WardrobeAI - An Outfit Recommendation System

Lecture	Data Science SoSe 2022
Participants	Ahmad, Zeba Karin (7023017) zeah00001@stud.uni-saarland.de Singh, Vikram (7023552) visi00003@stud.uni-saarland.de Burly, Keerthana Krishnamurthy (7023672) kebu00001@stud.uni-saarland.de Athreya, Advait Ganesh (7009332) adat00001@stud.uni-saarland.de
Submission date	22.07.2022
Chair	Univ.-Prof. Dr.-Ing. Wolfgang Maaß Chair in Information and Service Systems, Campus A5 4, 66123 Saarbrücken

Contents

Chair in Information and Service Systems, Campus A5 4, 66123 Saarbrücken, Germany, Tel: +49 (0) 681 302 64736

Executive Summary

Dread. Dilemma. Distraction. Picking out a daily outfit from your wardrobe often involves these emotions. Given the abundance of clothes in our wardrobe, picking an outfit should be simpler and quicker. And not necessarily dependent on having a friend around.

We develop in this project a solution named **WardrobeAI**, a system-based console in which you add images of your clothes, and the proposed month you would like to wear the outfit. WardrobeAI displays an outfit recommendation for you consisting of 3 components - topwear, bottomwear, and footwear. We perform the prediction of outfits by integrating convolutional neural networks with domain knowledge i.e. by introducing the notion of complementary colours. We train this model on a well-documented, lightweight yet abundant dataset (see Section 2) consisting of a good cross-section of different types of clothing articles. The models are trained to recognize:

- Bodypart the clothing article most likely belongs to
- The type of dress the given article is
- Gender most associated with the article
- Conventionally the most appropriate season
- Scenarios of usage of given clothing item
- Colour of the article given in the image

We set no constraints on the user on any size of the image to be uploaded. Users may upload as many pictures, and they get added to the prediction model. We further give users the ability to modify outfit recommendations by editing parts of them or altogether reject the recommendation by removing it.

The fashion industry is one of the fast-growing sectors. On a consumer front, there exists an opportunity for AI based fashion applications to plan out outfits for daily use of consumers. The belief that AI can come up with better predictions for users can be seen by the success of apps such as ASOS, Closet Space to mention a few.

List of Abbreviations

SVD - Singular Value Decomposition

CNN - Convolution Neural Network

List of Figures

List of Tables

1 Introduction

We all at some point stand in front of our wardrobe and wonder: 'What do I wear today?' For a few, it might be a decision of the day. To help them, we have developed an outfit recommendation application named WardrobeAI. It takes images of your wardrobe as input and adds them using the WardrobeAI interface. It then generates an outfit recommendation that the user might choose to wear, edit parts of, or simply reject the image. It also indicates the type, color, season, etc. of the outfit.

1.1 Related works

Outfit recommendation is a vast field containing many approaches.
Classical approaches include:

- Finding outfits using complementary relationships. In this approach, recommending the outfit is looked at as a multi-objective task [2]. It involves having a score for the similarity between clothes, and a score for finding visual similarities between two clothing items. The overall objective is defined by function modeled as maximizing these two scores.
- We also looked at personalized outfit recommendation prediction [3]. This works on collaborative filtering techniques involving some form of matrix factorization techniques (eg. SVD). However, this requires looking at a larger superset of users interacting with recommendations, comparing given user's preferences against the above superset data and combining them to create a ranked list of suggestions.

Deep Learning approaches:

- Approaches involving CNN in outfit recommendation are multi-modal in nature [6]. They work by taking image + text descriptions in combination. The dataset that we have in consideration however is purely image-based. This doesn't fit our use-case.
- An approach that we came across used a hierarchical model of clothing [7] that learns in 2 stages - learn categories and then attribute of categories in subsequent models. This is an idea used in our models.

