 2.

3.1.1 Polyvore-21k

The Polyvore dataset is a fashion oriented dataset; it was built crawling user generated images, metadata and outfit composition information from a specialised fashion website called Polyvore.com.
There are many variants of this set of data currently in use in research, depending on the research team that crawled the website, the period when it was crawled and the information collected. One of the most widespread version of this dataset, used also in this thesis, is the one created by Han et al., commonly called "Polyvore21k" or "Polyvore Maryland" by other authors.
In this version of the dataset, there are in total 21889 outfits, divided as 17316 for training, 1497 for validation and 3076 for testing; the structure of the available data, and of the information contained, is as follows:

```
{
    "name": Name of the outfit,
    "views": Number of views of the outfit,
    "items": [
        Fashion items in the outfit.
        {
            "index": Index of item in this outfit on Polyvore,
            "name": Description of the fashion item,
            "price": Price of the fashion item,
            "likes": Number of likes of the item,
            "image": Image url of the item,
            "categoryid": Category ID of the item,
        },
        {
            ...
        },
        ...
    ],
    "image": Outfit image url,
    "likes": Number of likes of the outfit,
    "date": Upload date of the outfit,
    "set_url": Outfit url,
    "set_id": Outfit ID,
    "desc": Outfit description.
}
```

Listing 3.1: *Polyvore items JSON*

In this version, it is possible to also have access to evaluation data for both of the major tasks carried out in fashion prediction as they have been previously discussed in this thesis (CE and FITB). As far as Compatibility Estimation is concerned, the evaluation file contains 7076 outfits, where 3076 are listed as positive examples for compatibility and 4000 are listed as negative examples; analysing the FITB tasks, on the other hand, is a slightly more complicated matter, and as a consequence a JSON file is there provided in the form of questions to be answered by the algorithm:

```
{
    "question": Fashion item sequence to form the question,
    "answers": Multiple choice set to choose from,
    "blank_position": The blank position to be filled in.
}
```

Listing 3.2: *FITB task structure*

in this context, the algorithm's response is considered correct if it manages to choose, in order to complete the proposed outfit, the exact item that was part of the original outfit among the subset of items passed to the algorithm as "answers". In this formulation, the number of choices in the "answers" field is limited to 4, and the field "blank_position", that refers to the position of an item in a sequence-like outfit, was added as necessary to the model presented by Han et al., since it is based on RNN, as described in Section 4.1.3. In the original setting, the answers need not be of the same category, but this constraint reduces the capability of such task to test the real performance of a model. E.g., a model only capable to distinguish clothing category would perform as one able to learn stylistic notions by just pointing the item whose category did not appear in the "question", if the answers set includes element from different categories.
As it has already been discussed in Section 2.1.3 before, there are some limitations in evaluating the industrial effectiveness of a recommendation algorithm for fashion in such a way, limitations that we aim to step over introducing the novel task in Section 2.1.3.
Even tough the dataset described until here has been the starting point for many research papers in the domain, and has been also one of the most utilized benchmark in the research field for CE and FITB taks, soon it became clear the need to further refine the available data, by cleaning both the images present and the metadata associated; as a consequence of that decision, the dataset was reduced by removing items that were clearly out of context (furniture, house appliances, bicycles etc..) and their associated metadata.

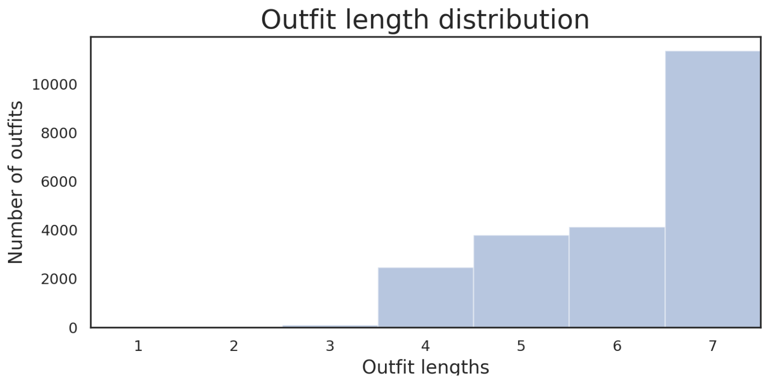

Figure 3.1: *Outfits lengths distribution on Polyvore-21k*

3.1.2 Polyvore-68k

In the context of this thesis, the Polyvore version provided in the work by Vasileva et al. [Vas+18] has been used as the reference dataset for the development of the proposed architecture.
This version of the dataset contains, more specifically, 68306 different outfits, built with 251008 items (Tab. 3.1). Each item is represented by an image, a textual description and a category. Even if much cleaner and bigger than the previous version, even this dataset contains some noisy and dirty data (non-clothing items, wrong labelling, duplicated data, images without labelling, white images). Also this data has been cleaned, e.g. by removing those images that were not labelled and discarding some easily identifiable duplicates. The resulting dataset has been used as the main reference point for all the experiments carried out in this thesis work, due to the fact that having a cleaner starting dataset allows for a better understanding of the actual performance of the architecture, and that this clean version is built form the dataset that is commonly used as benchmark for all recent state-of-the-art publications.
This dataset presents two different ways of splitting the data into training, validation and test set. The difference between the versions of the dataset, referred to with the name of "disjoint" and "non-disjoint", is that in the first there is no overlap of items between the outfits that are seen at training time and the ones used to build both the test and the validation set while in the non-disjoint split it is instead only guaranteed that the outfits present at training time will not be used to build testing and validation sets [Vas+18].

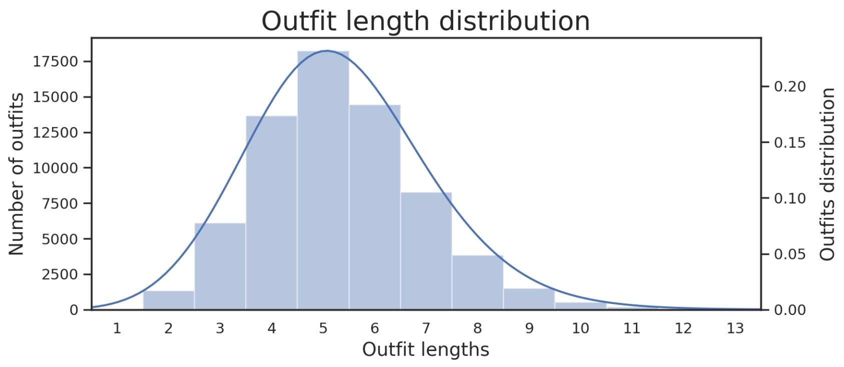

Figure 3.2: *Outfits lengths distribution on Polyvore-68k*

3.2 Industrial Dataset

As previously anticipated in the context of this thesis work, thanks to the very close and open cooperation in place with our industrial partner H&M, it was possible to leverage and work with a completely new industrial dataset, with a completely different data structure and quality statistics than any publicly available benchmark.
Since dataset was provided by a large industrial player in the field of fashion, it cannot be shared in this context; it is possible, although, to provide a description of its statistics and its main characteristics in a way that conclusions reached with the use of that same dataset may be of interest for any reader.
The structural difference that can be observed between the publicly available dataset and the industrial one is very significant, and the large difference in performance it fosters (differences that are going to be better discussed and presented in Chapter 6) also suggest that the dataset that is used today as benchmark reference for state of the art studies in the context of fashion recommendation is not a good proxy for performances in a real industrial environment.
One of the most interesting difference in structural composition between the two datasets can be observed in the different frequency with which a same item, being part of the training set, is seen as part of different outfits:
in case of the Polyvore dataset, items tend to appear only once (they tend to be part of a single outfit) with an average of outfits-per-item ratio of 1,45 while, in case of the private industrial dataset, items are often used to compose a significant number of different outfits with an average number of 69, 11 outfits-per-item ratio (Tab. 3.1).

The industrial dataset contains 659494 different outfits, composed using 33399 distinct items; another relevant characteristic of the dataset is that each of those items, instead of being assigned a single category like in the Polyvore dataset, is assigned 4 different categories.

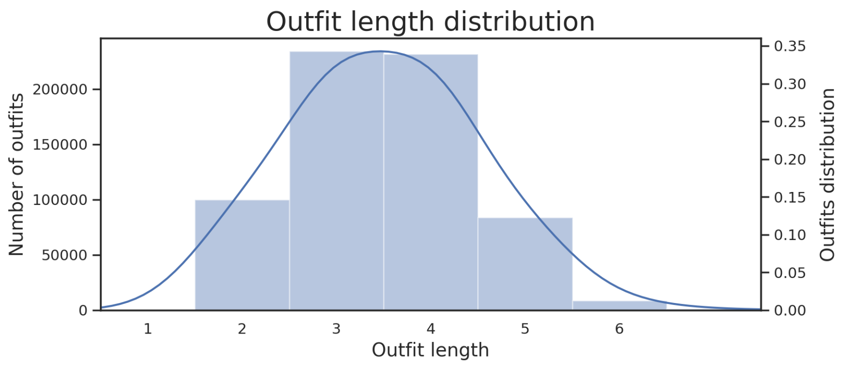

Figure 3.3: *Outfits lengths distribution on the industrial dataset*

On top of the structural differences that have been so far reported, one of the most relevant characteristics of the industrial dataset used to conduct this research is that it was originally provided, by the partner company, as a dataset consisting of only pairwise compatibility relationships between items; in order to extract a suitable outfit oriented dataset out of that, the pairwise relationships have been used to build a compatibility graph for the entire dataset, and the outfits have been built by applying a Maximum Clique Mining technique [BH92] on the resulting graph.
Since pairwise relationships do not guarantee the enforcing of fashion specific relationships that are central in the production of a well-formed outfit (for example, no two items in an outfit can belong to the same category etc..), we manually enforced a set of rules on the edges of the resulting graph in order to account for that, and the above mentioned rules are the following:

- For each item, no two items that are part of the same category can be in the same outfit

- No item belonging to category "Jeans" can be in an outfit together with an item belonging to category "Skirts"

- No item belonging to category "Shirts & Blouses" can be in an outfit together with an item belonging to category "Tops"

- No item belonging to category "Jeans" can be in an outfit together with an item belonging to category "Trousers & Leggings & Pants"
- No item belonging to category "Cardigans & Sweaters" can be in an outfit together with an item belonging to category "Hoodies & Sweatshirts"
- No item belonging to category "Jeans" can be in an outfit together with an item belonging to category "Shorts"
- No item belonging to category "T-Shirts & Vests" can be in an outfit together with an item belonging to category "Shirts & Blouses"
- No item belonging to category "Shorts" can be in an outfit together with an item belonging to category "Skirts"
- No item belonging to category "T-Shirts & Vests" can be in an outfit together with an item belonging to category "Tops"
- No item belonging to category "Trousers & Leggings & Pants" can be in an outfit together with an item belonging to category "Skirts"
- No item belonging to category "Trousers & Leggings & Pants" can be in an outfit together with an item belonging to category "Shorts"

The idea behind the technique used to mine the outfits from this dataset is that, given the graph $G = (V, E)$, where V is the set of items and E the set of edges connecting them, once removed the edges violating the above mentioned rules, the outfits are represented by the maximum cliques inside the graph G. The choice of mining the maximum cliques is based on the idea that all the vertices in a clique are adjacent, i.e. all clothing items represented by a clique are pairwise compatible. The basic assumption that the outfit compatibility is correlated to the pairwise compatibility of the items composing an outfit was already used by Vasileva et al. It is important to remember that the clique mining technique used is not based on any Machine Learning trainable model, but it just traverses the graph completely and extracts the subgraphs that are complete, i.e. without learning any new model nor data representation.

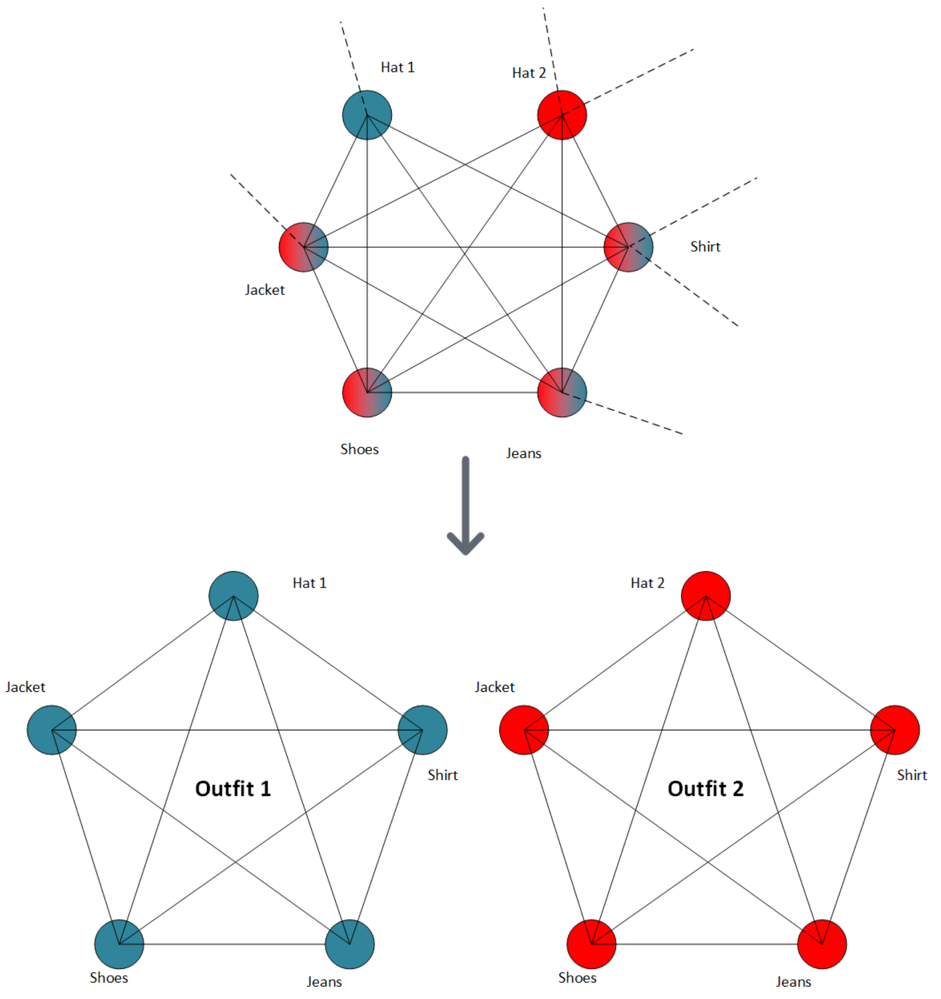

Figure 3.4: *Example of clique mining from graph*

Statistics	**Polyvore21k**	**Polyvore68k**	**Industrial**
unique items	111589	251008	33399
outfits	21889	68306	659494
items (including repetitions)	142480	365030	2308362
average (std) outfits length	6.50 (1.40)	5.35 (1.60)	3.50 (0.95)
average (std) number of outfits per item	1.28 (0.95)	1.45 (1.69)	69.11 (341.96)
sparsity	99.994%	99.998%	99.990%
number of categorical hierarchies	1	1	4
number of categories for hierarchy	380	153	15, 6, 261, 581
items appearing only once	84.9%	80.5%	4.3%
items appearing twice	9.5%	11.0%	4.9%
items appearing 3 or 4 times	4.1%	5.5.%	8.4%
items appearing 5 or more times	1.5%	3.0%	82.4%

Table 3.1: *Datasets Statistics*

Chapter 4

Related Work

In this chapter, the most relevant papers in literature will be described, starting from those that tackled the CE and FITB tasks, and continuing with those that described algorithms and techniques used in the proposed architecture.

4.1 State of the Art Baselines

As for all research projects, the first step in investigating the problem at hand has been to evaluate relevant literature and select the most promising state-of-the-art methods to be firstly reproduced and then improved upon to provide a novel solution. As previously discussed in Chapter 2, multiple solutions have been proposed to tackle the main tasks that are being actively researched in the fashion domain, so that the evaluation of both the architectures involved and the experimental results provided in the papers led to the selection of three main approaches to serve as baselines, them being the works from Han et al. [Han+17], Vasileva et al. [Vas+18] and Li et al. [Li+17].
On top of the work executed regarding the state-of-the art, in order to be able to provide a compelling work and to be sure that the complexity of the proposed solution is justified in the facts, inspired by the work from Dacrema, Cremonesi, and Jannach [DCJ19] two "shallow" (as in implemented without using any kind of Deep Learning techniques) baselines were also implemented and compared the results achieved with the different algorithms.

4.1.1 Shallow Baselines

As anticipated in the previous paragraph, implementation of simple baselines to test the proposed algorithm against has been a logical passage in the process of delivering a credible justification of the complexity of our solution. The approaches selected as comparison baselines for the fashion industry are mainly approaches derived from the idea of representing compatibility of two items using as a proxy their occurrence together in a style coherent ensemble (i.e. an outfit), stemming from the hypothesis that items that appeared together have to share at least some common style features.

Maximum Compatibility

The first of the shallow baselines implemented is a co-occurrence counting approach. With this approach, the occurrence of two distinct items in the same fashion outfit is counted for each item of the catalogue, and the resulting number of co-occurrences is used as a direct proxy for estimating compatibility of two fashion items.
It is possible to define the profile ${}^{i}p$ of an item i as a binary vector, namely:

$$\begin{aligned} &{}^{i}p \in \{0,1\}^{|\mathcal{O}|} \text{ such that:} \\ &{}^{i}p_k = \begin{cases} 1 & \text{if item } i \in O_k \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4.1}$$

where $k \in [0, |\mathcal{O}|]$ is the index of the outfit $O \in \mathcal{O}$.
Once the profile is computed in this way for all the fashion items, it is possible to compute the scores for the CE and FITB tasks as follows:

- Compatibility Estimation:

$$\hat{y}_O = tanh\left(\frac{1}{|O|(|O|-1)} \sum_{i,j \in o:\, i \neq j} {}^{i}p \cdot {}^{j}p \right) \tag{4.2}$$

- Fill In The Blank:

$$\hat{c}_{O'} = \underset{c \in C_{O'}}{argmax} \frac{1}{|O'|(|O'|+1)} \sum_{\substack{i,j \in O' \cup \{c\}: \\ i \neq j}} {}^{i}p \cdot {}^{j}p \tag{4.3}$$

The AUC for CE is computed on the $\hat{y}_O$ for all the outfits O in the testset and the Accuracy for FITB is computed on the $\hat{c}_{O'}$ for all the incomplete outfits O' in the testset.

Association Rules Mining

The second shallow benchmark implemented in the context of this thesis is based on the idea of Association Rules Mining for Market Basket Analysis, as it was introduced in [AIS93].
Since its first introduction, a large number of better optimized algorithms have been proposed by researchers to address the issue of mining frequent itemsets in very large datasets. In the context of this thesis, a python version of the ECLAT algorithm as described in Zaki [Zak00] was implemented as the core of this baseline. It is based on the analogy between an outfit and a transaction in the Association Rules terminology. An outfit is treated as a transaction of many item categories and rules between the different categories are retrieved (examples of rules are {coat, scarf, gloves} $\implies$ {hat}, {blouse, shirt} $\implies$ {skirt}).
Then, all the rules involving frequent itemsets with a confidence above certain threshold are kept. These rules, associated with their confidence, are used to score the outfits. It is possible to define an association rule as an ordered tuple of two sets, $r = (r_l, r_r) \in \mathcal{P}(cat) \times \mathcal{P}(cat)$, where cat is the set of all the item categories in the dataset and $\mathcal{P}(cat)$ is its powerset. The set of all the rules above a min_conf threshold is defined as $\mathcal{R}$. The confidence is a function $conf : \mathcal{R} \to [0, 1]$ [AIS93]. The set of the categories of the items belonging to an outfit O is defined as O_{cat}, while the category of the item i is i_{cat}. Given these definitions, it is possible to derive the function that outputs the logits for the CE task and the one that selects the missing item for FITB task:

- Compatibility Estimation:

$$\hat{y}_O = tanh\left(\sum_{s\in\mathcal{P}(O_{cat})}\sum_{\substack{r\in\mathcal{R}:\\ s=r_l\cup r_r}} conf(r)\right) \tag{4.4}$$

- Fill In The Blank:

$$\hat{c}_{O'} = \underset{c\in C_{O'}}{argmax} \sum_{s\in\mathcal{P}(O'_{cat})}\sum_{\substack{r\in\mathcal{R}:\\ s=r_l\wedge\\ \{c_{cat}\}=r_r}} conf(r) \tag{4.5}$$

In this baseline, the score of an outfit are computed from the confidence of the rules whose left-hand and right-hand side match the set of categories of the items belonging to the outfit.

4.1.2 Fully-connected layers approach

The work presented by Li et al. [Li+17] introduced the CE and the FITB tasks. The approach used to score the compatibility of an outfit is based on a modular architecture composed only of FC layers.
One module was deputed to extract visual features from the images, using the features extracted from the *fc6* layer of AlexNet [KSH12], one module composed of stacked linear layers extracted features from the text using GloVe embeddings [PSM14], while the categories where used to learn categorical embeddings. All the features extracted from these these module were then fed to a new FC layer that reduced the feature vectors' dimensions. The score of an outfit was the average of the feature vectors of its items. Then, a classifier was trained on top of this architecture to distinguish fashionable from non-fashionable outfits. The authors used popularity (i.e. the count of likes, available only on the Polyvore-21K dataset, cf. 3.1.1) associated to each outfit as a proxy of their "fashionability", namely they labeled the outfits whose likes count was above the 90^{th} percentile as popular and below the 40^{th} percentile as unpopular [Li+17]. Since this architecture relies only on FC networks, the authors had to select only outfits containing a predetermined number of items, in this case 4.
In order to tackle the FITB task, the architecture had to classify the outfits obtained by completing the partial outfit with the different choices in the choice set, and then select the one with the highest score.
It is meaningful to mention that neither the source code nor the experiments data were made available, neither upon request to the authors, and that it was not possible to reproduce the results stated in [Li+17], to the best of our ability.

4.1.3 Bidirectional LSTM Approach

One of the most promising approaches, in terms of reported performances on benchmark tasks, that could be found in literature for fashion analysis and Outfit Composition is the one based on the use of Bidirectional LSTMs to represent outfits as a learnable sequence.
As previously mentioned, the best performing solution that takes advantage of this approach, and that we decided to reproduce and test, is the one detailed by Han et al. [Han+17]; in this specific paper, outfits are considered a "sequence where each item in the outfit is a time step" in order to then be able to "train a bidirectional LSTM model to sequentially predict the next item conditioned on previous ones to learn their compatibility relationships" [Han+17].

The core of the architecture is based on a BiLSTM with 512 cells that is fed with the features extracted from the items' images. The image features are a 2048-dimensional vector derived from an Inception V3 [Sze+16] pre-trained on ImageNet, then reduced to a 512-dimensional vector [Han+17]. The items in an outfit were pre-ordered following a top-down approach (from hat to shoes, then accessories). In this way, each of the two LSTM learns to model the probability of the next item given a sequence of items. Since the LSTM is bidirectional, this works in both directions, top-down and bottom-up. Thus, given an outfit containing n items, the item at the blank position t is predicted from the sequence $(1, \ldots, t-1)$ using the forward direction and from in the sequence $(t+1, \ldots, n)$ using the backward direction of the BiLSTM.
The first step in studying this work has of course been to reproduce the results as they are exposed in Table 1 of [Han+17]; in order to do so, the authors of the paper have made public the weights that have been obtained at the end of their best performing training for the architecture, as well as the code itself implemented using Tensorflow [Aba+16] as the powering backend technology; since in the context of this thesis we have been working on a comprehensive framework, with common data structures and common data pipelines (more on this will be comprehensively presented in Section 5.1) all implemented using PyTorch [Ket17] as a backend tecnology, in order to be able to incorporate the baseline into the above mentioned framework in a faster and easier way, the reproduction of the results was achieved by adapting the PyTorch porting of the original codebase available on GitHub[1]. On the architecture, the results were reproduced by way of loading the provided pre-trained weights on top of the implemented model.

4.1.4 Type-Aware Embeddings

Another approach that was validated in the context of reproducing baseline results is the one outlined in the work from Vasileva et al. [Vas+18]; in this second approach, the founding idea behind the research is that "A representation for building outfits requires a method that can learn both notions of similarity...and compatibility..." so that what is presented in the paper is an "...approach to learning an image embedding that respects item type, and jointly learns notions of item similarity and compatibility in an end-to-end model". The authors clearly distinguish between item-similarity and item-compatibility.
The process on which this model is based takes advantage of the assumption

[1]codebase: github.com/arubior/bilstm

that compatibility in fashion is a property that doesn't appear to be intrinsically transitive, and the approach to force embedding for items to be close in a general shared space severely limits a compatibility model by basing its scoring system on a property (being close in that same space) that may not hold, since different categories may need different dimensions and latent spaces to be properly represented. The aim of making up for this limitation and leveraging the full informative power of the compatibility training, without facing the risk of "...encouraging the creation of *improper triangles*", led to the main innovative solution presented in this work.
Instead of just learning the embedding of each item of the dataset in a common shared space, a first embedding space is firstly created by means of using the visual features extracted from a CNN (in this specific work, the one used is a slightly modified version of the ResNet18 network [He+16]) and features representing the textual description of the item via a visual-semantic loss; as a second, further, step the authors use a "learned projection which maps the general embedding to a secondary embedding space that scores compatibility between two item types" and the embeddings are then used together with a generalized distance metric, in order to compute compatibility scores between items.
The fundamental assumption on which this work is based is that the outfit compatibility can be replaced by an item-level proxy: instead of processing an outfit as a whole, Vasileva et al. computed the compatibility between all the pairs of items contained in an outfit and then averaged such score. Each item category pairs (e.g. *(shoes,hat)*, *(jeans,sweater)*, *(shoes, jeans)*) was assigned a space where to compute the compatibility between the items belonging to such categories. This approach allows the model to learn several different compatibility metrics depending on the items category, but, on the other hand, increases the time complexity of the algorithm (in an outfit containing n items, there are $O(n^2)$ pairs) and loses the relationships relating the different couples in an outfit and all the other subsets of elements present in an outfit.

4.2 Cross-Domain technologies

As Machine Learning analysis in fashion compositions remains a relatively young field for research, numerous approaches that could give a significant contribution in both performances and generality of the tasks carried out remain untested, and there is still a huge space for improvement of existing techniques.
In the context of this thesis work, the main focus has been to develop a novel

architecture to tackle the numerous challenges present in the space of fashion outfit composition and compatibility estimation; the main drivers behind the choice of the building blocks composing the proposed architecture have been two:

- Improve state-of-the-art performances for the tasks commonly tackled by literature
- Provide a general enough solution to be able to tackle the problem of unconstrained outfit generation

In order to achieve the above mentioned goals, it was decided to incorporate into the proposed architecture two main techniques taken directly from research in other field, such as NLP.

4.2.1 Attention Mechanism

The Attention Mechanism is a concept first laid out in the work from Bahdanau, Cho, and Bengio [BCB14], which tries to relax the assumption that in an encoder RNNs the final state holds information about the whole sequence seen so far; instead, the approach suggests that a decoder should look at the RNN's hidden state at each time step and produce the corresponding output using all the encoder's hidden states and the decoder's hidden states computed so far. In this way, it is possible to avoid the long-term dependency problem that afflict recurrent networks while learning the degree of "attention" that should be assigned to each of the hidden states. The attention function, in its broad definition, is a dynamic weighted sum of some vectors, where dynamic means that the weights change depending on some variable (usually, related to the input positions and/or elements). In the sequence-to-sequence domain, the weights depend on the position of the input and output element in their corresponding sequences, i.e. the weight w_{ij} determines how well output elements around position j match the input element around position i in the respective target and source sequences[BCB14]. The are many different versions of the actual function that shapes the weights, depending on the different use cases [Cha+19].
There has been a first try to implement this kind of approach in the context of fashion visual analysis, precisely in the work of Wang et al. [Wan+18], where attention mechanism has been used together with an externally defined grammar in order to better understand the regions of a fashion item that are most relevant to define its category, i.e. solving a multi-label multi-class classification task.

4.2.2 Transformer

The Transformer is a particular sequence-to-sequence NN presented for the first time by Vaswani et al. [Vas+17]. This net architecture relies on an encoder-decoder structure, where both parts are composed of a sequence of self-attention layers interleaved with FC layers. Self-attention layers allow the net to learn the relationships between the different elements of a single sequence. This network went beyond the state-of-the-art in several NLP tasks [Vas+17] and proved to be the new reference architecture in sequence-to-sequence mapping.
The encoder is composed of a stack of 6 identical layers, each composed of one self-attention mechanism sub-layer and one FC sub-layer, both with residual connections and layer normalization. The input of the encoder is a sequence of vectors (representing lexical tokens in NLP domain). The vectors flow from one sub-layer to the next and from one layer to the next. The decoder has a similar 6-layers structure, but each layer has two subsequent attention mechanism and one final FC layer. The first attention mechanism is a self-attention one fed with the target sequence (i.e. the ground truth) shifted by one position, while the second one is fed with the output of the first mechanism and the output of the corresponding encoding layer. The elements in the target sequence can attend elements in their own sequence only if such elements precede them, while the connections to the others are masked. This is necessary in order to create an autoregressive model, i.e. the prediction of the next element in a sequence cannot depend on the following ones 4.1.

In their work, [Vas+17] used multi-head version of the scaled dot-product attention. The scaled dot-product is defined as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.6}$$

where $Q \in \mathbb{R}^{s \times d_k}$, $K \in \mathbb{R}^{t \times d_k}$, $V \in \mathbb{R}^{t \times d_v}$ and d_k, d_v are hyperparameters defining respectively the size of the source and target sequences hidden spaces, while s, t are the source and target sequences' lengths. The $softmax$ shapes the weights based on how much the encoding (i.e. hidden state) of the elements in the target sequence (Q) match the encoding (i.e. hidden state) of those in the source sequence (K). The match is defined by a dot-product, scaled by the dimension of the source embedding, in order to avoid the product to grow large and end up in the region where the $softmax$ gradients are small. V is a matrix of learned embeddings for each element of the target sequence. The multi-head version of this attention mechanism

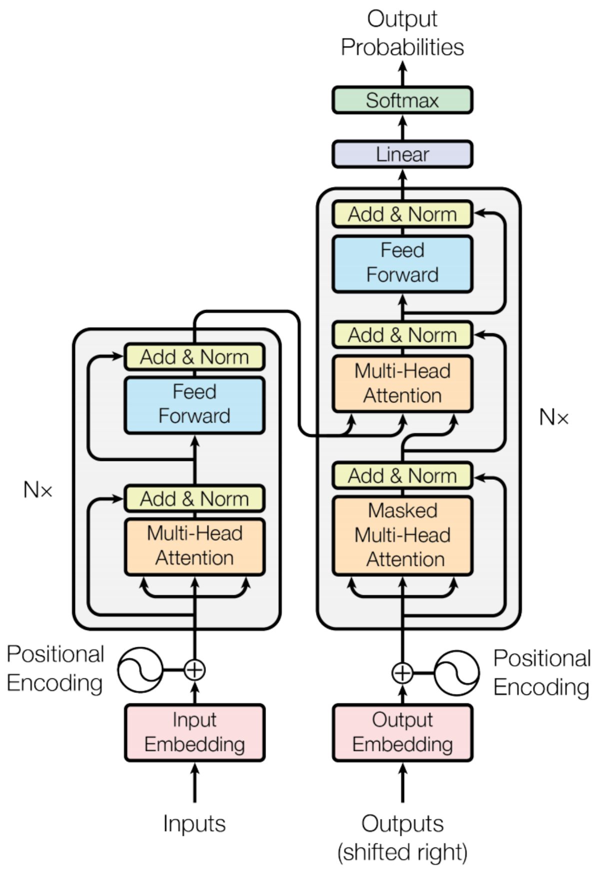

Figure 4.1: *Transformer architecture (1 layer) [Vas+17]*

projects the embeddings into h different subspaces, making the model capable of learning from different representation subspaces in parallel instead of learning just one space.

$$\begin{gathered} MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \\ \text{where} \quad head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \end{gathered} \tag{4.7}$$

and W_i^Q, W_i^K, W_i^V, W^O are the learned matrices that project the embeddings into the different subspaces and back to a common space [Vas+17].
In the encoder and in the first part of the decoder, the self-attention mechanism is used, and thus all three Q, K, V are the embeddings of the elements of the same sequence, respectively the source and the targets, letting the model learn the relationships between elements of the same sequence. In the decoder's self-attention, each element can attend only those preceding it, maintaining the autoregressive property; in the second mechanism in the

decoder layers, on the other hand, Q comes from the embedding of target sequence, while K, V come from the embedding learned by the respective encoder layer, letting the model learn the relationships between target and source sequence.
As shown in Fig 4.1, the Transformer is agnostic about the position of the elements in the sequence and learns their match just based on their features content [Cha+19]. This is the reason why Vaswani et al. had to add positional embeddings to each token embedding, since in the NLP domain the position of a token or word in a sentence is a relevant information that should not be lost.

4.2.3 Set Transformer

Most of the work with NN deals either with fixed-size inputs or with variable-size sequential inputs. None of these structures can properly represent sets, which are neither fixed in size nor they impose an order on their elements. The problem of processing sets with NN was first addressed by Zaheer et al. [Zah+17]. The authors claimed that any function f acting on an input set must be permutation-invariant in the order of the elements in order to be a valid set function, i.e. it must respect the following property:

$$\forall \pi : f(\{x_1, \dots, x_M\}) = f(\{x_{\pi(1)}, \dots, x_{\pi(M)}\}) \tag{4.8}$$

where $\pi : [1..M] \to [1..M]$ is a permutation, namely a bijection from the indices to themselves. Zaheer et al. proved that a function f meets the property in (4.8), *if and only if* it can be decomposed in the form:

$$\rho(\sum_{x \in X} \phi(x)) \tag{4.9}$$

where X is a finite set, for suitable transformations ρ and ϕ. ρ and ϕ can be learned by properly-shaped FFN since they are universal approximators [Zah+17].
Moreover, in a trasductive setting, each element of a set is associated with a label (or vector) [Zah+17]. A permutation-equivariant mapping $\mathbf{f}$ must respect the following property:

$$\forall \pi : \mathbf{f}([x_{\pi(1)}, \dots, x_{\pi(M)}]) = [f_{\pi(1)}(\mathbf{x}), \dots, f_{\pi(M)}(\mathbf{x})] \tag{4.10}$$

where $\mathbf{x}$ is the concatenation of the elements in X. Namely, $\mathbf{f}$ is permutation-equivariant if it permutes the output labels (or vectors) upon permutation of the input elements.
Lee et al. [Lee+19] proposed an architecture, the Set Transformer based

on the same multi-head attention described in [Vas+17] in order to encode-decode sets. They proposed to model the ϕ function in (4.9) with an encoder based on a sequence of transformer encoder layers. The authors call each encoder layer SAB (Self-Attention Block) [Lee+19], defined exactly as in [Vas+17] as follows:

$$\begin{aligned} SAB(X) &= LayerNorm(H + FF(H)) \\ \text{where } H &= LayerNorm(X + MultiHead(X, X, X)) \end{aligned} \tag{4.11}$$

and FF is a feed-forward network. The SAB is permutation-equivariant and so is the whole encoder [Lee+19]. The function $\rho(\sum(\cdot))$ in (4.9) is modelled with a sequence of SAB and FFN, where the summation is represented by slightly different version of SAB having $H = LayerNorm(S + MultiHead(S, Z, Z))$ where S is a learnable matrix and Z is the matrix containing the embeddings of the set elements. Lee et al. proved that, given enough nodes, this architecture can approximate the permutation-invariant function in (4.9).

4.2.4 BERT and RoBERTa

Devlin et al. [Dev+18] introduced a novel architecture to tackle different NLP tasks. It is based on the transformer encoder by Vaswani et al., on top of which multiple different FFN (heads) can be placed in a parallel way. The training is divided in two phases: pre-training and fine-tuning. During the pre-training, the transformer is trained to solve two different tasks (Masked Language Modelling, MLM, ans Next Sentence Prediction, NSP) simultaneously; while, during the fine-tuning, its parameters are adjusted to solve specific NLP tasks, one per head. The model input is composed of two sequences (each containing many natural sentences) of tokens embeddings, including some special tokens, such as:

- [CLS], one per sequence, that holds the representation embedding of the whole sequence for sentence classification purposes;
- [SEP], that delimits when a sentence ends and another one starts inside a sequence;
- [MASK], used to replace some tokens for the MLM task.

The [CLS] token can attend to all the tokens in a sequence and is supposed to aggregate the representation of the corresponding sentences. It is then used to solve sentence-level classification tasks. The [SEP] token is used to

delimit two segments such that the model can learn how the sentences in a sequence are related, in order to solve the NSP task (i.e. connecting two following sentences extracted from a larger text, such as a question and the respective answer). The [MASK] is used to replace one random token (15% of all tokens) and in the MLM task the goal is to predict the features of the the original token that was masked by attending the features of the other tokens in the sentence. In BERT, all the tokens can attend each other, not only the previous ones placed leftward, hence the name "bidirectional".
RoBERTa [Liu+19] is an improvement of the BERT model, in which the NSP task is removed, the data are augmented so that each sentence is masked differently in the different epochs, hyperparameters are fine-tuned differently. In both cases, BERT and RoBERTa, they created an encoder-only model using the Transformer, and then stacked some FFN specific to each different finetuning task.

Chapter 5

Solution

5.1 Proposed Solution

While carrying out the study and evaluation of the state-of-the-art techniques available, what could be concluded as a general observation about the current state of the proposed solutions can be summarized in two main facts:

- Performance in the two main tasks that are used as benchmark for the fashion domain rose significantly from first attempts, but the benchmark tasks themselves are structured in a very constrained way.
- There is a lack of experimental results, in literature, for an industry relevant benchmark task about outfit generation.

As a consequence of this two facts, in the context of this thesis work the focus has been to provide a novel architecture able to offer performances in line with the state-of-the-art techniques for the benchmark tasks as they are usually carried out in literature, as well as being able, at the same time, to provide industry acceptable performances in an unconstrained version of the outfit generation tasks that makes it viable to be used in an industrial context.

5.1.1 Architecture Description and Design

Most of the main works in literature that tackled CE and FITB tasks [Li+17; Vas+18], started from designing a classifier able to solve the compatibility task and then reused it to solve the other one. The common approach to the FITB task was based on the idea to add each item in the set of proposed answers to the incomplete outfit separately, thus creating one different outfit

for each different item in the answers' set. Then, each of these new outfits was classified (binary classification, with "compatible" class as positive and "non-compatible" as negative). Since the only difference between these outfits was the item coming from the answers set, the answer composing the outfit with the highest score (the probability of being "compatible") was chosen as the predicted missing item. This approach requires creating $|S|$ outfits and $|S|$ evaluations, thus making it unfeasible for a more general task, such as the unconstrained outfit generation introduced here in Section 2.1.3, where $|S|$ is the number of items in the dataset belonging to a specific category (≈ 1640 in the Polyvore-68k dataset, ≈ 60 in the industrial dataset).
Another problem faced in literature was the way to deal with outfits of different lengths. Outfits can be considered as sets of unordered items, but dealing with such variable structures is not easy in the domain of neural networks, since feed-forward neural networks require fixed-dimensions inputs and recurrent neural networks impose an order to the items they process. Li et al. worked around this problem by fixing the size of the outfits, i.e. removing all outfits containing less than 4 items and removing the exceeding items from the outfits longer than 4 items. Han et al. used a BI-LSTM with 8-layers, inducing a specific order and specific composition of the outfits, that are treated as ordered 8-items-long sequences, where each position in the sequence matched a specific super-category (i.e hat, top, bottom, shoes, accessories ...). Vasileva et al. moved the problem from the outfits space to the item space: they trained a model able to score an item-item compatibility, based on the idea that items that appear in the same outfits are compatible. In order to evaluate the compatibility of an outfit, they averaged the item-item compatibility of all the items in an outfit, increasing the time complexity of the algorithm (in [Li+17; Han+17] the score of an outfit required the analysis of $|O|$ items, while [Vas+18] required $\frac{|O|\times(|O|-1)}{2}$ couples of items to be checked).
In contrast to these approaches, the proposed architecture tackles both tasks (CE and FITB) together at the same time, can process sets of any cardinality and can generate the features embedding of the missing items.
The algorithm is composed of two parts: the first one is tasked with extracting the relevant features from the data, while the second part is responsible for generating the features of the predicted missing item of the outfit. Similarly to [Li+17], the first part of the algorithm (Fig. 5.1) extracts features from the textual, visual and categorical data available for each item. The textual features are extracted using a pretrained general-purpose vocabulary (one vector per token) [PSM14], which contains 42 billion 300-dimensional vectors for the Polyvore-21k and the industrial dataset,

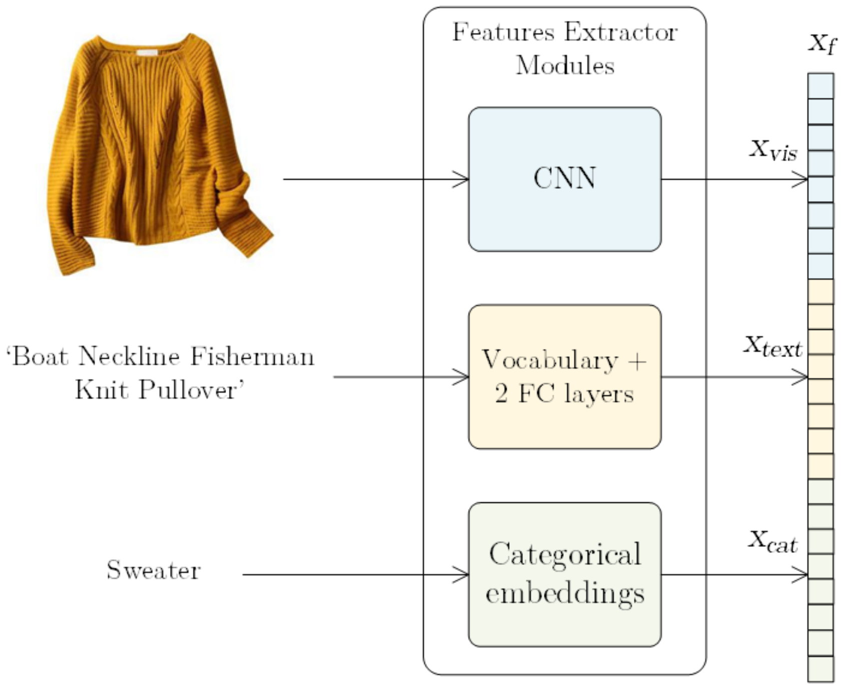

Figure 5.1: *Structure of the first part of the proposed architecture*

and a specific pretrained 6000-dimensional vocabulary (used and published in [Vas+18]) for the Polyvore-68k, containing one vector per each sentence in the dataset. The vectors of the first vocabulary are averaged for each word in the textual description of the items. The resulting vectors are then fed to a fully-connected layer with ReLU activation that reduces the features number to 64 ($|x_{text}| = 64$). The categorical data are used as lookup indices for embedding vectors sized 64. For the industrial dataset, that contains 4 category groups, 4 corresponding embedding matrices are learned ($|x_{cat}| = 64 \times cat\_groups = 256$). The visual features are learned by the same ResNet-18 version used in [Vas+18] for fair comparison. The visual features vectors are sized 512 ($|x_{vis}| = 512$), as in the best model version reported in [Vas+18].
The complete features vector for an item is the concatenation of the three vectors described above:

$$x_f = [x_{vis}, x_{text}, x_{cat}] \tag{5.1}$$

The items belonging to an outfit pass through this first stage of the network all together, so that their embeddings can be used to feed the second part without losing the notion of co-occurrence within the same outfit, as in Fig. 5.2. The second part of the algorithm takes advantage of the Set