2 Dataset

The dataset used in the project is **Fashion Product Images (small)** which is publicly available on Kaggle. It consists of 44,000 images of different outfits including tops, bottoms, and footwear. The size of the data set is approximately 593MB. The dataset includes outfit images scraped from an e-commerce platform for e-commerce website named Myntra. It contains RGB images of size 60 x 80.
Each product's image is identified by an ID (e.g. 42431). The mapping of the products has been given along with the dataset in a '**styles.csv**' file. The CSV file has the following entries for every image:

- *Gender:* classifies clothes into men, women, boys, girls, and unisex

- *masterCategory:* classifies data into Apparel, Accessories, Footwear, etc.
- *subCategory:* defines the clothes as Bottomwear, Topwear, Watches, etc.
- *articleType:* describes the type of article, e.g. Shirts, T-Shirts, Jeans, etc.
- *baseColour:* tells the basecolor of the article represented in the image
- *Season:* defines the season for which the article can be used, e.g. Summer, Fall, etc.
- *Usage:* categorize articles into Casual, Sports, Formal, etc.
- *Year:* describes the year of the image
- *productDisplayName:* provides further information about the article

The dataset is split in three viz., the training data of 60%, validation data of 20%, and test data of 20%.

3 Data processing

For this project, we followed the below-mentioned data processing steps.

3.1 Analysis and preprocessing

The preparation of data was done in several pre-processing steps including:

- dropping unnecessary columns like *productDisplayName* and *Year* that are not needed for outfit recommendation

	id	gender	masterCategory	subCategory	articleType	baseColour	season	usage
0	15970	Men	Apparel	Topwear	Shirts	Navy Blue	Fall	Casual
1	39386	Men	Apparel	Bottomwear	Jeans	Blue	Summer	Casual
3	21379	Men	Apparel	Bottomwear	Track Pants	Black	Fall	Casual
4	53759	Men	Apparel	Topwear	Tshirts	Grey	Summer	Casual
5	1855	Men	Apparel	Topwear	Tshirts	Grey	Summer	Casual
...	...	...	...	...	...	...	...	...
44417	12544	Women	Apparel	Topwear	Tshirts	Peach	Fall	Casual
44418	42234	Women	Apparel	Topwear	Tops	Blue	Summer	Casual
44419	17036	Men	Footwear	Footwear	Casual Shoes	White	Summer	Casual
44420	6461	Men	Footwear	Footwear	Flip Flops	Red	Summer	Casual
44421	18842	Men	Apparel	Topwear	Tshirts	Blue	Fall	Casual

27265 rows × 8 columns

Figure 1: Dataset after dropping unnecessary columns

- removing redundant entries
- removing incomplete rows
- removing NaN values, and
- some sub-categories are dropped such as nightwear, socks, and innerwear.

3.1.1 Biases

We noted that dataset is skewed towards some classes. Below are few instances (given in the *exploratory_data_analysis.ipynb*) :

- Dataset is skewed towards topwear. Inside topwear category, articles are heavily skewed towards T-shirts and shirts.

- There is also a gender bias. Of all the clothing articles present, there are 16,762 articles belonging to men, while only 10,503 belong to women.

3.2 Augmentations

The following augmentations have been applied to the dataset:

- Horizontal flips
- Random rotation with a factor of 0.2, which results in an output rotated by a random amount in the range `[-20% * 2pi, 20% * 2pi]`
- Random zoom with a height factor of 0.1 and a width factor of 0.1
- Contrast with a factor of 0.2

4 Model

The approach involves a hierarchical model flow. The 2-phase recommendation system includes learning the category of cloth (determining whether the clothing article is topwear, bottomwear, or footwear). This part is handled by the '*model_category.ipynb*'. We then use the '*model_subcategory.ipynb*' which learns the attributes for each category namely: type of article, gender, usage, base colour, and conventional season.

We leverage transfer learning by using ResNet50 which is pre-trained on ImageNet dataset. The architecture of ResNet50 is illustrated in Figure 2. On top of the frozen ResNet50, we add a convolution layer and a series of dense layers. We also add dropout and L2 regularisation to make the model generalize well and deal with the problem of overfitting. The full architectures of the category model and the attribute learning model are illustrated in Figures 3 and 4.