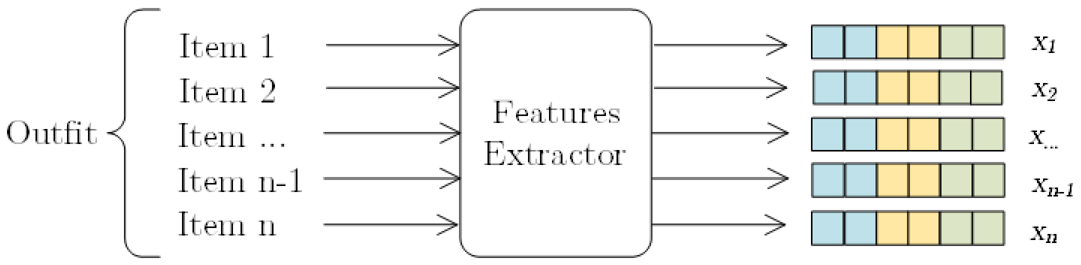

Figure 5.2: *Features extraction from items belonging to the same outfit*

Transformer and the RoBERTa training described in Sections 4.2.3, 4.2.4. In [Liu+19; Dev+18], approximately one token per sentence was replaced with a [MASK] token, whose feature were learned during the training. Similarly, in the fashion domain, one item of the outfit is hidden, i.e. its features are replaced by the feature-wise average of the other items, as depicted in Fig. 5.3. Moreover, a fictitious item, representing the embedding of the outfit, is added to the items composing an outfit. This fictitious item is the equivalent of the [CLS] token used in [Liu+19; Dev+18], and is responsible for attending all the other items and collect their most important features during its process through the transformer layers. This fictitious item is initialized as the feature-wise average of all the items in the outfit. Using the averages of all the items in the outfit as a starting vector for the fictitious item instead of learning a "neutral" embedding for the [MASK] token (as in [Liu+19]), empirically proved to give a boost to the algorithm performances. This is also supported by the fact that in [Vas+18] the average of the item-item compatibility was still an acceptable proxy for the score of the whole outfit compatibility. The same idea applies to the hidden item features, even if the features of the item to be predicted are obviously excluded from this other average. Both this vectors are masked to the other items, i.e. they can attend the other clothes in the outfit but the other clothes cannot attend them and neither they can attend each other.

The use of the transformer's architecture brings along some relevant design improvements:

- this solution removes any constraint on the outfits size (differently from [Li+17]);

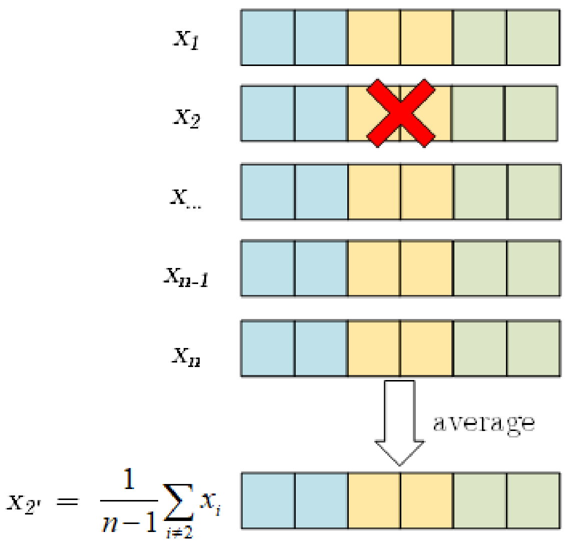

Figure 5.3: *Masking of an outfit (item 2 in this example)*

- it removes any constraint on the items order inside an outfit: the outfit is treated as an orderless set, the learning is focused only on the content of the items belonging to an outfit and not on the direction in which they are processed (differently from [Han+17]);
- it also processes the outfit as a whole: it tries to model the stylistic relationships at an outfit-level, rather than item-level (differently from [Vas+18]).

The transformer version used in this thesis is the Set Transformer encoder introduced in [Lee+19] and described in Section 4.2.3, that is permutation-equivariant and models the high-order interactions between the different subsets of elements contained in an outfit. Namely, a single encoding layer, models the relationships between items, stacking two encoding layers, models the relationship between couples of items, stacking three encoding layers, models the relationship between triplets of items, and so on [Lee+19]. This characteristics make the transformer set suitable for the fashion domain, where the stylistic relationship between the items depends on their features content grouped in all the possible subsets belonging to the powerset of an outfit, rather than the sequence of the items. While in [Han+17], given an outfit

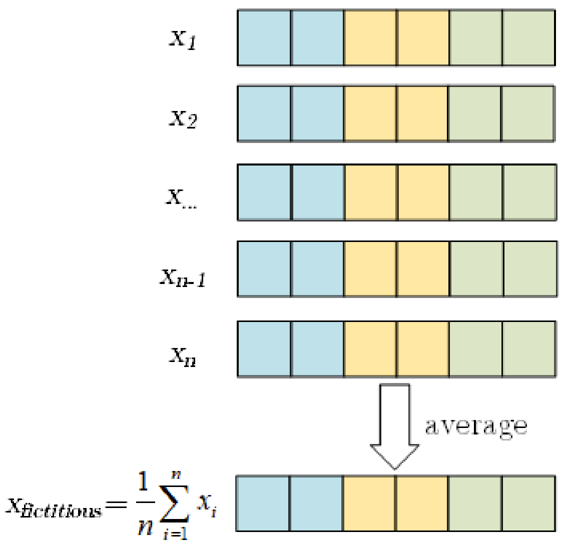

Figure 5.4: *Creation of a fictitious item*

containing n items, the score of the missing item in position t depends on the sequence $(1, \ldots, t-1)$ and on the sequence $(t+1, \ldots n)$, in the transformer the prediction depends on all the subsets in $\mathcal{P}(O^{'})$, as detailed in Eqn **??**, **??** and described in [Zah+17].

5.1.2 Training and Deployment

As mentioned, the whole architecture can be considered as composed of two parts: the first modular part that extracts features from the different input modalities and the second part dedicated to the inference of the outfit compatibility and the prediction of the missing item. However, the whole architecture is trained end-to-end.
The weights of the modules deputed to extract the features are initialized with those of pretrained models (similarly to what was done in all the other papers in literature [Li+17; Han+17; Vas+18]), and then they are fine-tuned during the training of the whole architecture. The CNN is pretrained on ImageNet, while the vocabularies, glove and the specific one for Polyvore68k, are pretrained on general web text and on the text descriptions of the items in Polyvore68k, respectively. While the weights of the CNN are updated

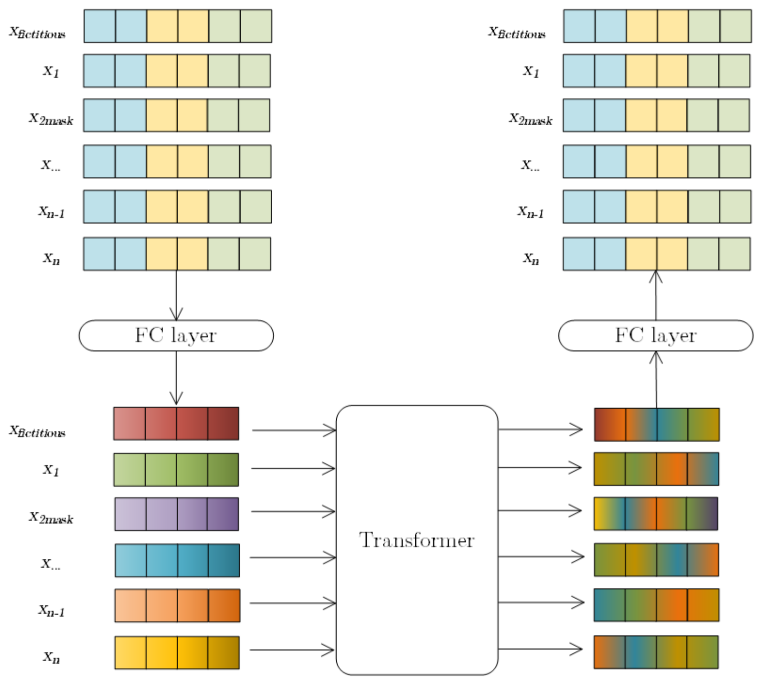

Figure 5.5: *Second stage of the architecture*

during the training, the vocabularies embeddings are not updated, but they have a 2-layer FC network on top, whose weights are learned at training time.
The second part of the module is initialized with Xavier initialization, as in [Vas+17]. The pre-training tasks on which BERT is trained (cf. Section 5.1.1, [Dev+18]) are conceptually similar to the CE and FITB (classification of sentences (outfits) and prediction of a missing items (words/tokens) from a sentence (outfit)). Anyhow, even if the structure of this part takes advantage of the transformer, and the tasks are conceptually similar to those faced by BERT and RoBERTa [Liu+19], it is necessary to point out some key differences in the training process of the architectures, due to the different domains:

- BERT is pretrained on such NSP and MLM tasks (RoBERTa only on MLM), but then is finetuned on many diferrent NLP tasks, the proposed algorithm is only deputed to solve CE and FITB tasks;

- BERT and RoBERTa are pretrained on a single positive class, i.e. real sentences, the proposed algorithm on positive and negative samples;

- BERT and RoBERTa have a much larger dataset at their disposal, and each single token/word appears many times in the dataset, while the fashion datasets are much sparser (especially Polyvore versions);
- in NLP the features of a token can be extracted just from the co-occurrence of the other tokens in the same sentences due to the abundance of appearances of the same word in many different sentences, in the fashion domain the features are extracted from the raw data defining clothes (image, description, category);
- BERT and RoBERTA fail to extract features from words never seen during training, the proposed architecture can extract features from the raw data, even if they belong to an item never seen at training time;
- the training process is much longer in the NLP domain than in the fashion domain due to the different raw data and amaunt of data described before ($\approx$ 3 days, using 8 GPU for BERT and RoBERTa vs $\approx$ 12 hours on one GPU of the same type);

The reference dataset used for the development of the algorithm is the non-disjoint version of the Polyvore68k. The hyperparameter used for traininig on such dataset are shown in Tab. 5.1. In order to contrast overfitting, we adopt early-stopping during the training and validation process and dropout in both stages of the net.
The whole implementation relies on Pytorch v1.2.0+cu10 [Ket17] and was deployed on a machine with one GPU Nvidia Tesla V100.

Hyperparameter	value
learning rate	10^{-4}
dropout	0.1
learning rate decay	0.1
category embedding dimensions	64
text embedding dimensions	64
visual features embedding dimension	512
number transformer layers	4
batch size (number of positive outfits sampled per batch)	4
negative outfits created per positive outfit	5
loss weight λ	0.75

Table 5.1: *Hyperparameters summary table*

Training on Polyvore

As already mentioned in Section 3.1, both Polyvore datasets suffer from severe sparsity. Most of the items are seen only once, thus it is very difficult to infer how many different "styles" they would fit, and especially what items they would not fit with. It is possible, anyway, to create negative outfits, by replacing each item in an outfit with another one belonging to the same category. This approach, already used by [Vas+18], can be apllied multiple times to create many different negative outfit from one real outfit. This becomes necessary, given the very coarse granularity of Polyvore categories. Sampling many negative items for each positive one allows the model to separate the items belonging to the real outfits from the feature-wise average of the respective category-aware negative item. In case of Polyvore, the best results were achieved with a sampling factor $r = 5$, i.e. sampling 5 outfits composed of random items for each real outfit.
Moreover, for each positive outfit O, $|O|$ incomplete outfits were created by masking every time a different item in O.

Training on industrial dataset

On the other hand, in the industrial dataset there are many more outfits and each item appears in many different outfits, thus it is possible to infer many possible stylistic relationships. So, while there was still a need to sample many negatives for each positive, the need to create $|O|$ different incomplete outfits became not so relevant, due to the abundance of outfits and item repetitions. During the training in the industrial dataset, only one random item was masked in each outfit for the FITB task.

5.1.3 Loss

The loss is composed of two parts, the first one accounting for the CE task and the second one accounting for the FITB task. The former part, is a simple binary cross-entropy loss, typically used in binary classification tasks:

$$L_{CE} = \frac{1}{|\mathcal{O}|} \sum_{O \in \mathcal{O}} t_O \cdot \log y_O + (1 - t_O) \cdot \log(1 - y_O) \tag{5.2}$$

where t_O is the binary target label of outfit O, 1 for real outfits and 0 for negative ones (cf. Section 5.1.2 to see how negative outfits are created) and y_O is the predicted classification score of the outfit O. The score y_O is computed from a 2-layer FNN that takes as input the "fictitious" item inserted in the transformer that attended all the other items in the outfit.

The second part of the loss is defined following the idea that the model should project different items in the same space by moving closer those items that appear in the same real outfits and pushing further away those that appear together only in the negative ones. Thus, the model should be penalized based on the distance of the generated embeddings, in order to encourage items clustering in this new space. This part of the loss is defined as:

$$L_{FITB} = \frac{1}{|\mathcal{O}|} \sum_{O \in \mathcal{O}} \frac{1}{|O|} \sum_{\substack{i \in \mathcal{O}: \\ i \neq i'}} TripletLoss(\hat{x}, i, i^{-}, m) \tag{5.3}$$

where

$$\begin{aligned} TripletLoss(a, p, n, m) = {} & \max(\|a, p\| - \|a, n\| + m, 0) \\ & + \max(\|a, p\| - \|p, n\| + m, 0) \end{aligned} \tag{5.4}$$

where $i^{'}$ is the masked item whose features should be predicted by the model based on the features of the other items, i^{-} is a randomly sampled item of the same category of $i^{'}$, $\hat{x}$ is the vector of features of the missing item predicted by the model, m is the margin, a hyperparameter modelling the distance between the embeddings. The triplet loss is the same used by Vasileva et al., but in this case, it is used to push the learned embeddings closer to the embeddings of the other items present in the outfit and push a random item of the same category of the missing one away from the outfit's items and from the predicted one. Acting at the outfit-level, it helps retaining the notion of outfit, that gets lost when used in a normal item-level triplet as in [Vas+18]. Moreover, adding the swap (i.e. replacing the anchor a with the positive element p) improves the performances [VM16]).
The selection of the negative sample i^{-} has a great impact on the learning process of the model. Since in all the available datasets there is no hint about what items are incompatible with each other or what items would not fit in an outfit, the assumption is that a random item would be less compatible in an outfit if compared to the one that was present in the real outfit. This assumption is the same made by Han et al. and Vasileva et al. However, this implies that all the items not appearing in an outfit are *equally* incompatible with those present inside such outfit, but most likely there are multiple *levels* of incompatibility, of which there is no proper representation in the dataset. In such approach, there is a clear mismatch between the confidence of positive relationships and the negative ones and since the L_{FITB} depends both on the items present in an outfit and in a random negative, the gradient that shapes the weights of the projections made by the architecture would be highly influenced towards a wrong direction if the random negative item

is *not so incompatible*. In order to reduce this variance in the selection of the negative items, the sampling of i^- is repeated multiple times, controlled by the r hyperparameter mentioned in 5.1.2. By doing this, the gradient would reshape the projection matrices in order to consider the features that most differentiate the positive one from the average features of the group of negatives, assuming that these average features are closer to the representation of a general item belonging to the category of $i^{'}$ and i^-. Therefore, the new L_{FITB} becomes:

$$L_{FITB} = \frac{1}{|\mathcal{O}|} \sum_{O \in \mathcal{O}} \frac{1}{|O|} \sum_{\substack{i \in \mathcal{O}: \\ i \neq i^{'}}} \frac{1}{r} \sum_{k=1}^{r} TripletLoss(\hat{x}, i, i_k^-, m) \tag{5.5}$$

The overall loss function is the weighted sum of the two, so that the model can learn how to cluster items that fit well together and to score such clusters:

$$Loss = (1 - \lambda) L_{CE} + \lambda L_{FITB} \tag{5.6}$$

where λ is an hyperparameter.
During the conducted experiment, it became clear that the L_{CE} and L_{FITB} have relatively close minima, but at some point the gradient moves the weights of the algorithm towards the minimum of the loss that has the highest relative weight, diverging from the minimums of the other loss, slightly improving the performances on one task but severely worsening the performances on the other task at validation time. Swapping λ with $1-\lambda$ at the end of every epoch empirically proved to keep high performances on both tasks simultaneously at validation time. The early-stopping technique compares the performances on AUC for CE and Acc for FITB with those obtained at the previous epoch and stops whenever one of the two decreases. After each epoch the learning rate was decayed by a factor 0.1.

Chapter 6

Results & Discussion

In this chapter, an overview of the resulting performances for all the tested algorithms and approaches is going to be presented; results are then going to be analyzed and discussed, in order to provide insight and interpretation over the obtained performances, compared to previous solutions and available state of the art techniques.
Furthermore, for the novel task of Unconstrained Outfit Generation (cf. Section 2.1.3) the first ever experimental results are going to be presented and discussed.

6.1 Comparative Table of Results

In this section, a comparative overview of the best performance for each model is going to be provided; best performances are going to be evaluated on the Polyvore 68k - NonDisjoint dataset split.
This particular split has been chosen to serve as summary for the general performances of the architectures taken into consideration in order to keep the evaluations consistent with the works carried out in the field recently.
The table below shows performances of the best scoring model for each of the approaches mentioned:

Polyvore 68k Results		
Algorithm	CE - AUC	FITB - Acc
Bi-LSTM	0,65	0,379
TAE	0,86	0,562
OutfitTransfomer (ours)	**0,91**	**0,625**

Table 6.1: *Polyvore 68k Results*

As shown in the above table, on the dataset split that is commonly used as a reference point for performances evaluation on benchmark tasks, this thesis proposed solution is able to significantly outperform the current state of the art architecture in both tasks, providing a 5,91% increase in performance for the Compatibility Estimation task and a 11,21% increase on Fill In The Blank task.

6.2 Shallow Baselines Results

This section is dedicated to the evaluation of performance for two non-deep approaches that have been used as baselines for confrontation.
Experimental results show that, for all the metrics of both benchmark tasks as described in Section 2.2, shallow baselines that are able to leverage only the categorical features in the dataset never achieve performances that are decisively above random.
The first and most relevant indication that we can infer from those results is the confirmation that, in a domain that is heavily influenced by visual compatibility of items such as the fashion domain, not being able to leverage a measure of visual representation for the tasks here described proves to be too big of a limitations for any shallow algorithm to effectively perform. This conclusion points us to the necessity of leveraging a deep-learning based architecture in order to make use of the visual representation of each single item in the available catalogue.
Stemming from this conclusion we directed our efforts towards those models that could have been better suited to fully exploit the hidden information deriving from extracted visual features, and couple them together with the information inferred from both textual and categorical features.

6.3 Benchmark Tasks

This section is dedicated to the presentation and comparative evaluation of performances for all deep models used in this research work. Results are going to be presented for each dataset and split available, in order to give an intuitive idea of the comparative performance for each approach.
The models that have been re-implemented to serve as baseline confrontation in the context of this thesis work have been the following:

- Bi-LSTM approach as presented in 4.1.3 from Han et al. [Han+17]
- Type-Aware Embedding (TAE) model as presented in 4.1.4 from Vasileva et al. [Vas+18]

- Fully connected approach as presented in 4.1.2 from Li et al. [Li+17]

The proposed solution network for this research has been named Outfit-Transfomer. It is necessary to note that, although it has been tried, it was not possible to reproduce results presented in Section 4.1.2 as they have been reported in [Li+17]; as mentioned in Section 4.1.2 the available description from the paper was lacking too many details to be properly implemented, the dataset was missing and the authors, although contacted, failed to provide any working version of their codebase.

6.3.1 Polyvore 21k Results

In this subsection, results are presented for the clean version of the Polyvore21k dataset, as described in 3.1.1:

Polyvore 21k Results		
Algorithm	CE - AUC	FITB - Acc
Bi-LSTM	0,94	0,649
TAE	0,93	**0,65**
OutfitTransfomer (ours)	**0,94**	0,60

Table 6.2: *Polyvore 21k Results*

As we can see from above results, on this specific dataset the proposed architecture performs in a comparable way on the CE task with state of the art results, but fails to reach the same performances on the more complex FITB task.
That distance in performance can be explained by three main hypothesis:

- Since the proposed architecture is more powerful and more complex than previous state of the art approaches, it suffers more the sensible reduction in available data for training
- In this particular version of the dataset, it is not possible to use super-categories, which are integral to our sampling method during training
- Since this version wasn't the main dataset used for development of the solution, using the same hyperparameters may not be a viable solution when dealing with this dataset

6.3.2 Polyvore 68k - Disjoint Results

This subsection is dedicated to the presentation and discussion of results obtained on the Disjoint version of Polyvore as described in Section 3.1.2:

Polyvore 68k - Disjoint Results		
Algorithm	CE - AUC	FITB - Acc
Bi-LSTM	0,62	0,394
TAE	0,84	0,552
OutfitTransfomer (ours)	**0,903**	**0,645**

Table 6.3: *Polyvore 68k - Disjoint Results*

The first thing that can be pointed out about results shown in the above table is that, on this different construction of the dataset, the solution proposed in this research work is able to consistently outperform all available state of the art approaches on both the benchmark tasks analyzed. It is interesting to notice how, on this more complex construction of the public dataset, two architectures that performed in very similar ways on a simpler dataset (as shown in Section 6.3.1) demonstrate a very large difference in performance. This disjoint version of the dataset is also a proxy to evaluate the generalization capabilities of the networks, and experimental results shown here, even more if compared with results shown in Section 6.1, suggest that the proposed solution not only outperforms other state of the art approaches on this dataset split, but is able to better generalize knowledge learnt at training.

6.3.3 Polyvore 68k Results

This subsection is dedicated to the presentation and discussion on results obtained on the NonDisjoint version of Polyvore dataset, as described in Section 3.1.2. As already mentioned before in Section 6.1, this particular split of the public dataset has been used as the main proxy to evaluate performance of algorithms by the majority of state of the art works in literature.

Polyvore 68k Results		
Algorithm	CE - AUC	FITB - Acc
Bi-LSTM	0,65	0,379
TAE	0,86	0,562
OutfitTransfomer (ours)	**0,91**	**0,625**

Table 6.4: *Polyvore 68k Results*

Sampling Variance Study

In order to have the fairest possible comparison, on the clean version of the Polyvore 68k - NonDisjoint split of the dataset, we also decided to compare results with the approach from [Vas+18] (the only one that leverages a sampling method similar to the one in this thesis proposed solution) by looking at performances of both models while varying the quantity of negative samples selected for the loss function for each iteration.
Results for this study are presented below. In Table 6.5, r is the same parameter described in Section 5.1.2 and indicates how many negative samples (i.e . randomly generated ‘fake‘ outfits) are created for each positive outfit.

Polyvore 68k Results						
Algorithm	r=1		r=3		r=5	
	CE - AUC	FITB - Acc	CE - AUC	FITB - Acc	CE - AUC	FITB - Acc
TAE	0,86	0,562	0,786	0,482	0,785	0,478
OutfitTransfomer	0,85	0,545	0,903	0,620	**0,91**	**0,625**

Table 6.5: *Polyvore 68k - Sampling Variance Results*

Results collected in the above table for our Sampling Variance study show that, by augmenting the availability of data points during the training of the networks, the proposed novel methodology is able to significantly outperform the current state of the art approach. What can be inferred by the fact that, with the progression of data augmentation, performances of the state of the art tend to degrade while performance of the proposed solution keep improving is that the novel architecture proposed is able to better generalize the concept of outfit, while augmenting the samples in the algorithm by Vasileva et al. probably overfits on the item triplets, losing the notion of outfit.
The analysis of above stated results, together with what has already been reported in Section 6.3.2, provide a strong support to the notion that the new proposed architecture is better able to generalize and retain knowledge with respect to previous state of the art techniques.

Ablation Study for Proposed Solution

This split of the dataset, since it has established itself as the reference point for state of the art performances of models, has also been used to carry out an ablation study on the proposed model for this research, in order to be able to asses the separated impact for each set of features that composes the final fusion model.
Results for the Ablation Study are summarized in the table below:

Polyvore 68k - Ablation Study		
Modules	CE - AUC	FITB - Acc
Vis	0,878	0,586
Cat	0,825	0,407
Text	0,830	0,462
Vis + Cat	0,897	0, 593
Vis + Text	0,90	**0,625**
Text + Cat	0,850	0,459
Full	**0,91**	**0,625**

Table 6.6: *Polyvore 68k - Ablation Study*

What can be noticed by analysing data reported in the above table points out to a number of main insights, both relevant to the study of the field and future work for Recommender Systems in the domain:

- Visual Features are integral to the ability of the network to have satisfactory performances in the FITB task; as it is possible to notice from results analysis, the network is able to get over the threshold of 0,5 Acc in the task only when provided with visual features inside the fusion model

- On this particular dataset, the construction of the categories does not allow to properly exploit the information in that set of features; we can notice that, in the ablation study carried out by removing categorical feature from the fusion model, the network is able to achieve the same performance as the complete fusion model on the FITB task.

- Even tough information inferred from categorical feature is not relevant for the FITB task, it still allows to have slightly better performances on the CE task

6.3.4 Industrial Dataset Results

As described in Section 3.2, the possibility to have access to an industrial dataset allowed for performances to also be tested in a more real-world scenario.
Due to the structural differences between the publicly available dataset and the industrial one, and the limitations presented by the architecture described in [Han+17], it was not possible to test that specific model in the industrial environment; as a consequence, the approaches tested and confronted on this dataset have been only the proposed solution and the ap-

proach from [Vas+18].
Below are reported the results for both approaches:

Industrial Dataset Results		
Algorithm	CE - AUC	FITB - Acc
TAE	0,925	0,641
OutfitTransfomer	**0,972**	**0,722**

Table 6.7: *Industrial Dataset Results*

As shown in the above table summarizing results for benchmark tasks, it is possible to state that the solution proposed in this theis is able to significantly outperform state of the art approach also on the available industrial dataset, providing an increase of 5,08% for the CE task and an increase of 12,64% for the FITB task.
It is also interesting to notice the fact that, with respect to the performances achieved using Polyvore dataset, the higher data quality and different construction of this dataset allow for a far higher Accuracy and AUC.
The difference in performance between industrial dataset and the publicly available one seem to suggest that, due to the way the public dataset has been collected, it may not be a good enough proxy for performances of an algorithm in an industrial, real use-case setting, and this definitely is something worth looking into for future research in the field.

6.4 Unconstrained Outfit Generation Task

This section is dedicated to the description and presentation of the results achieved by this research solution in the novel task as proposed in Section 2.1.3.
Results in this section are the first of their kind, and both the testing process and the metrics used have been designed and selected in order to bring the field of personalization in fashion one step closer to the industry level required and reached in other more mature fields, such as video or e-commerce recommendations.
Unfortunately, as a consequence of the difference in structure of the networks and technologies used, no other state of the art solution was able to be deployed on this novel task with a computing complexity, and as a consequence inference time, acceptable in order to properly test performance.
In order to give a better idea of the differences above expressed, the following table summarizes the computational complexity difference existing between

Inference Computational Complexity table	
Algorithm	Complexity
BiLSTM	$O(m \cdot h)$
TAE	$O(m \cdot n^2)$
OutfitTransformer	$O(m)$

Table 6.8: *Inference Time Complexity for predicting 1 item from 1 outfit*

our solution and the current state of the art from [Vas+18]: where n is the incomplete outfit length and m is the number of items in the catalogue, and h is the number of items removed from the outfit. The complexity of BiLSTM and TAE is assumed directly from their implementation and the description of the inference process they provide, even if, at least for TAE, it is possible to reduce the complexity to $O(m \cdot n)$. While theoretically it would be possible to run the BiLSTM only once for each outfit (reducing the complexity to $O(m)$), the authors themselves adfirm that this does not guarantee style coherence if the items composing such incomplete outfits are not contiguous in the ordering assumed by the RNN [Han+17]. So, in case of 'blanks' between the positions of the items in the sequence, first all the blanks must be filled, repeating such procedure for all the blanks in the incomplete outfits.
The dependence of the inference complexity from the length of the outfit (or the number of removed items, that in this task sometimes is higher than incomplete outfit length) makes the testing of these algorithms on the new task unfeasible, since for each single question (an outfit), one item at a time is removed generating $|O|-2$ incomplete outfits, and the task is evaluated on all the outfits that share the non-removed items.
So, it is possible to present experimental results describing the performances only of the proposed solution, laying the ground work for those experimental results to serve as a first baseline confrontation for future research in the domain.
It is important to notice that, by proposing a solution that is both generative and guarantees a linear computational complexity for inference time, it is now technically possible to build a recommendation engine able to serve users at scale in a real-world scenario.

6.4.1 Recommendation Metrics

The metrics that have been adopted in order to evaluate performances of the models are the following, and they all are some of the most common and widely used metrics for evaluating Recommender Systems:

- Mean Reciprocal Rank @k
- Recall @k
- Precision @k
- Mean Average Precision @k
- F1 score @k

Some of the above mentioned metrics required to be adapted to the new domain and use case since evaluation of performances, in the context of recommendation and generation of outfits for the fashion industry, presents some relevant differences with what is considered to be "classic" use case of video or product recommendation.
Construction and usage of the adapted metrics for this new domain has been extensively described and discussed in both Sections 2.1.3 and 2.2.

6.4.2 Polyvore Dataset

This subsection will present results obtained by challenging the proposed architecture on the novel task, using as benchmark the clean version of the Polyvore 68k - NonDisjoint dataset.
For the public dataset, it was only possible to test performances in an offline fashion since we had no availability of fashion experts to online test results. Metrics have been tested and reported separately for outfits composed, in total, by a number of items ranging from 4 to 9, in order to also be able to share some insight on the progression of performance related to the size of the outfit evaluated.

Experimental Results - Mean Average Precision

Below are presented, in summarizing tables, results for offline testing considering evaluation metric MAP:

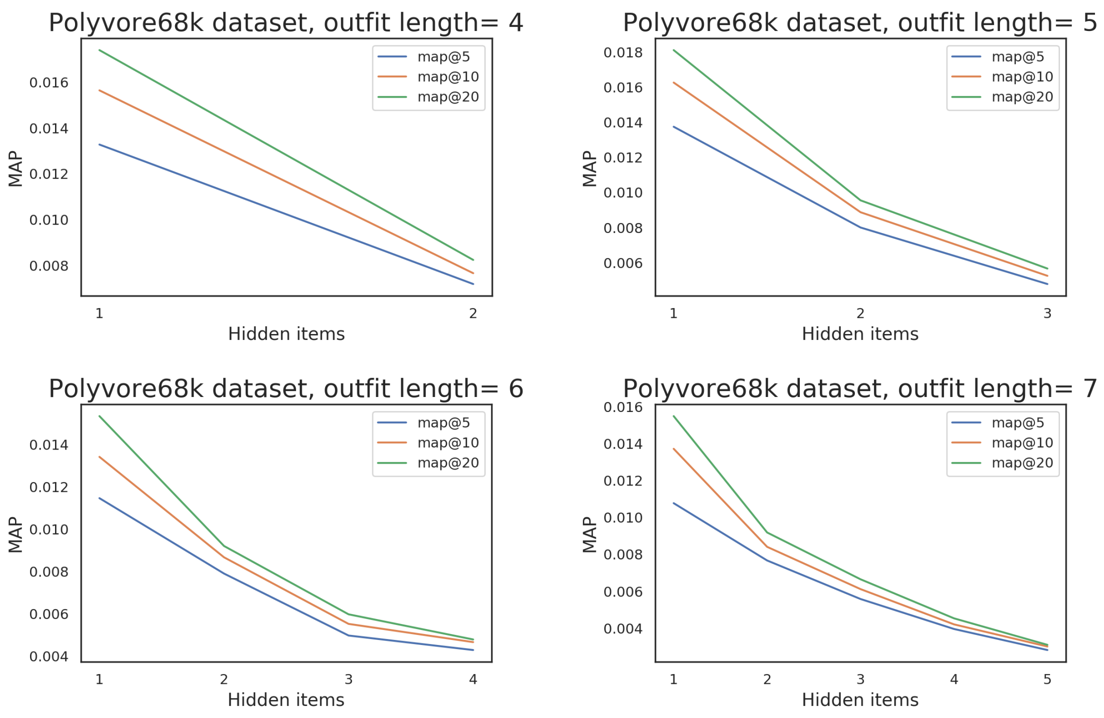

Figure 6.1: *MAP scores for different outfit lengths*

As shown in Fig. 6.1, the MAP decreases when hiding more items, regardless for all outfits length. This is an expected behavior, since hiding more items means decreasing the quantity of information provided to the model, therefore the chances of predicting the exact items that appear in the original outfits decrease. The relationship is better than linear, i.e. decreasing the number of visible items produces a less than linear reduction in the MAP score.

Experimental Results - Recall

Below are presented, in summarizing tables, results for offline testing considering the evaluation metric recall. Even for the recall, there is not a clear dependency with the outfit length, if not a slight increase with longer outfits. This proves that the algorithms is able to generalize the prediction of the missing item regardless of the outfit length. On the other side, recall decreases when hiding more items, since it becomes more difficult to model more missing items.

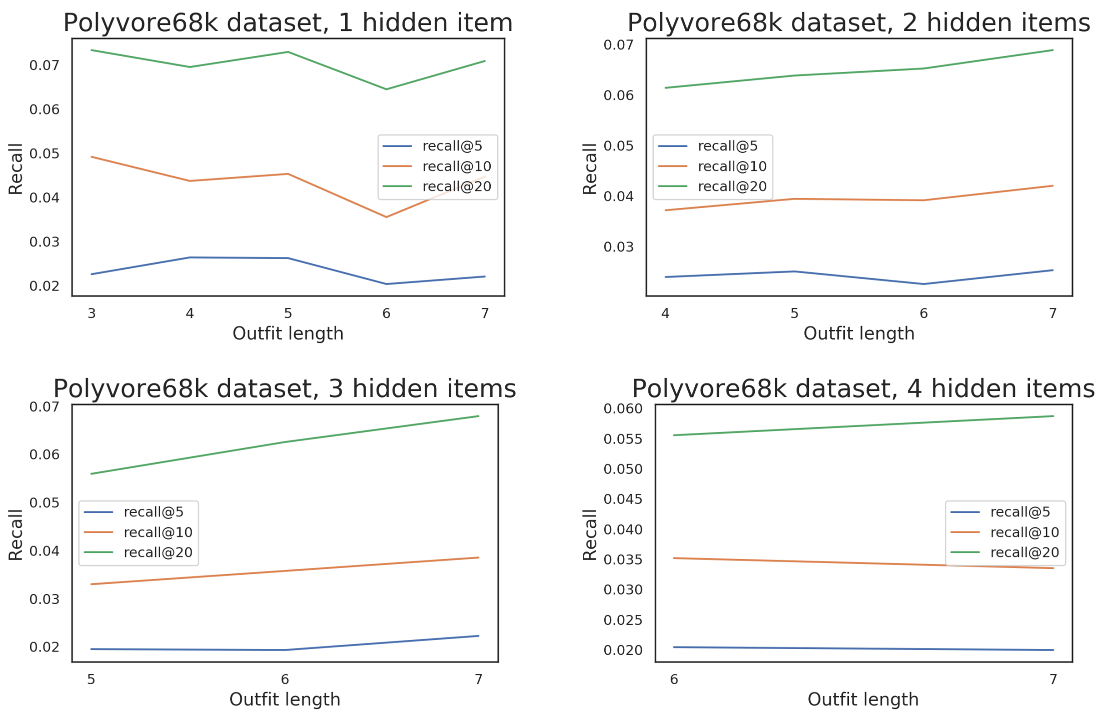

Figure 6.2: *Recall scores for different hidden items*

Experimental Results - Precision

Below are presented, in summarizing tables, results for offline testing considering the evaluation metric precision. For precision, the outfits containing 6 items have much lower scores than the others. The fact that with a lower cut-off there are higher scores is reasonable since usually the maximum number of possible items to guess is one, since the items usually appear once in the Polyvore dataset. Therefore, even if with a higher cut-off the probability of recommending the hidden item increases, increasing the cut-off will not allow the finding of more missing items, but it just will give a higher penalty due to the increased number of predicted items.

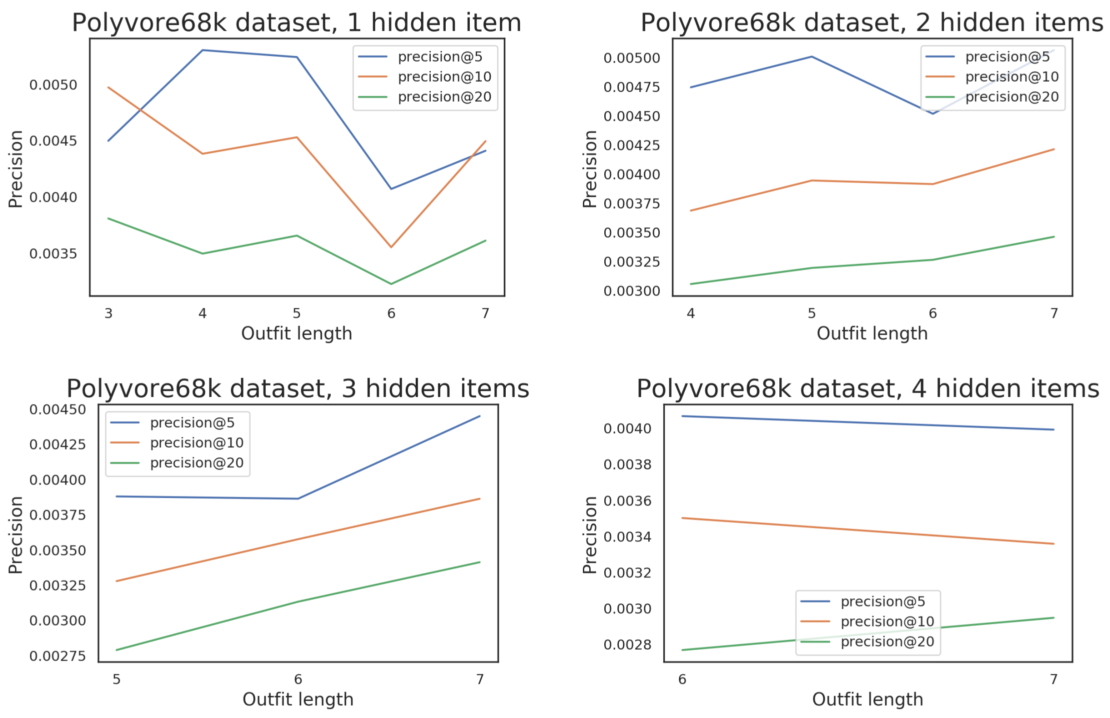

Figure 6.3: *Precision scores for different hidden items*

Polyvore Experimental Results Discussion

As it is shown in the above plottings, which summarize performances obtained on the newly proposed unconstrained task, the public Polyvore dataset shows a number of behaviours pointing to the fact that a lot of noise is encapsulated in the available information on which the network was trained. Even tough absolute values for the registered metrics are rather low, compared to other domains of applications for Recommender Systems, it is possible to notice how at least it can be confirmed that the proposed solution is able to generalize knowledge and take advantage of the number of samples shown, as for example in Section 6.1 performances go down as expected with the growth of number of masked items.

6.4.3 Industrial Dataset

This subsection will present results obtained by challenging the proposed architecture on the novel task, using as benchmark the industrial dataset made available for this research.

Below are presented, in summarizing tables, results for offline testing considering the evaluation metric MAP.
As expected, in this dataset we can see how the MAP increases more than linearly with length of the outfits (at least for 1 and 2 hidden items). Moreover, the scores are close for all cut-offs and for longer outfits the MAP@5 tends to perform better than the others, meaning that the missing items are usually predicted in the first 5 positions.
A linear, or even sublinear decrease in the MAP scores is evident when the number of hidden items increases.