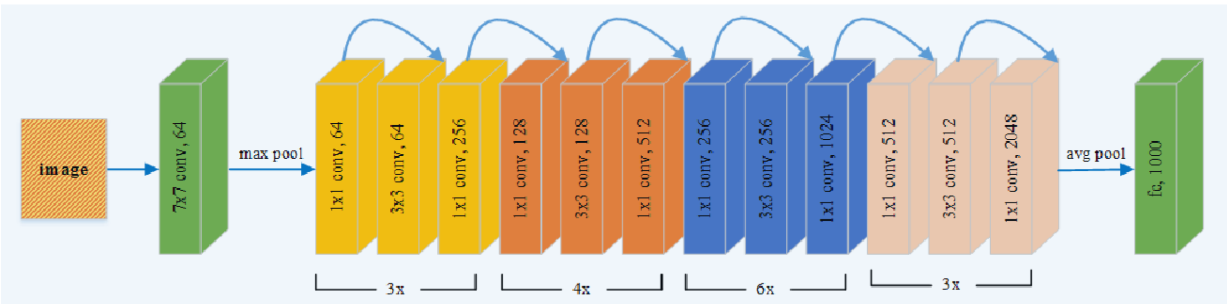


Figure 2: ResNet50 architecture

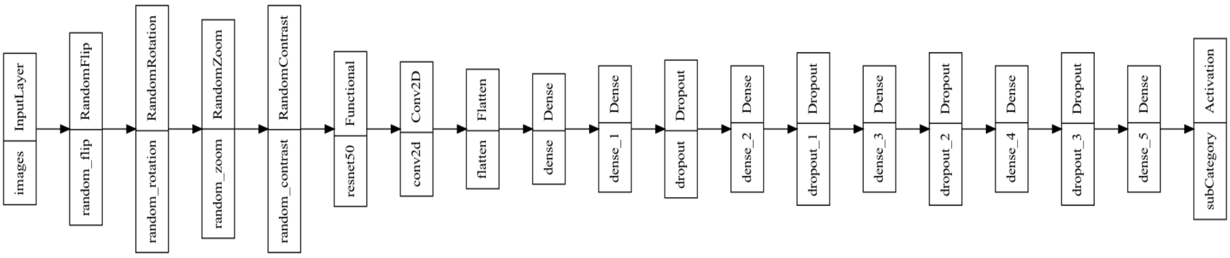


Figure 3: Model category architecture

4.1 Hyperparameter tuning

A grid search is performed over the following hyperparameter values:

- Number of epochs = {5, 10, 15}
- Dropout rate = {0.1, 0.25}
- Learning rate = {1e-3, 1e-4}
- Regularization factor = {0.1, 0.25}

The best hyperparameter for each model is mentioned in Table 1.

4.2 Model training

For this project, we have used the **Multi-class Cross-Entropy loss** function and **Adam optimizer.** Moreover, we have used the following hyperparameters for the category and sub-category models:

	Hyperparameters				
Models	**Epochs**	**Learning rate**	**Regularization (L2)**	**Dropout**	**Batch size**
model_category	15	1e-4	0.1	0.25	32
model_subcategory (Top wear)	10	1e-4	0.1	0.25	32
model_subcategory (Bottom wear)	15	1e-4	0.1	0.30	32
model_subcategory (Footwear)	5	1e-4	0.1	0.25	32

Table 1: Hyperparameters for category and subcategory models

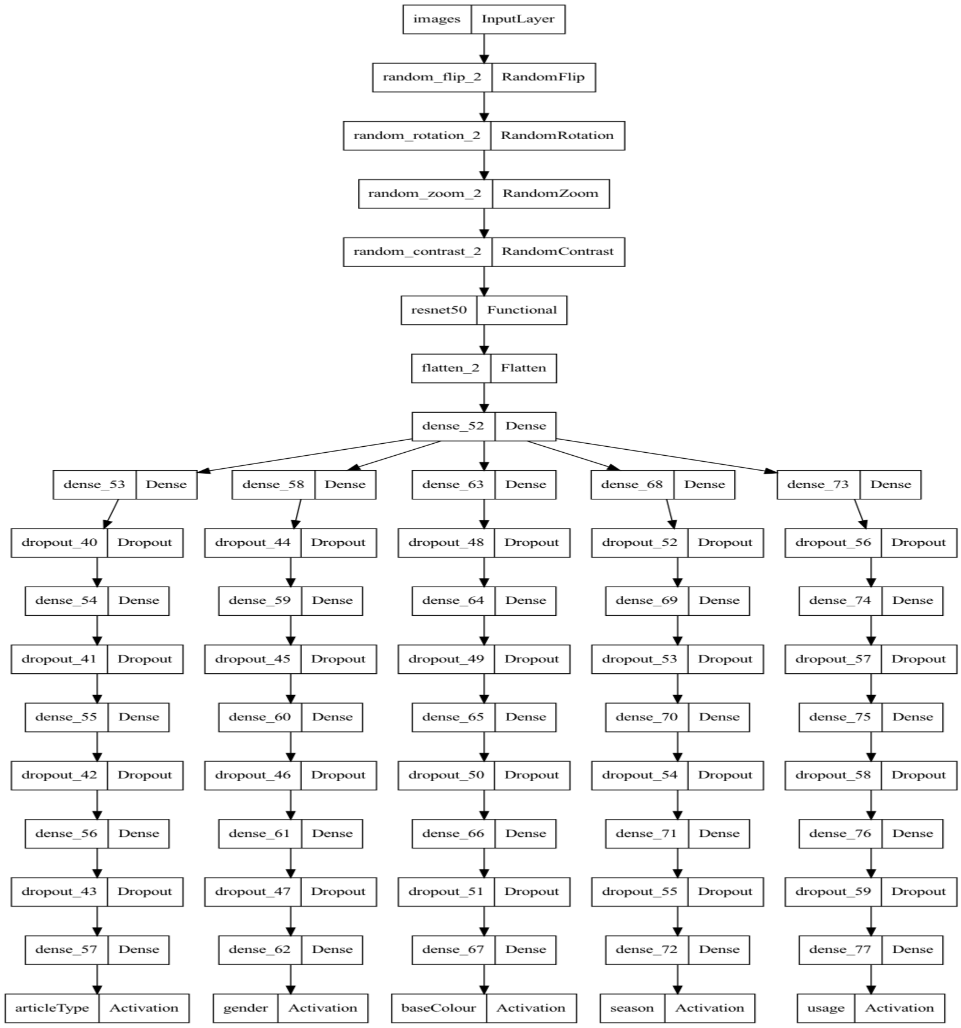


Figure 4: Model subcategory architecture

5 Results and Discussions

For *model_category*, we achieved a test accuracy of **98.93%** and a test loss of **0.1454**.

For *model_subcategory,* below are the results corresponding to the best hyperparameters.

Metrics	**model_subcategory (Topwear)**				
	ArticleType	*Gender*	*BaseColor*	*Season*	*Usage*
Loss	0.9128	0.3360	1.4551	0.7152	0.4235
Accuracy (%)	72.20	87.92	54.30	67.38	85.97
	model_subcategory (Bottomwear)				
Loss	1.4028	0.5488	1.5307	0.8568	0.6476
Accuracy (%)	52.54	77.15	50.20	59.38	73.83

	model_subcategory (Footwear)				
Loss	1.0893	0.5764	1.3982	1.0426	0.4871
Accuracy (%)	55.10	80.26	57.51	53.40	77.03

Table 2: Loss and accuracy results for subcategory of models

We note that the Topwear model performs best with an accuracy of 72.20%. The bottomwear and footwear model perform poor relatively to topwear, which we think is due to the imbalance in the Topwear data as opposed to the other two.

6 Deployment

The end-user interface is a window application based on PyQT5. The '*ui_module.py*' has a predict function to predict the type of image that is uploaded by the user and is appended to a list of tops, pants, or shoes based on classification. While uploading the picture, the size of the image is resized to 60 x 80 to make it compatible with our trained models. After the images are added to different lists, the '*generate*' function recommends outfit combinations based on the season of the selected month. This is done by invoking '*recogniton_module.py*' to: identify the colours, evaluate complementary colours, call models to generate predictions, and output list of outfit recommendations. PyQT5 widgets and core elements are imported to build the interface and its elements. The application is a proof of concept, it requires exception handling for boundary use cases.