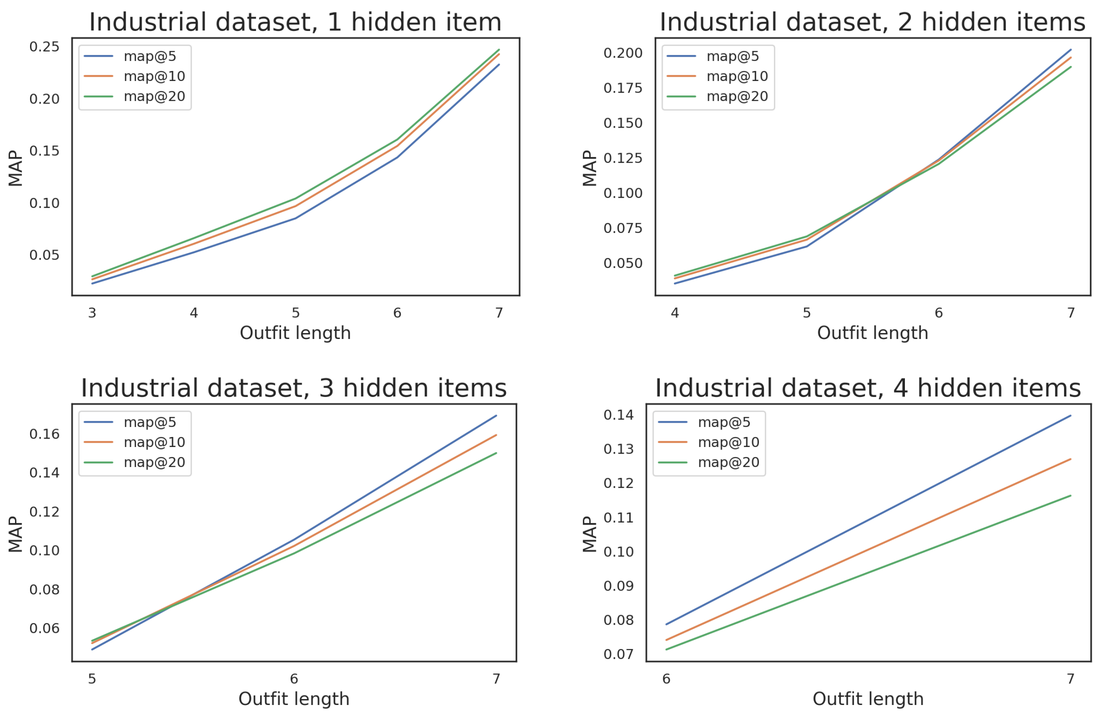

Figure 6.4: *MAP scores for different hidden items*

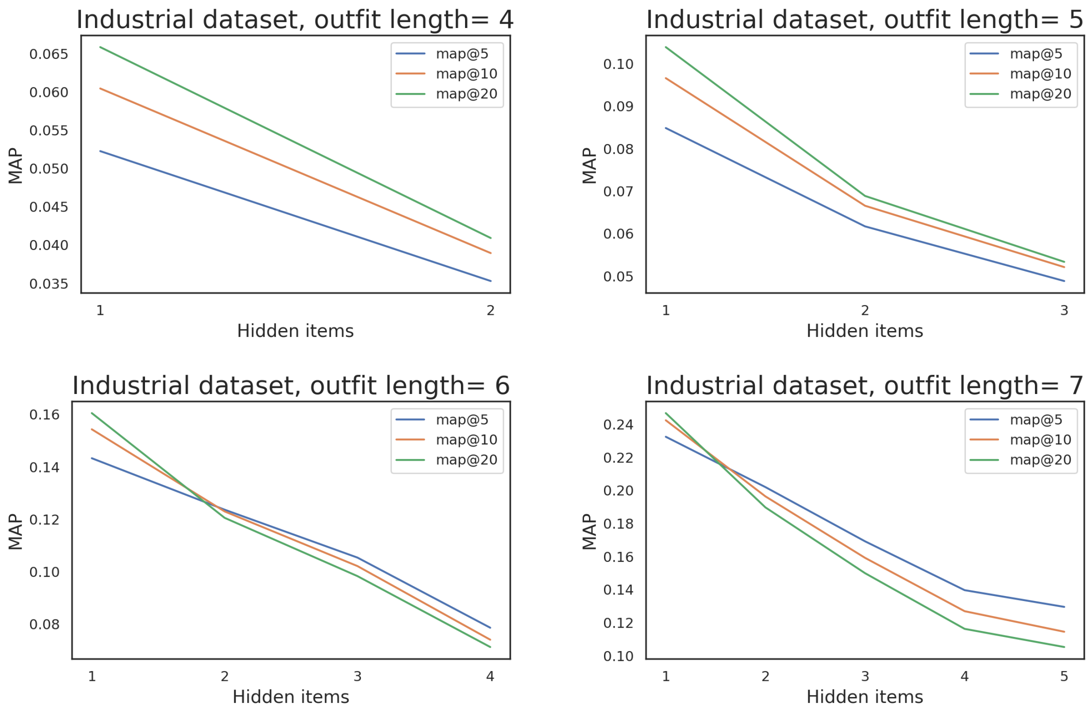

Figure 6.5: *MAP scores for different outfit lengths*

Experimental Results - Recall

Below are presented, in summarizing tables, results for offline testing considering the evaluation metric recalll. The increasing recall with regard to the number of hidden items means that, with a dataset where the same items appear multiple times, the model is able to infer more and more combinations of missing items. Of course, a fair comparison would require to check, for an outfit O, all the k^h possible predicted subsets (where h is the number of hidden items) with all the outfits that have in common only the visible items in O. Unfortunately, doing such comparison for all the outfits in the testset requires an exponential time complexity. The closest proxy to this optimal measurement is to evaluate the recall as the intersection of the set of recommended items and the one of all the possible missing items, given the all the outfits sharing the visible ones.

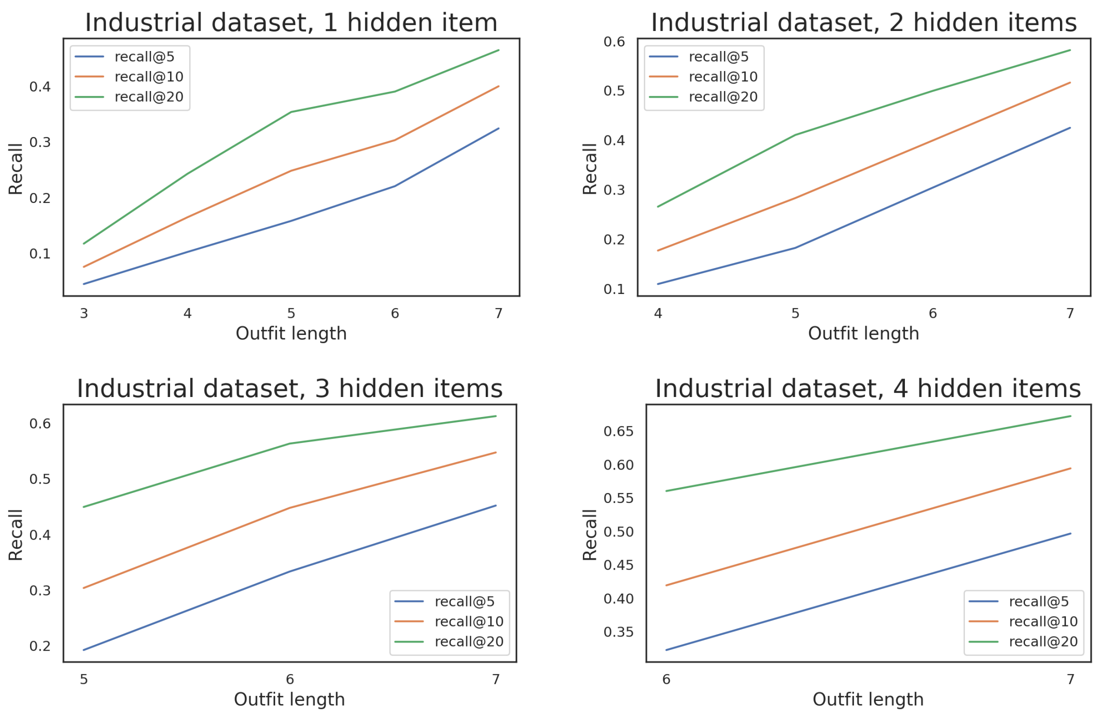

__Figure 6.6:__ Recall scores for different hidden items

Experimental Results - Precision

Below are presented, in summarizing tables, results for offline testing considering evaluation metric Precision. In all the following graphs plotting precision, the cut-off @ 5 has always the highest score, meaning that the model predicts the missing items and returns them in the highest ranking position (top 5) more often than in the others.

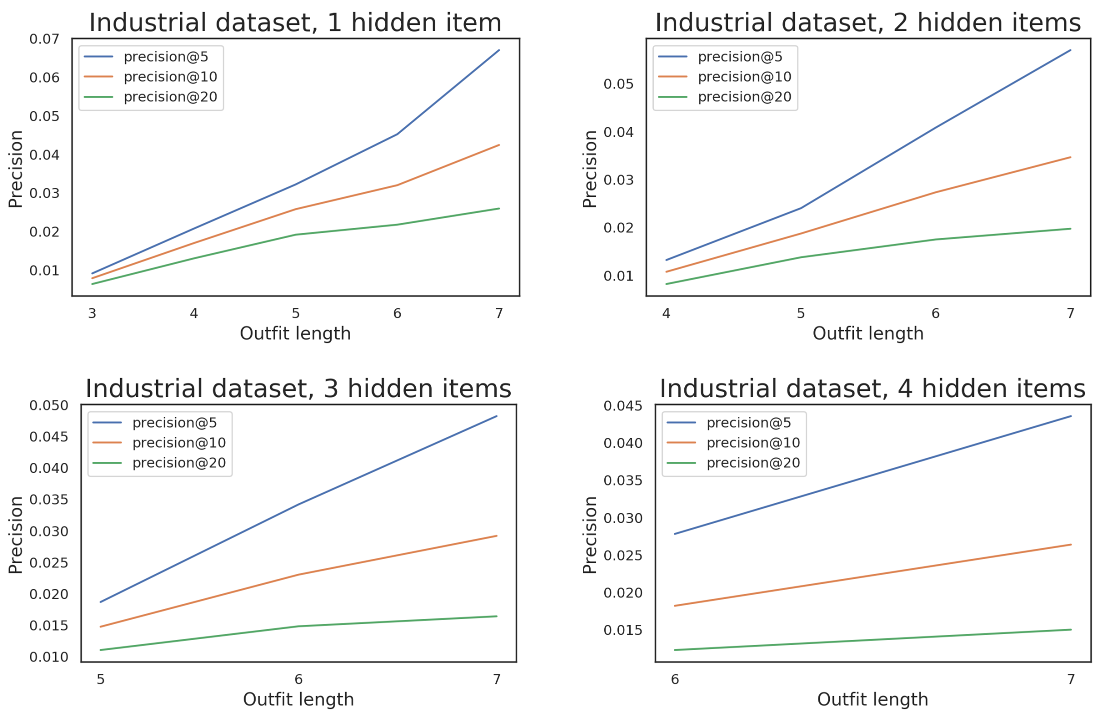

Figure 6.7: *Precision scores for different hidden items*

Online Expert Testing

Thanks to the cooperation with an industry partner, for the results evaluated in this context was also possible to include, on top of the same kind of offline evaluation carried out also for the publicly available dataset, an analysis of the results obtained running an online evaluation with professional experts in the fashion domain.
Online evaluation has been developed in the form of a survey, asking three different professional stylists from our industrial partner to evaluate a series of 215 outfits; for each of the proposed outfits one of the composing elements had been removed, and substituted instead with a recommended item chosen among the entire catalogue available for the category of the removed ground-truth item, with the exclusion of the ground-truth one. Evaluators were asked to judge if the presented outfit was style coherent or not, and in case of a negative judgement to point out which items didn't fit into the outfit.
In order to have a single evaluation able to encompass all the feedback received, the approach of Majority Voting has been used in order to choose whether to consider the evaluation of a specific outfit a positive or a negative

hit.
Results below show two different evaluation methods to define how to assign a feedback in the class of positives or negatives:

- The first and strictest option considers as positive feedback only that where the entire outfit was judged as style-coherent and as negative that where the outfit was judged as non-style-coherent
- The second option considers as positive feedback the same feedback as in the previous case, but removes from the negative feedback those outfit that were judged as non-style-coherent but where the evaluators failed to identify the recommended item as non-fitting.

The summarizing tables below show results for industrial online evaluation:

Online Evaluation Summary				
	# Answers	Positive	Negative	% Positive
Evaluator 1	215	197	18	**91,63**
Evaluator 2	215	144	71	**66,98**
Evaluator 3	215	186	29	**86,51**
AVG	215	175,7	39,3	**81,71**

Table 6.9: *Online Evaluation Summary*

Majority Voting Results	
Positive	190
Negative	25
% Positive	**88,37**

Table 6.10: *Majority Voting Results*

Majority Voting Results	
Positive	190
Negative	4
% Positive	**97,90**

Table 6.11: *Majority Voting Results - predicted items correctly identified*

Industrial Experimental Results Discussion

As it is possible to notice from results reported in the above section, it can be easily seen how absolute performance values for registered metrics differ

significantly from the ones observed on the available public dataset.
On a dataset which is considerably closer to a real use case scenario, the plotted graphs for performances achieved show all the behaviours that would be expected from a well functioning architecture, on all registered metrics.
On top of that, it can be stated that the above results allow to draw a first of its kind parallel between typical Information Retrieval metrics used in most research fields and actual performances that could be achieved in a real use case via the expert evaluation study provided.

Chapter 7

Conclusion

In this chapter, the key takeaways from this research work are going to be presented and discussed.
The chapter is also going to be dedicated to the proposal of relevant future work that can be conducted in the field of Recommendation Systems for the fashion domain stemming from the results and improvements presented, while at the same time laying out the most relevant limitations that are still in place with today's available technologies and approaches.

7.1 Improvements Offered

While the novel approach to Recommender Systems for the Fashion industry proposed here provides a number of improvements over previous state of the art approaches, the most relevant ones, both in terms of the impact they may have in providing a viable solution for the industry and of performance improvement, can be articulated as follows:

- The proposed solution simplifies the process of recommending style compatible items to complete a given outfit from a computational complexity of $O(n^2)$ to $O(n)$, de facto allowing to build a system that is able to scale linearly with the number of items in the catalogue at inference time,

- It provides performance improvements for all metrics on all considered benchmark tasks, compared to previous state of the art approaches.

- A performance study has been carried out and presented on an industrial dataset, built from live data coming from an operating fashion industry player, allowing for a parallel to be inferred between performances tested in a publicly available dataset and actual real world performances in an industrial setting.

- The ability to inspect an industrial dataset allowed for some relevant insights to be collected on the intrinsic limitations of the currently used publicly available dataset, such as the significantly different structure and statistical composition.
- A novel task has been proposed, tested and documented that is more closely related to real use cases for Recommender Systems in the Fashion industry than previous benchmark tasks, in order to serve as baseline confrontation for future researchers in the domain.

As those are some of the most relevant contributions of this research work to the field, it is still clear that numerous limitations are still present and need to be addressed, as is going to be better discussed in the next section.

7.2 Limitations

As already stated in the above section, even tough many interesting steps forward have been implemented as a consequence of this research work, a number of limitations still need to be addressed and solved in order for solutions in this specific domain to get on par, in terms of both performance and industrial practicality, with more mature industries.
It is possible to lay down the most relevant of those limitations as follows:

- The first severe limitation in producing high performance algorithms in the fashion domain is the lack of availability of a properly structured, publicly available dataset that could be used as a proxy for performances in a a real use case scenario.
- Another important limitation that should be duly noted, speaking of available data, is that sparsity of the datasets in the fashion domain is larger than sparsity in other domains. For example, when comparing sparsity levels for available data with the sparsity in widely used datasets from other domains, e.g. MovieLens datasets [1], it is possible to notice that all the the datasets used in this thesis have 50x $\sim$ 1000x less outfit-item connections, that is the equivalent of user-item interactions in general recommender systems terminology, (cf. Tab. 3.1 sparsity entry with 99.46% from MovieLens 20M, 98.15% from MovieLens 1M, 94.12% from MovieLens 100k).

[1]url: https://grouplens.org/datasets/movielens

- Considering test cases for UOC task, this work still lacks a fair comparison of the proposed solution with general purpose Recommender Systems algorithms
- The process of generating a recommendation is still completely category dependent, and a study on impact of waiving the category constraint should be carried out
- Lastly, the set of features that have been used in order to feed the model and learn style compatibility are, for the moment, taken directly from previous works in the domain, but there is no formal or empirical guarantee that those are the most relevant features to be considered.

It is worth to notice that recently, towards the end of the development of this research thesis (during RecSys 2019 and ICCV 2019), some new published works ([Tan+19] [LM19]) tried to apply similar techniques based on Attention Mechanism to the tasks previously defined in literature. Even though the results they report are closer (but still slightly worse) to what we achieved than the ones reached by the referred papers, it was not possible to directly test and compare such works due to the time constraints typical of thesis works.

7.3 Future work

A lot remains be done for research about Recommender Systems in the Fashion domain, both in the process of addressing the limitations highlighted in previous section and trying to advance performances in tasks that are better correlated with actual use case scenarios.
From the machine learning point of view, some of the possible continuation of this work could be the investigation of the features importance and impact on the process of outfit composition, the explanation of what clothing features are more correlated in a fashionable outfit, the exploration of different type of visual inputs, such as full-body images, through techniques of visual object segmentation. Moreover, from a more recommender-related side, it would be relevant to join the information about outfit compatibility and completion and those related to users profiles and histories in typical e-commerce setting, and try to create models able to recommend items using the probability of completing an outfit given the his user's past purchases.

Bibliography

[Pre08] C.U. Press. *Cambridge Advanced Learner's Dictionary.* Cambridge University Press, 2008. ISBN: 9780521674683. URL: dictionary.cambridge.org/dictionary/english/fashion.

[RRS11] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to recommender systems handbook". In: *Recommender systems handbook.* Springer, 2011, pp. 1–35.

[Vas+18] Mariya I Vasileva et al. "Learning type-aware embeddings for fashion compatibility". In: *Proceedings of the European Conference on Computer Vision (ECCV).* 2018, pp. 390–405.

[Bos+12] Lukas Bossard et al. "Apparel classification with style". In: *Asian conference on computer vision.* Springer. 2012, pp. 321–335.

[Yam+12] Kota Yamaguchi et al. "Parsing clothing in fashion photographs". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE. 2012, pp. 3570–3577.

[Kia+14] M Hadi Kiapour et al. "Hipster wars: Discovering elements of fashion styles". In: *European conference on computer vision.* Springer. 2014, pp. 472–488.

[Han+17] Xintong Han et al. "Learning fashion compatibility with bidirectional lstms". In: *Proceedings of the 25th ACM international conference on Multimedia.* ACM. 2017, pp. 1078–1086.

[Vas+17] Ashish Vaswani et al. *Attention Is All You Need.* 2017. arXiv: 1706.03762 [cs.CL].

[Liu+19] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach.* 2019. arXiv: 1907.11692 [cs.CL].

[Li+17] Yuncheng Li et al. "Mining fashion outfit composition using an end-to-end deep learning approach on set data". In: *IEEE Transactions on Multimedia* 19.8 (2017), pp. 1946–1955.

[BHV16] Christian Bracher, Sebastian Heinz, and Roland Vollgraf. "Fashion DNA: merging content and sales data for recommendation and article mapping". In: *arXiv preprint arXiv:1609.02489* (2016).

[LSL17] Hanbit Lee, Jinseok Seol, and Sang-goo Lee. "Style2vec: Representation learning for fashion items from style sets". In: *arXiv preprint arXiv:1708.04014* (2017).

[NG18] Takuma Nakamura and Ryosuke Goto. "Outfit generation and style extraction via bidirectional lstm and autoencoder". In: *arXiv preprint arXiv:1807.03133* (2018).

[BH92] Ravi Boppana and Magnús M Halldórsson. "Approximating maximum independent sets by excluding subgraphs". In: *BIT Numerical Mathematics* 32.2 (1992), pp. 180–196.

[DCJ19] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. "Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches". In: *arXiv preprint arXiv:1907.06902* (2019).

[AIS93] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. "Mining association rules between sets of items in large databases". In: *Acm sigmod record*. Vol. 22. 2. ACM. 1993, pp. 207–216.

[Zak00] Mohammed Javeed Zaki. "Scalable algorithms for association mining". In: *IEEE transactions on knowledge and data engineering* 12.3 (2000), pp. 372–390.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543. URL: nlp.stanford.edu/projects/glove.

[Sze+16] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016). DOI: 10.1109/cvpr.2016.308. URL: http://dx.doi.org/10.1109/CVPR.2016.308.

[Aba+16] Martín Abadi et al. "Tensorflow: A system for large-scale machine learning". In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 2016, pp. 265–283.

[Ket17] Nikhil Ketkar. "Introduction to pytorch". In: *Deep learning with python*. Springer, 2017, pp. 195–208.

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[Cha+19] Sneha Chaudhari et al. *An Attentive Survey of Attention Models*. 2019. arXiv: 1904.02874 `[cs.LG]`.

[Wan+18] Wenguan Wang et al. "Attentive fashion grammar network for fashion landmark detection and clothing category classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4271–4280.

[Zah+17] Manzil Zaheer et al. *Deep Sets*. 2017. arXiv: 1703.06114 `[cs.LG]`.

[Lee+19] Juho Lee et al. "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. 2019, pp. 3744–3753. URL: http://proceedings.mlr.press/v97/lee19d.html.

[Dev+18] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. arXiv: 1810.04805 `[cs.CL]`.

[VM16] Daniel Ponsa Vassileios Balntas Edgar Riba and Krystian Mikolajczyk. "Learning local feature descriptors with triplets and shallow convolutional neural networks". In: *Proceedings of the British Machine Vision Conference (BMVC)*. Ed. by Edwin R. Hancock Richard C. Wilson and William A. P. Smith. BMVA Press, Sept. 2016, pp. 119.1–119.11. ISBN: 1-901725-59-6. DOI: 10.5244/C.30.119. URL: https://dx.doi.org/10.5244/C.30.119.

[Tan+19] Reuben Tan et al. "Learning Similarity Conditions Without Explicit Supervision". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[LM19] Katrien Laenen and Marie-Francine Moens. *Attention-based Fusion for Outfit Recommendation.* 2019. arXiv: 1908.10585 `[cs.CV]`.