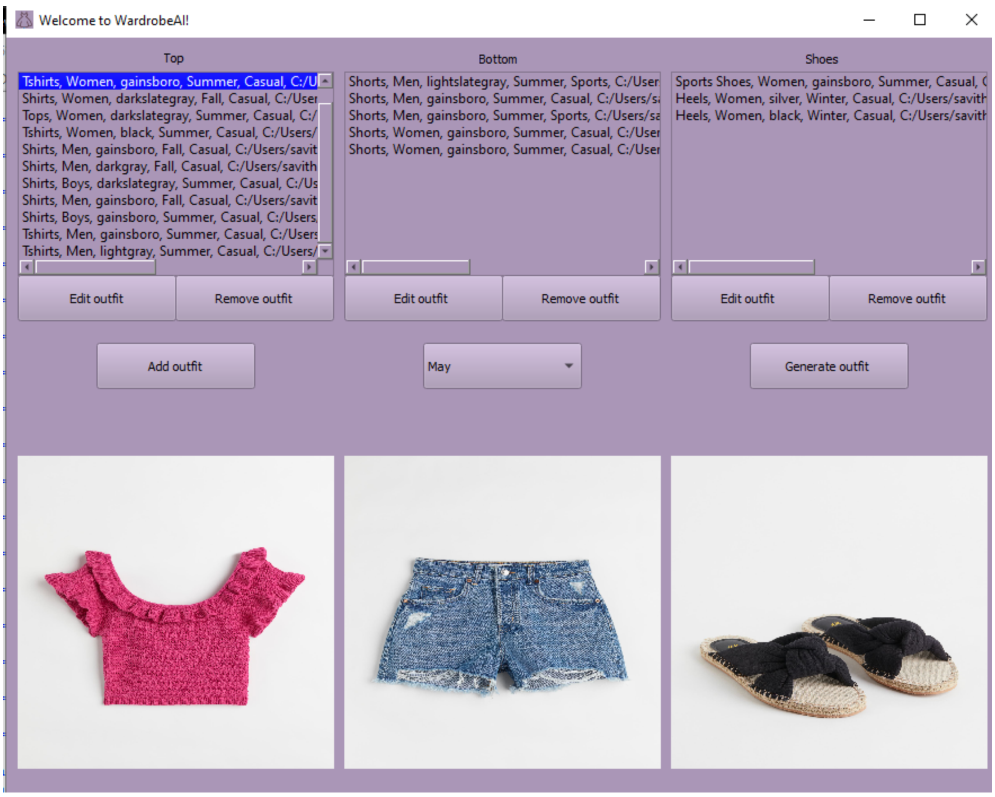


Figure 5: Outfit recommendation example for May (summer)

Our model also has some limitations, as it recommends only 3 broad categories i.e., topwear, bottomwear, and footwear. Accessories and watches are apparel that could be predicted in the future. It is a windows-based application and the user is expected to use it with only a desktop, which is not convenient for daily usage. For future work, the implementation can be extended to a smartphone application. The main window which provides different recommendations can be seen in Figures 5 and 6.

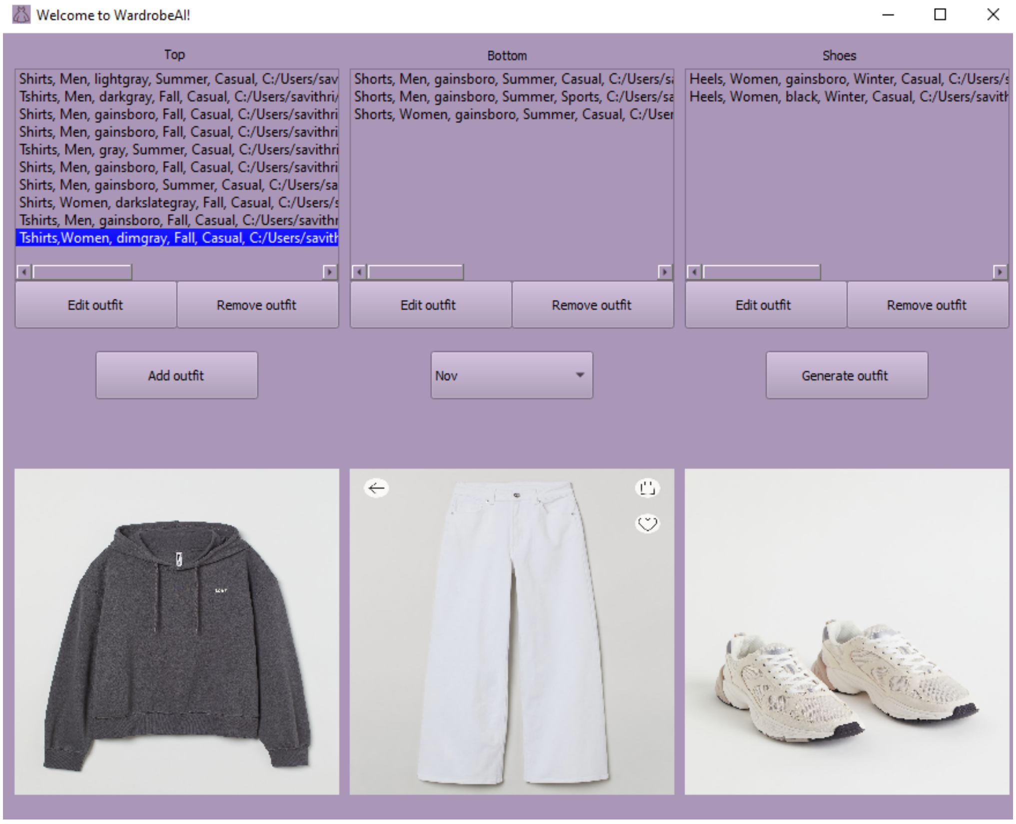


Figure 6: Outfit recommendation example for November (winter)

The code for the project is available at GitHub repository: https://github.com/vsingh1998/WardrobeAI

Bibliography

1. A Review of Modern Fashion Recommender Systems [Deldjoo et al.. 2022]
2. Image-Based Recommendations on Styles and Substitutes [McAuley et al.. 2015]
3. Collaborative Fashion Recommendation: A Functional Tensor Factorization Approach [Hu et al.]
4. AI-powered recommendation system: Building an AI-Powered Outfit Recommendation System With Dataiku
5. Learning Fashion Compatibility with Bidirectional LSTMs [Han et al.]
6. Fashion Outfit Generation for E-commerce [Bettaney et al., ECML PKDD 2020.]
7. Fashion Sensitive Clothing Recommendation Using Hierarchical Collocation Model [Zhou et al., MM '18: Proceedings of the 26th ACM international conference on Multimedia]
8. https://github.com/KefanPing/Outfit_Recommendation_Project/blob/main/models/train_module.py
9. https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-small
10. https://realpython.com/build-recommendation-engine-collaborative-filtering/
11. https://www.tensorflow.org/tutorials/images/classification
12. https://towardsdatascience.com/how-to-make-deep-learning-models-to-generalize-better-3341a2c5400c

Appendix

```
.
├── data
│   ├── images # A directory containing the dataset images
│   └── styles.csv # A csv file containing the annotations for the
images
├── models # A directory containing trained models and images of the
model architectures
│   ├── models # A google drive folder which contains our trained
models
│   │   ├── model_category # A model that distinguishes tops,
bottoms, and shoes
│   │   ├── model_bottomwear # A model that recognizes the type,
color, gender, season, and usage of bottoms
│   │   ├── model_footwear # A model that recognizes the type,
color, gender, season, and usage of shoes
│   │   └── model_topwear # A model that recognizes the type, color,
gender, season, and usage of tops
│   ├── model_bottomwear.png # Architecture of the model_bottomwear
│   ├── model_category.png # Architecture of the model_category
│   ├── model_footwear.png # Architecture of the model_footwear
│   └── model_topwear.png # Architecture of the model_topwear
├── ui_images # A directory containing images used for the ui_module
├── exploratory_data_analysis.ipynb # A notebook containing the
analysis of our dataset and corresponding inferences
├── model_category.ipynb # A notebook containing the training of the
model_category
├── model_category_hyperparam.ipynb # A notebook containing code for
hyperparameter tuning for model_category
├── models_subcategory.ipynb # A notebook containing the training of
the model_bottomwear, model_footwear, and model_topwear
├── models_subcategory_hyperparam.ipynb # A notebook containing code
for the hyperparameter tuning for subcategory models
├── recognition_module.py # A module that contains functions and
classes to generate the GUI
├── ui_module.py # A module to run the application
├── utils.py # A module containing helping functions for model
training
├── README.md # readme file
└── requirements.txt # packages used
```