Appendix A

Tables and Graphs for UOC task

A.1 Polyvore68k dataset

Experimental Results - Metrics @20

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 20:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@20	0.00350	0.00306
Recall@20	0.06956	0.06138
RR@20	0.01741	0.01612
F1@20	0.00666	0.00582
MAP@20	0.01741	0.00824

Table A.1: *Performances evaluated on K=20 for outfits of size 4*

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@20	0.00366	0.00320	0.00279
Recall@20	0.07298	0.06381	0.05594
RR@20	0.01811	0.01845	0.01589
F1@20	0.00696	0.00609	0.00531
MAP@20	0.01811	0.00955	0.00567

Table A.2: *Performances evaluated on K=20 for outfits of size 5*

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@20	0.00323	0.00326	0.00313	0.00277
Recall@20	0.06451	0.06519	0.06255	0.05554
RR@20	0.01533	0.01790	0.01683	0.01753
F1@20	0.00614	0.00622	0.00597	0.00527
MAP@20	0.01533	0.00920	0.00597	0.00478

Table A.3: *Performances evaluated on K=20 for outfits of size 6*

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@20	0.00361	0.00346	0.00341	0.00295	0.00253
Recall@20	0.07092	0.06883	0.06794	0.05870	0.05066
RR@20	0.01550	0.01776	0.01906	0.01672	0.01400
F1@20	0.00687	0.00659	0.00650	0.00561	0.00483
MAP@20	0.01550	0.00919	0.00665	0.00453	0.00310

Table A.4: *Performances evaluated on K=20 for outfits of size 7*

Unconstrained Metrics - 8 Items Outfits						
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden	6 hidden
Precision@20	0.00377	0.00352	0.00340	0.00307	0.00300	0.00307
Recall@20	0.07531	0.07037	0.06790	0.06142	0.06000	0.06173
RR@20	0.01614	0.01924	0.01835	0.01887	0.01577	0.01900
F1@20	0.00717	0.00670	0.00647	0.00585	0.00572	0.00585
MAP@20	0.01614	0.00999	0.00650	0.00511	0.00361	0.00368

Table A.5: *Performances evaluated on K=20 for outfits of size 8*

Unconstrained Metrics - 9 Items Outfits							
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden	6 hidden	7 hidden
Precision@20	0.00392	0.00478	0.00423	0.00420	0.00383	0.00356	0.00294
Recall@20	0.07837	0.09561	0.08464	0.08386	0.07649	0.07106	0.05956
RR@20	0.01877	0.02322	0.01857	0.01854	0.01928	0.01285	0.01349
F1@20	0.00746	0.00911	0.00807	0.00800	0.00730	0.00677	0.00561
MAP@20	0.01877	0.01188	0.00660	0.00519	0.00471	0.00269	0.00247

Table A.6: *Performances evaluated on K=20 for outfits of size 9*

Experimental Results - Metrics @10

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 10:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@10	0.00438	0.00369
Recall@10	0.04373	0.03718
RR@10	0.01565	0.01503
F1@10	0.00797	0.00670
MAP@10	0.01565	0.00766

Table A.7: *Performances evaluated on K=10 for outfits of size 4*

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@10	0.00453	0.00395	0.00328
Recall@10	0.04532	0.03943	0.03300
RR@10	0.01626	0.01712	0.01516
F1@10	0.00824	0.00717	0.00596
MAP@10	0.01626	0.00888	0.00526

Table A.8: *Performances evaluated on K=10 for outfits of size 5*

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@10	0.00355	0.00392	0.00358	0.00350
Recall@10	0.03553	0.03915	0.03576	0.03518
RR@10	0.01341	0.01693	0.01580	0.01722
F1@10	0.00646	0.00712	0.00650	0.00637
MAP@10	0.01341	0.00866	0.00552	0.00466

Table A.9: *Performances evaluated on K=10 for outfits of size 6*

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@10	0.00450	0.00421	0.00386	0.00336	0.00306
Recall@10	0.04470	0.04201	0.03854	0.03352	0.03075
RR@10	0.01373	0.01656	0.01765	0.01584	0.01369
F1@10	0.00817	0.00766	0.00702	0.00611	0.00557
MAP@10	0.01373	0.00841	0.00612	0.00421	0.00301

Table A.10: *Performances evaluated on K=10 for outfits of size 7*

Unconstrained Metrics - 8 Items Outfits						
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden	6 hidden
Precision@10	0.00457	0.00401	0.00354	0.00361	0.00343	0.00350
Recall@10	0.04568	0.04012	0.03539	0.03611	0.03432	0.03519
RR@10	0.01407	0.01741	0.01622	0.01755	0.01516	0.01827
F1@10	0.00831	0.00730	0.00643	0.00657	0.00624	0.00636
MAP@10	0.01407	0.00881	0.00563	0.00469	0.00334	0.00334

Table A.11: *Performances evaluated on K=10 for outfits of size 8*

Experimental Results - Metrics @5

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 5:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@5	0.00530	0.00475
Recall@5	0.02638	0.02399
RR@5	0.01328	0.01405
F1@5	0.00884	0.00791
MAP@5	0.01328	0.00718

Table A.13: *Performances evaluated on K=5 for outfits of size 4*

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@5	0.00524	0.00501	0.00388
Recall@5	0.02623	0.02509	0.01950
RR@5	0.01375	0.01554	0.01402
F1@5	0.00874	0.00835	0.00647
MAP@5	0.01375	0.00801	0.00479

Table A.14: *Performances evaluated on K=5 for outfits of size 5*

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@5	0.00407	0.00452	0.00386	0.00407
Recall@5	0.02035	0.02259	0.01932	0.02044
RR@5	0.01146	0.01533	0.01435	0.01608
F1@5	0.00678	0.00753	0.00644	0.00678
MAP@5	0.01146	0.00790	0.00497	0.00428

Table A.15: *Performances evaluated on K=5 for outfits of size 6*

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@5	0.00441	0.00507	0.00445	0.00399	0.00367
Recall@5	0.02205	0.02533	0.02225	0.01996	0.01847
RR@5	0.01078	0.01501	0.01645	0.01497	0.01289
F1@5	0.00735	0.00844	0.00742	0.00665	0.00612
MAP@5	0.01078	0.00767	0.00558	0.00396	0.00282

Table A.16: *Performances evaluated on K=5 for outfits of size 7*

Unconstrained Metrics - 8 Items Outfits						
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden	6 hidden
Precision@5	0.00469	0.00407	0.00346	0.00401	0.00370	0.00389
Recall@5	0.02346	0.02037	0.01728	0.02006	0.01852	0.01955
RR@5	0.01099	0.01438	0.01376	0.01556	0.01353	0.01685
F1@5	0.00782	0.00679	0.00576	0.00669	0.00617	0.00648
MAP@5	0.01099	0.00737	0.00470	0.00426	0.00286	0.00299

Table A.17: *Performances evaluated on K=5 for outfits of size 8*

Unconstrained Metrics - 9 Items Outfits							
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden	6 hidden	7 hidden
Precision@5	0.00502	0.00658	0.00564	0.00580	0.00527	0.00533	0.00469
Recall@5	0.02508	0.03292	0.02821	0.02900	0.02633	0.02665	0.02373
RR@5	0.01379	0.02040	0.01665	0.01729	0.01950	0.01261	0.01408
F1@5	0.00836	0.01097	0.00940	0.00967	0.00878	0.00888	0.00781
MAP@5	0.01379	0.01020	0.00572	0.00494	0.00427	0.00273	0.00259

Table A.18: *Performances evaluated on K=5 for outfits of size 9*

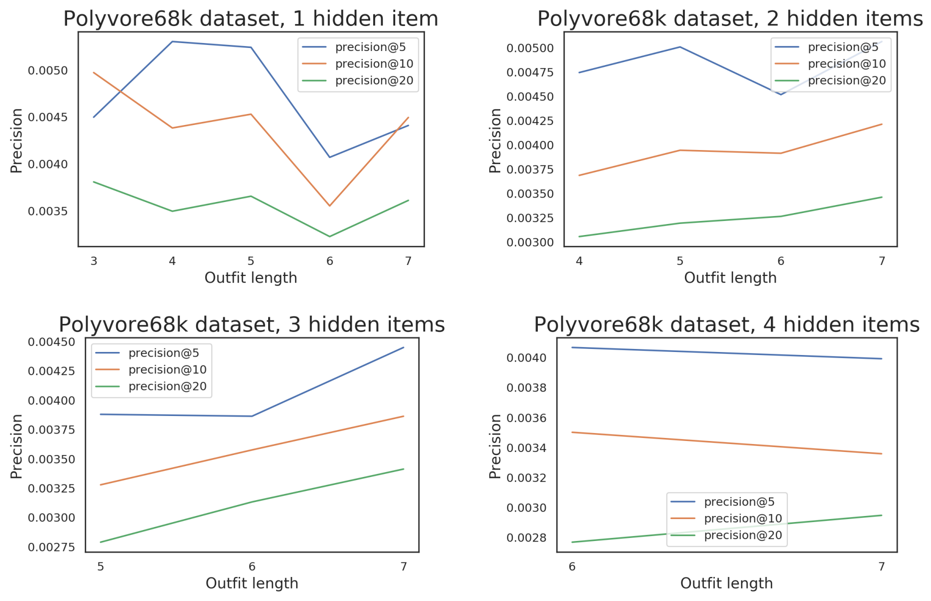

Figure A.1: *Precision scores for different hidden items*

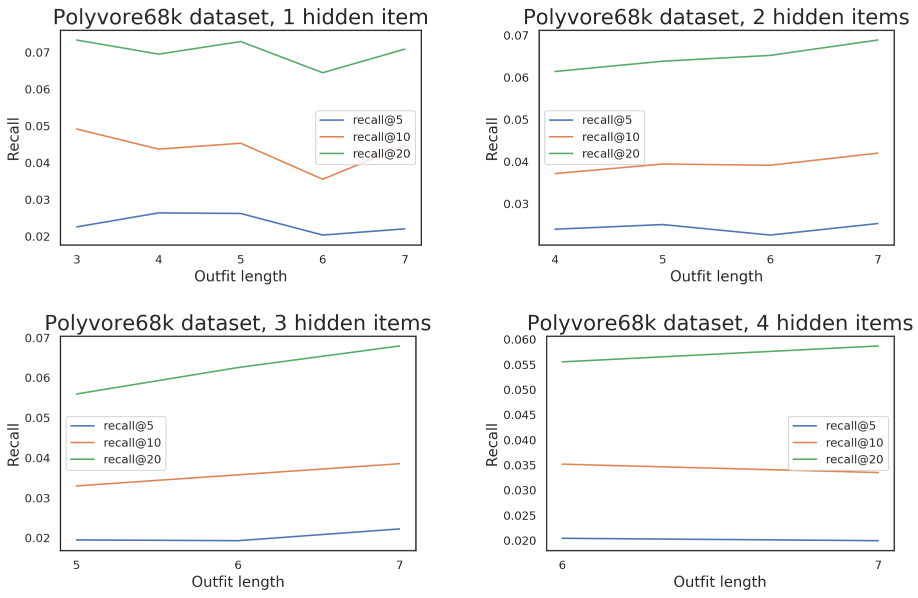

Figure A.2: *Recall scores for different hidden items*

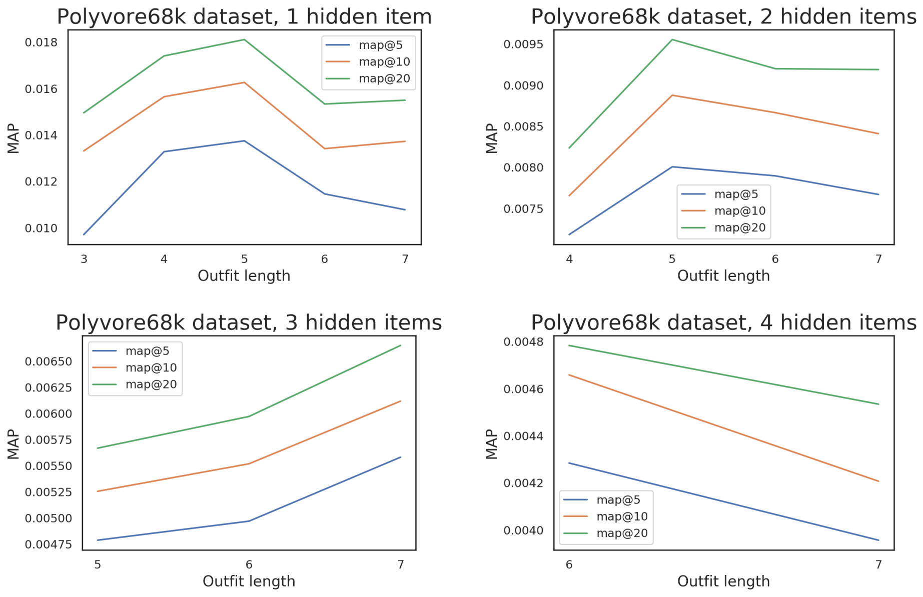

Figure A.3: *MAP scores for different hidden items*

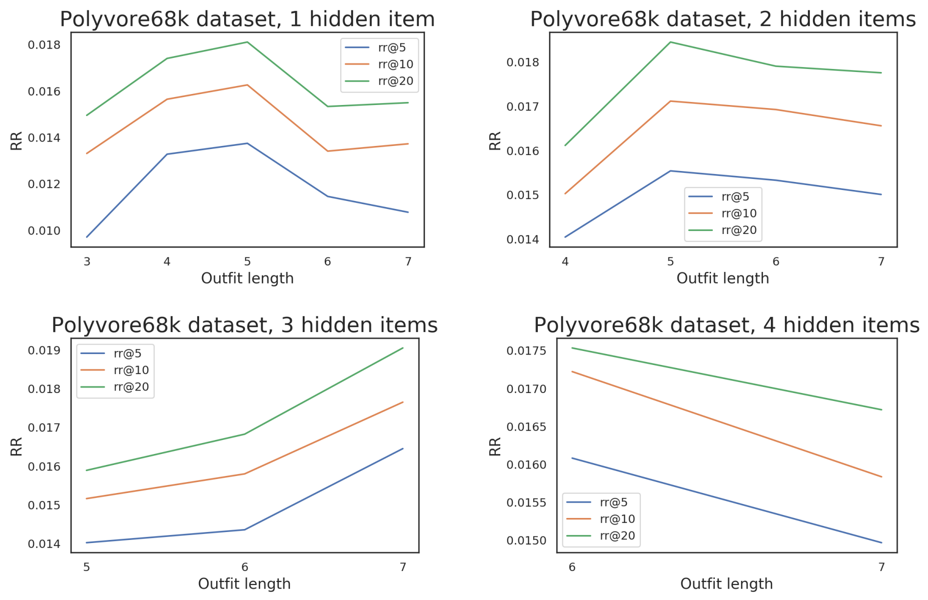

Figure A.4: *MRR scores for different hidden items*

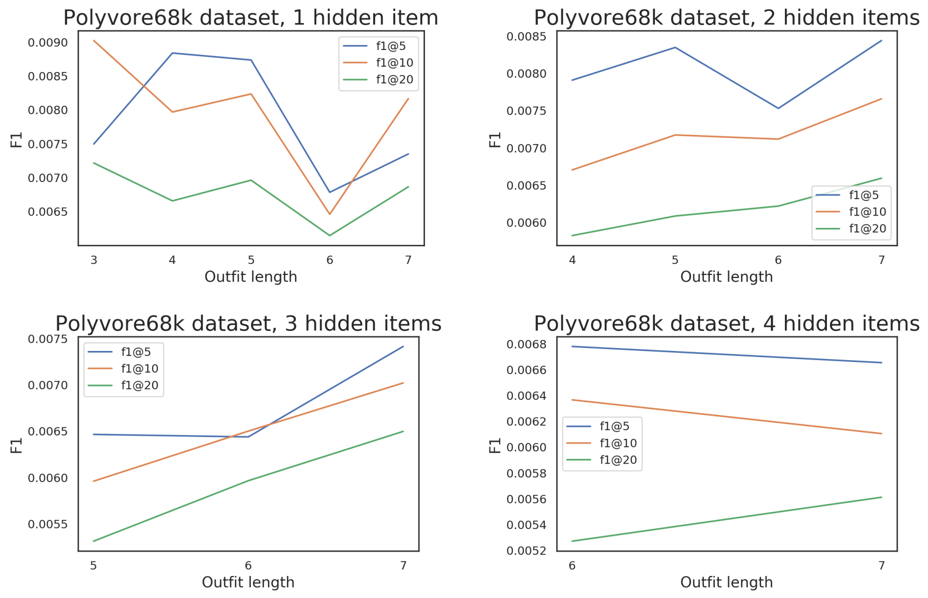

Figure A.5: *F1 scores for different hidden items*

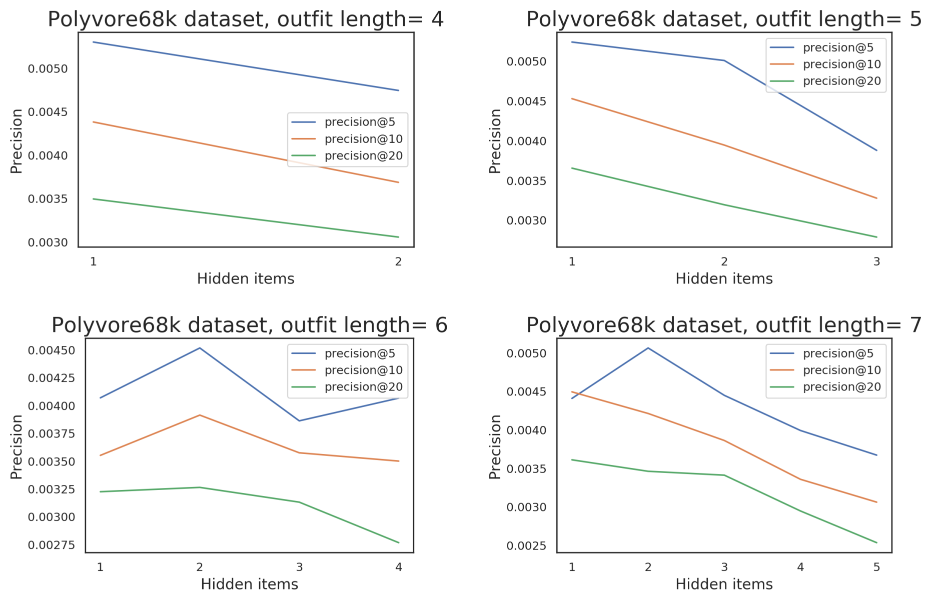

Figure A.6: *Precision scores for different outfit lengths*

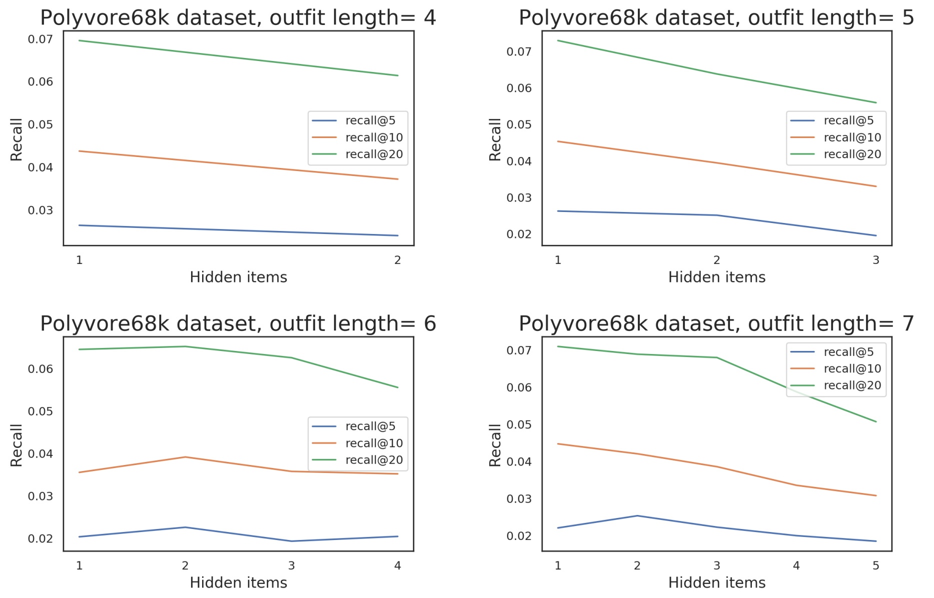

Figure A.7: *Recall scores for different outfit lengths*

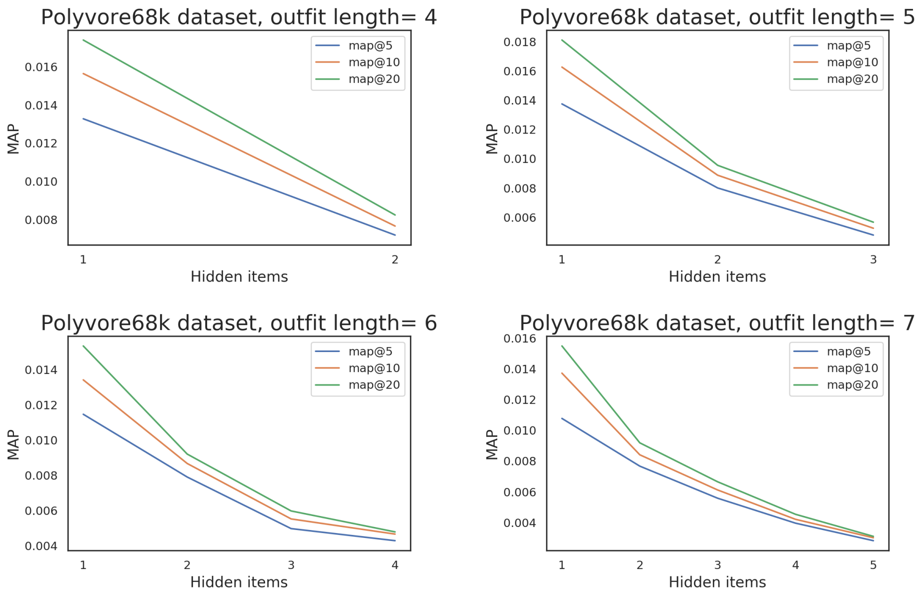

Figure A.8: *MAP scores for different outfit lengths*

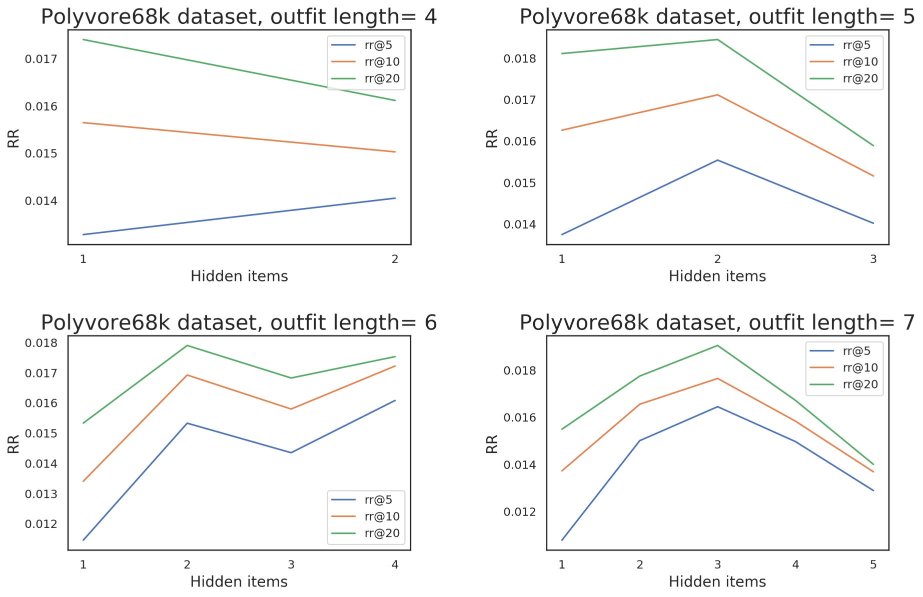

Figure A.9: *MRR scores for different outfit lengths*

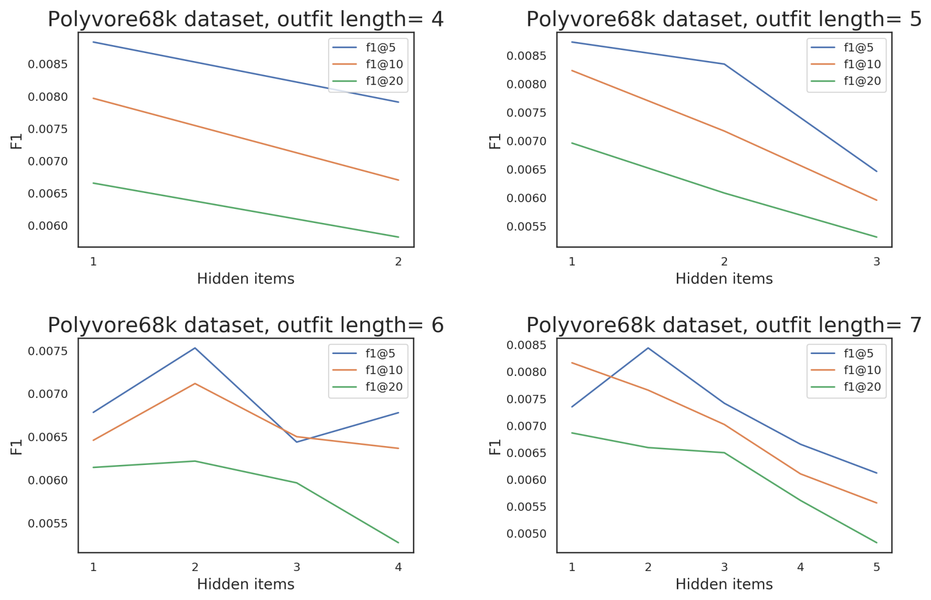

Figure A.10: *F1 scores for different outfit lengths*

A.2 Industrial Dataset

Experimental Results - Metrics @5

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 5:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@5	0.02076	0.01321
Recall@5	0.10246	0.10957
RR@5	0.05227	0.04110
F1@5	0.03444	0.02334
MAP@5	0.05227	0.03529

Table A.19: *Performances evaluated on K=5 for outfits of size 4*

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@5	0.03220	0.02404	0.01867
Recall@5	0.15823	0.18246	0.19247
RR@5	0.08485	0.07669	0.06595
F1@5	0.05335	0.04195	0.03354
MAP@5	0.08485	0.06170	0.04884

Table A.20: *Performances evaluated on K=5 for outfits of size 5*

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@5	0.04523	0.04085	0.03419	0.02781
Recall@5	0.22055	0.30404	0.33343	0.32232
RR@5	0.14329	0.15304	0.14392	0.11758
F1@5	0.07473	0.07109	0.06102	0.05039
MAP@5	0.14329	0.12367	0.10540	0.07865

Table A.21: *Performances evaluated on K=5 for outfits of size 6*

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@5	0.06703	0.05702	0.04823	0.04359	0.04173
Recall@5	0.32432	0.42490	0.45190	0.49667	0.55671
RR@5	0.23243	0.24348	0.23432	0.20726	0.19606
F1@5	0.10991	0.09926	0.08576	0.07880	0.07642
MAP@5	0.23243	0.20200	0.16922	0.13968	0.12956

Table A.22: *Performances evaluated on K=5 for outfits of size 7*

Experimental Results - Metrics @10

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 10:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@10	0.01700	0.01076
Recall@10	0.16477	0.17714
RR@10	0.06046	0.04498
F1@10	0.03064	0.02016
MAP@10	0.06046	0.03894

Table A.23: *Performances evaluated on K=10 for outfits of size 4*

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@10	0.02579	0.01875	0.01474
Recall@10	0.24829	0.28306	0.30385
RR@10	0.09660	0.08176	0.06916
F1@10	0.04637	0.03491	0.02789
MAP@10	0.09660	0.06656	0.05211

Table A.24: *Performances evaluated on K=10 for outfits of size 5*

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@10	0.03201	0.02735	0.02302	0.01818
Recall@10	0.30322	0.39922	0.44770	0.41913
RR@10	0.15433	0.15269	0.14024	0.11195
F1@10	0.05721	0.05079	0.04339	0.03454
MAP@10	0.15433	0.12300	0.10217	0.07408

Table A.25: *Performances evaluated on K=10 for outfits of size 6*

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@10	0.04243	0.03467	0.02919	0.02638	0.02380
Recall@10	0.40000	0.51613	0.54722	0.59419	0.62849
RR@10	0.24241	0.23974	0.22480	0.19391	0.18024
F1@10	0.07494	0.06449	0.05493	0.05005	0.04547
MAP@10	0.24241	0.19639	0.15923	0.12696	0.11454

Table A.26: *Performances evaluated on K=10 for outfits of size 7*

Experimental Results - Metrics @20

Below are presented, in summarizing tables, results for offline testing considering evaluation metrics on k = 20:

Unconstrained Metrics - 4 Items Outfits		
Metric	1 Hidden	2 Hidden
Precision@20	0.01303	0.00820
Recall@20	0.24285	0.26566
RR@20	0.06587	0.04699
F1@20	0.02446	0.01586
MAP@20	0.06587	0.04091

Table A.27: Performances evaluated on K=20 for outfits of size 4

Unconstrained Metrics - 5 Items Outfits			
Metric	1 Hidden	2 Hidden	3 Hidden
Precision@20	0.01917	0.01379	0.01103
Recall@20	0.35380	0.41041	0.44936
RR@20	0.10390	0.08413	0.07011
F1@20	0.03584	0.02657	0.02145
MAP@20	0.10390	0.06887	0.05335

Table A.28: Performances evaluated on K=20 for outfits of size 5

Unconstrained Metrics - 6 Items Outfits				
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden
Precision@20	0.02177	0.01750	0.01481	0.01226
Recall@20	0.39045	0.49935	0.56318	0.56030
RR@20	0.16050	0.15101	0.13678	0.10883
F1@20	0.04025	0.03367	0.02872	0.02388
MAP@20	0.16050	0.12056	0.09834	0.07129

Table A.29: Performances evaluated on K=20 for outfits of size 6

Unconstrained Metrics - 7 Items Outfits					
Metric	1 Hidden	2 Hidden	3 Hidden	4 hidden	5 hidden
Precision@20	0.02596	0.01974	0.01640	0.01499	0.01394
Recall@20	0.46486	0.58165	0.61247	0.67240	0.73559
RR@20	0.24673	0.23555	0.21804	0.18486	0.17155
F1@20	0.04698	0.03803	0.03178	0.02918	0.02724
MAP@20	0.24673	0.18973	0.14995	0.11627	0.10529

Table A.30: *Performances evaluated on K=20 for outfits of size 7*

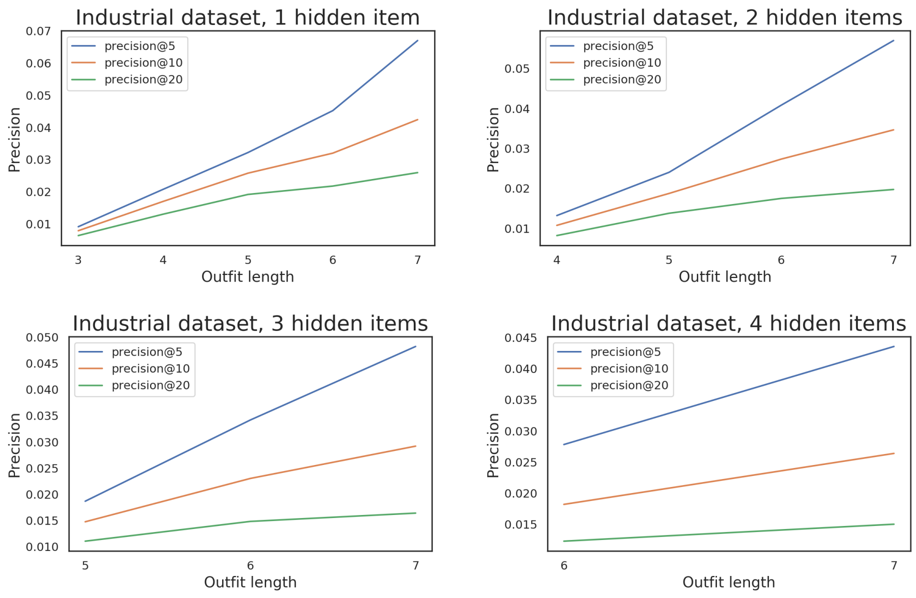

Figure A.11: *Precision scores for different hidden items*

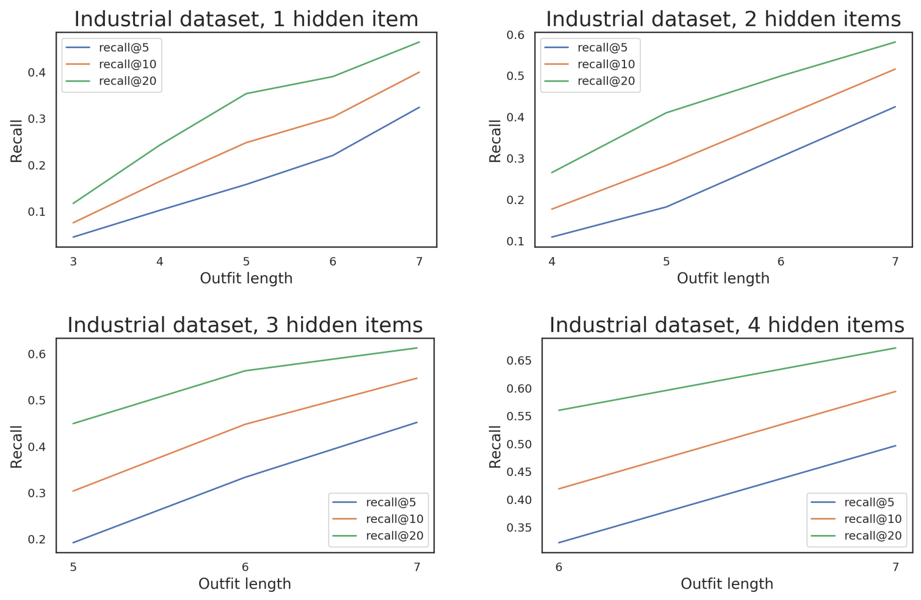

Figure A.12: *Recall scores for different hidden items*

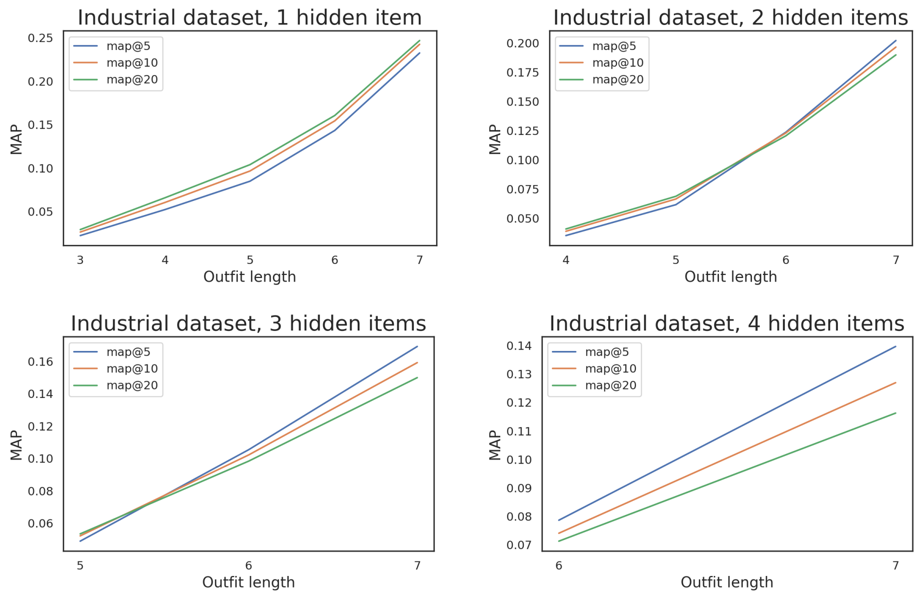

Figure A.13: *MAP scores for different hidden items*

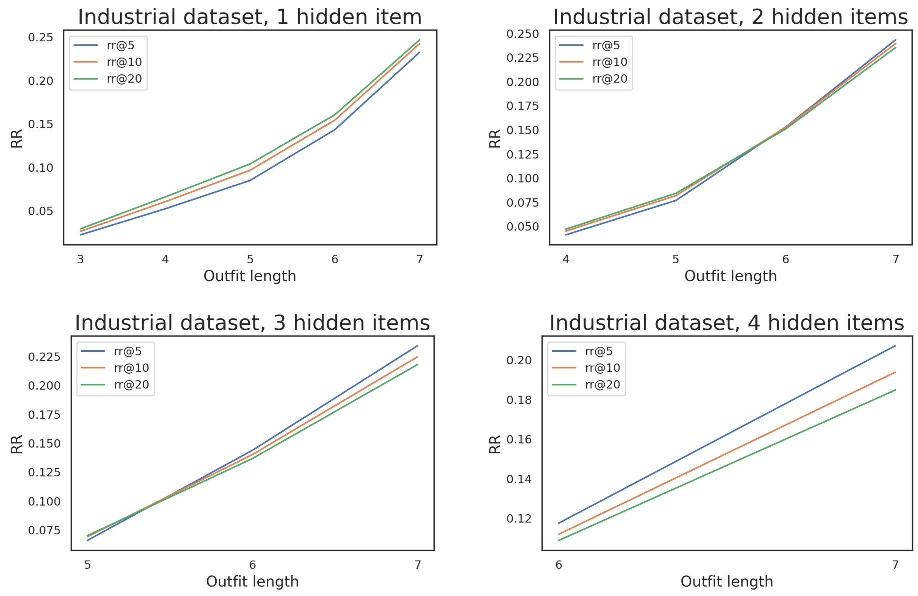

Figure A.14: *MRR scores for different hidden items*

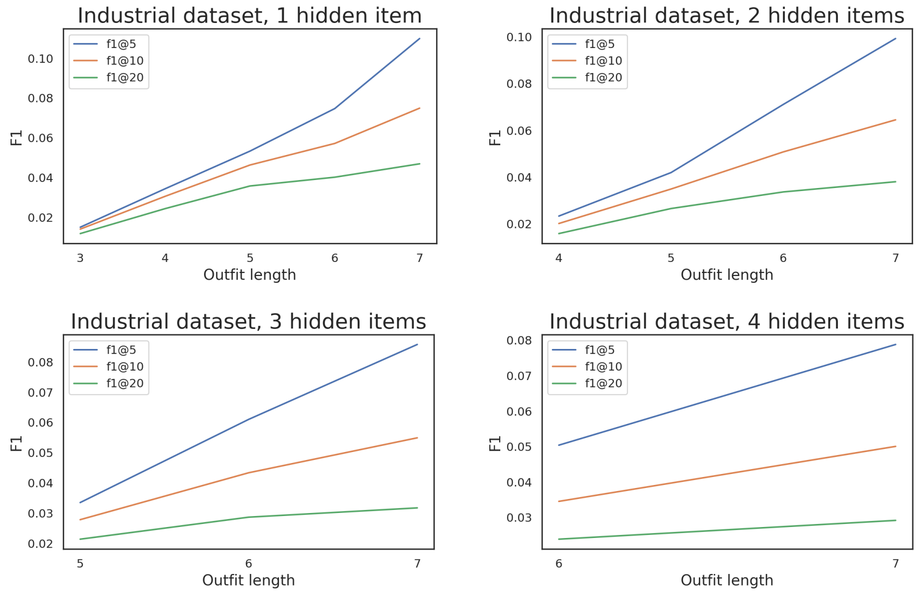

Figure A.15: *F1 scores for different hidden items*

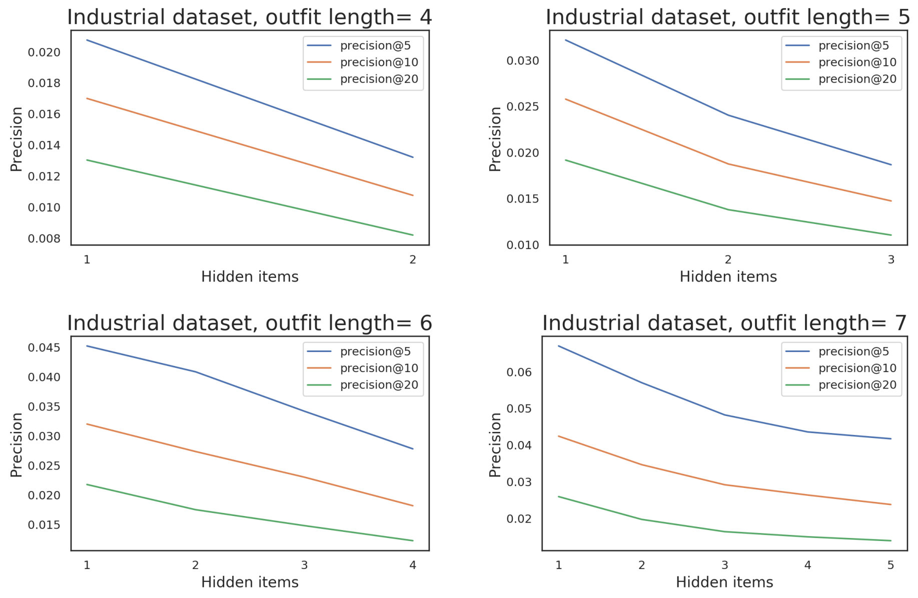

Figure A.16: *Precision scores for different outfit lengths*

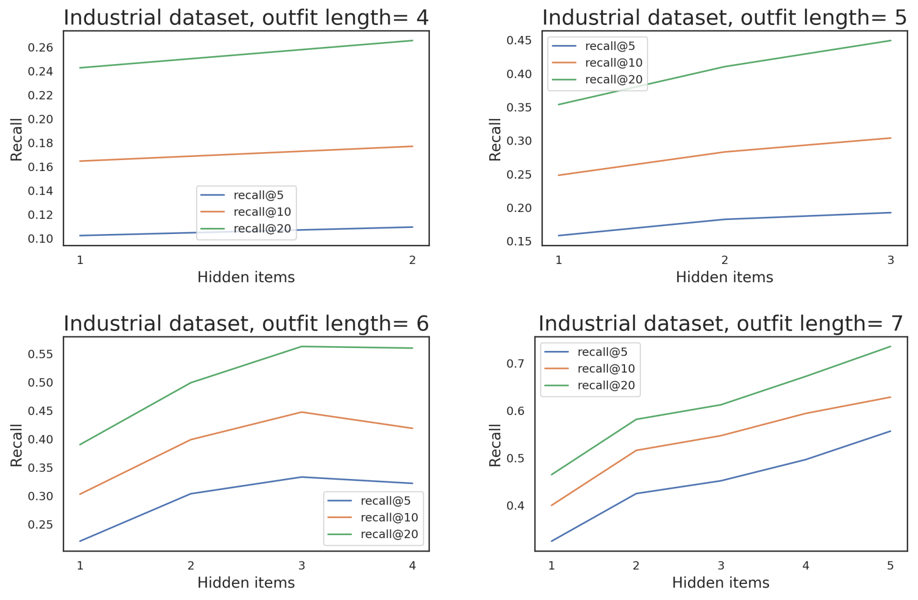

Figure A.17: *Recall scores for different outfit lengths*

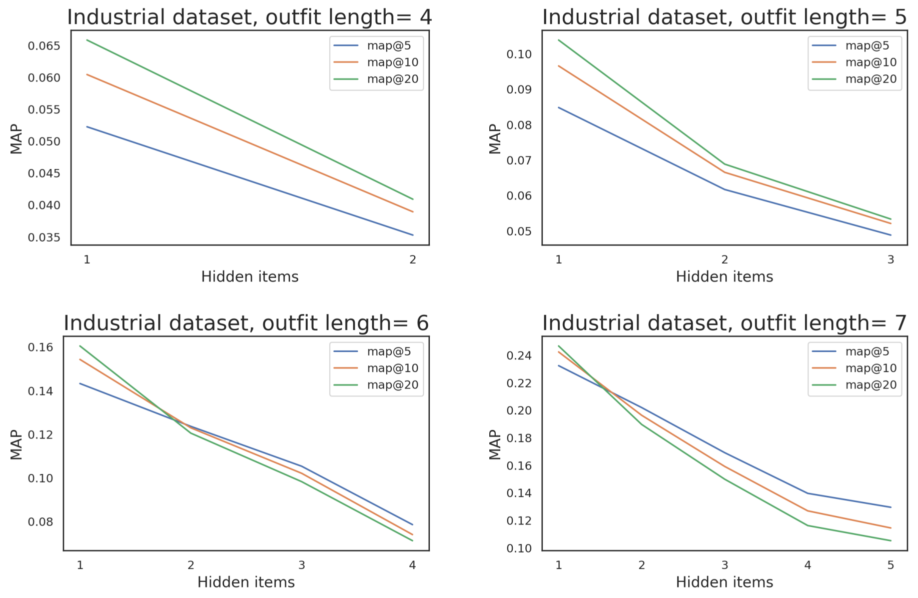

Figure A.18: *MAP scores for different outfit lengths*

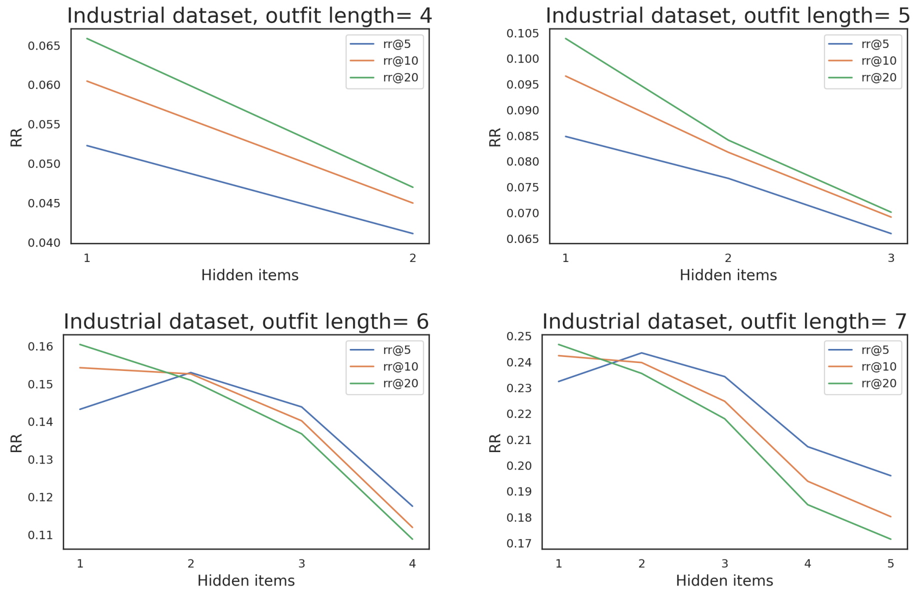

Figure A.19: *MRR scores for different outfit lengths*

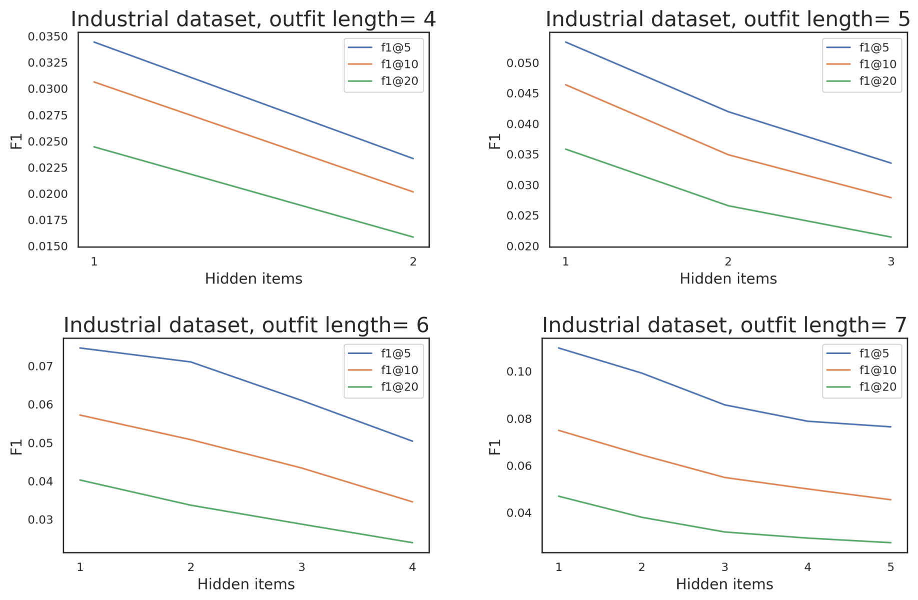

Figure A.20: *F1 scores for different outfit lengths*

Appendix B

Generated Outfits Examples

In this appendix we provide some outfits generated by our model.

B.1 Industrial Dataset

The following outfits are taken from the survey sent to the experts. In order to create a test similar to the FITB task, one item (the one framed in blue) was randomly removed from the outfit and replaced with an item recommended by our model.

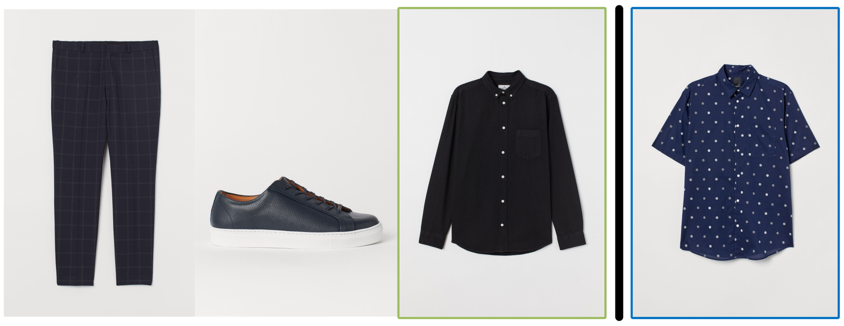

Figure B.1: *Outfit 1 - replacement*

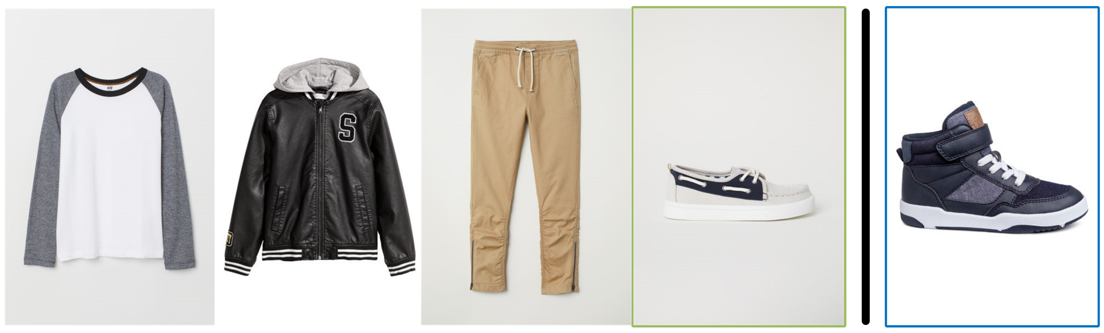
Figure B.2: *Outfit 2 - replacement*

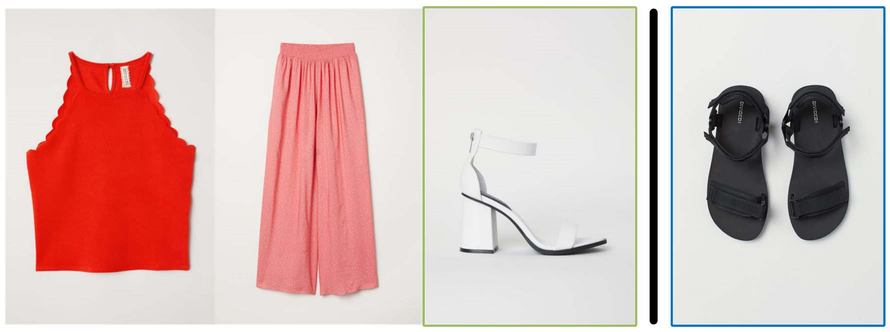
Figure B.3: *Outfit 3 - replacement*

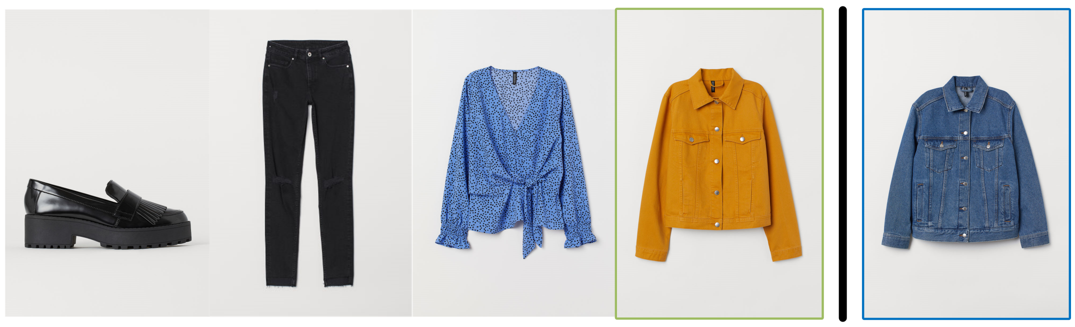
Figure B.4: *Outfit 4 - replacement*

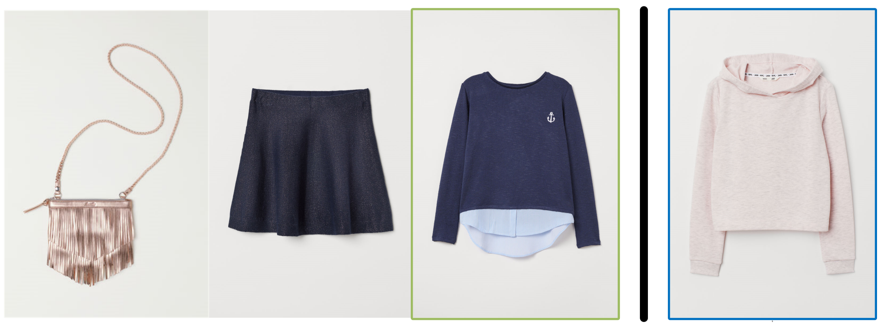
Figure B.5: Outfit 5 - replacement

The following outfits are generated using the items framed in black as seed of an incomplete outfit. The items framed in green are those generated by the model in order to complete the provided set of items.

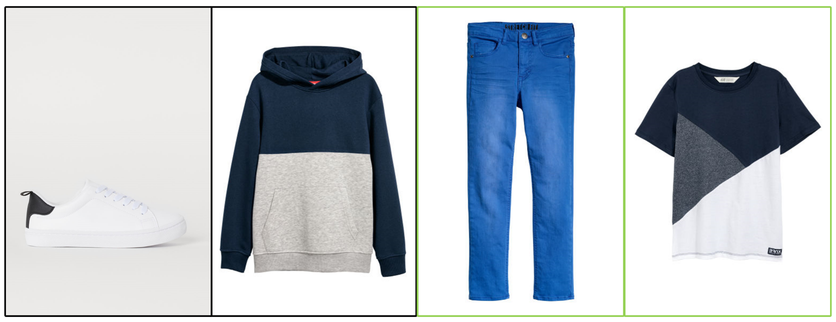
Figure B.6: Outfit 1 - Industrial dataset

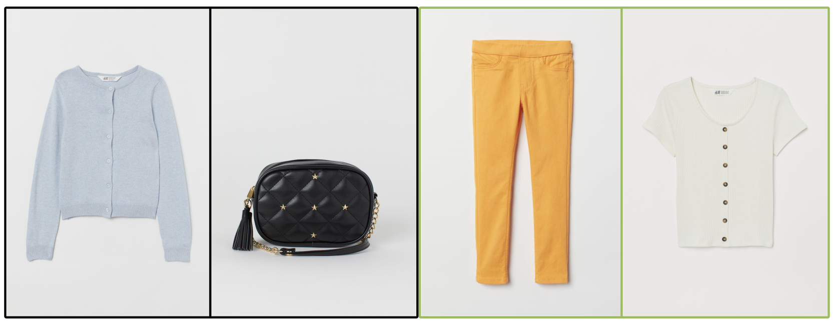
Figure B.7: Outfit 2 - Industrial dataset

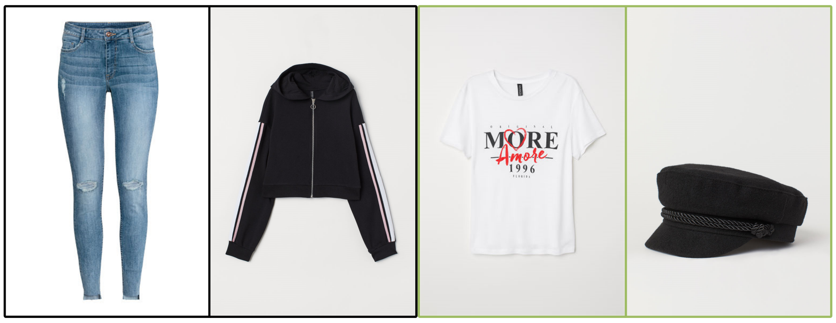

Figure B.8: *Outfit 3 - Industrial dataset*

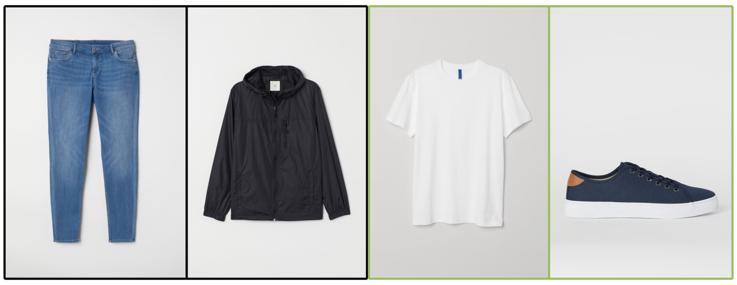

Figure B.9: *Outfit 4 - Industrial dataset*

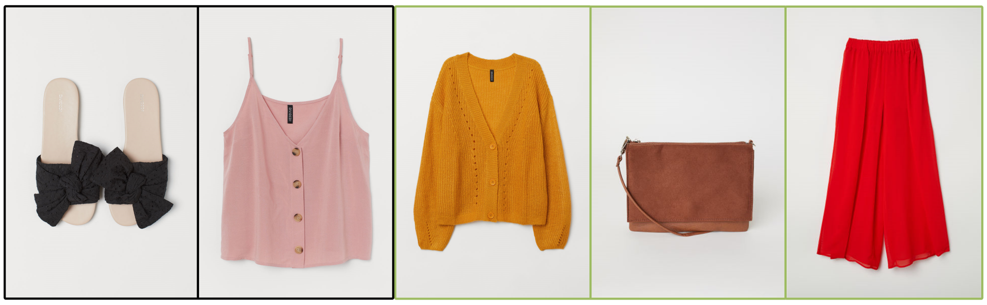

Figure B.10: *Outfit 5 - Industrial dataset*

B.2 Poyvore68k Dataset

The following outfits are generated using the items framed in black as seed of an incomplete outfit. The items framed in green are those generated by the model in order to complete the provided set of items.

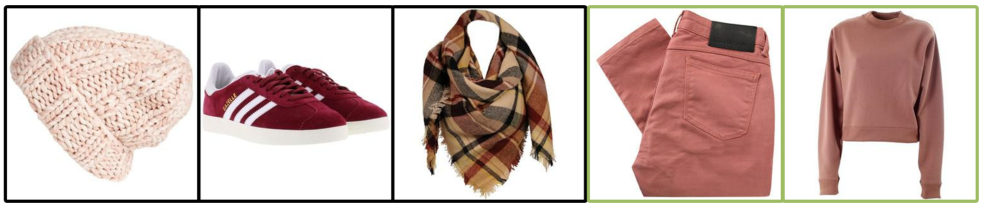

Figure B.11: *Outfit 1 - Polyvore 68k*

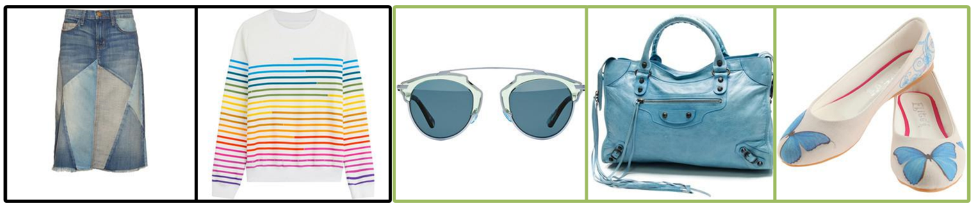

Figure B.12: *Outfit 2 - Polyvore 68k*

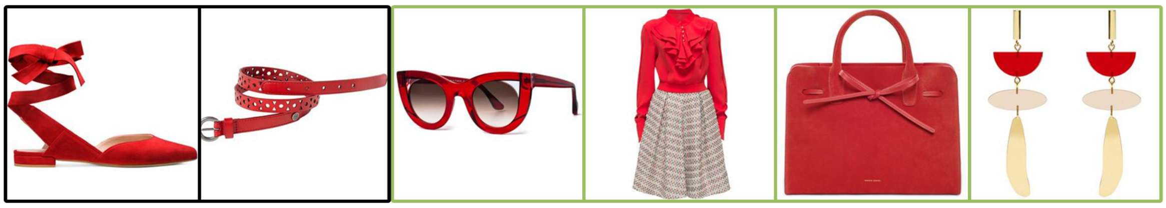

Figure B.13: *Outfit 3 - Polyvore 68k*

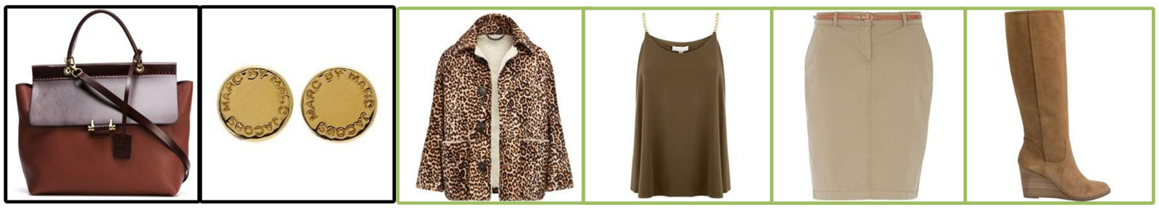

Figure B.14: *Outfit 4 - Polyvore 68k*

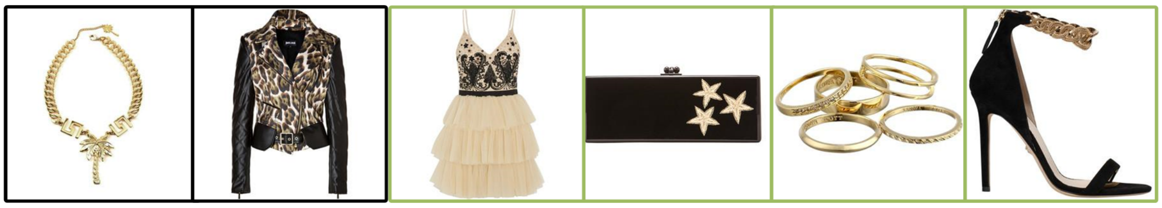

Figure B.15: *Outfit 5 - Polyvore 68k*