Figure 7: Code structure

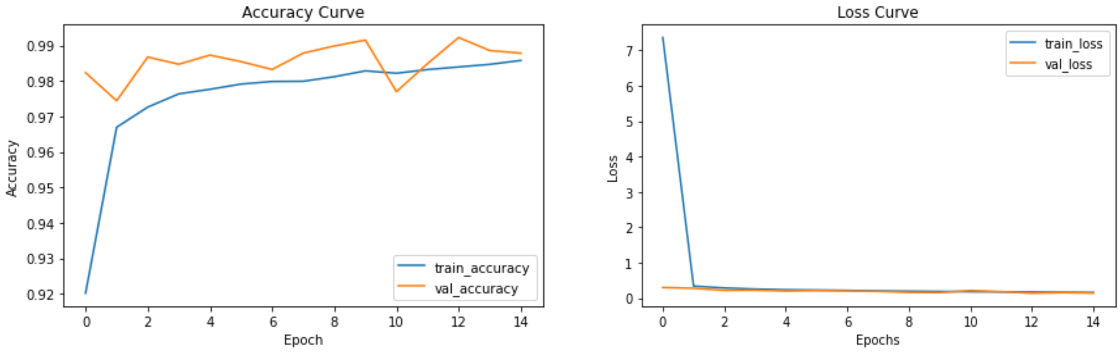


Figure 8: Accuracy and loss curves for model category

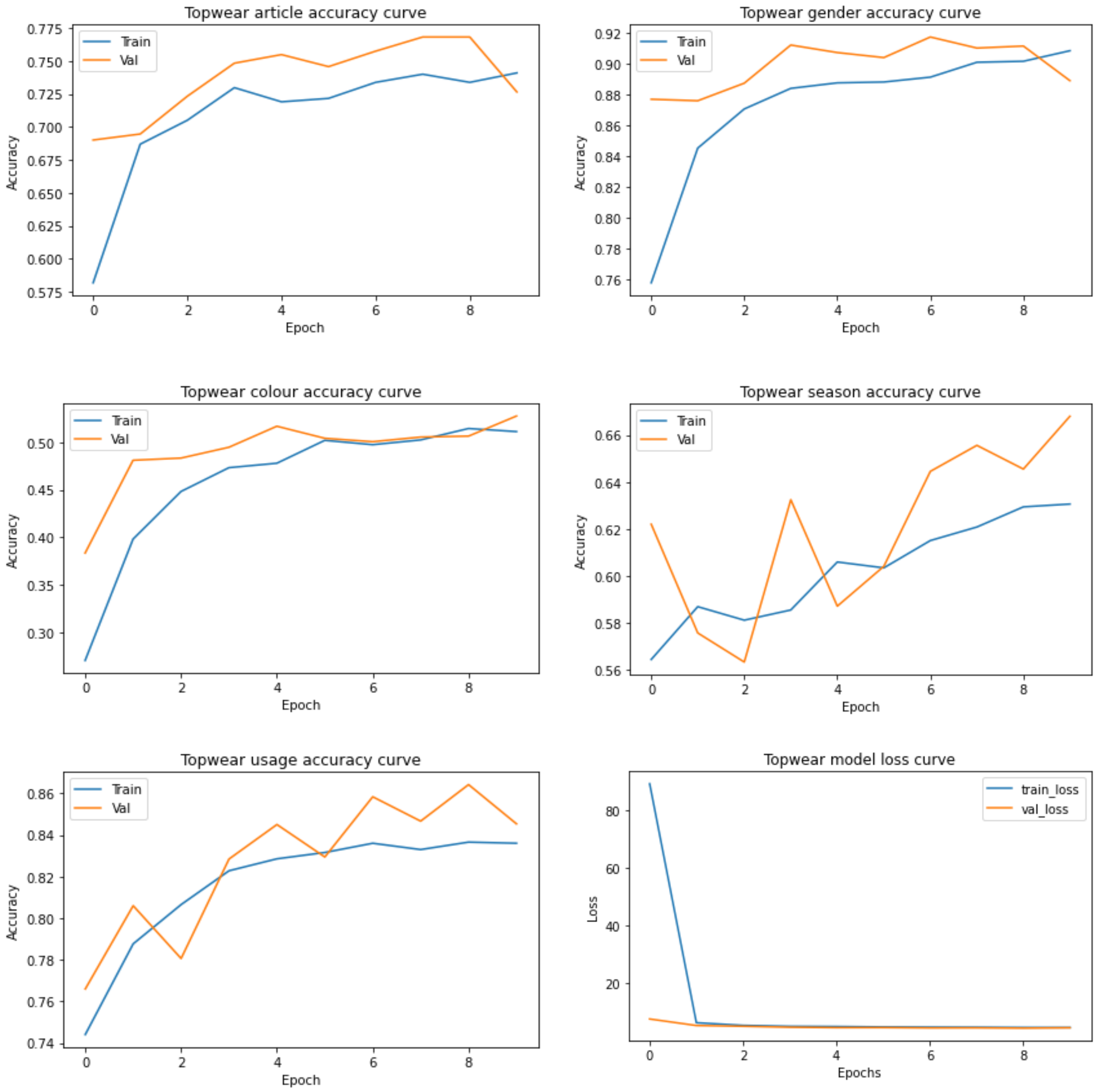


Figure 9: Topwear model curves (left to right) - article, gender, colour, season, usage, and loss

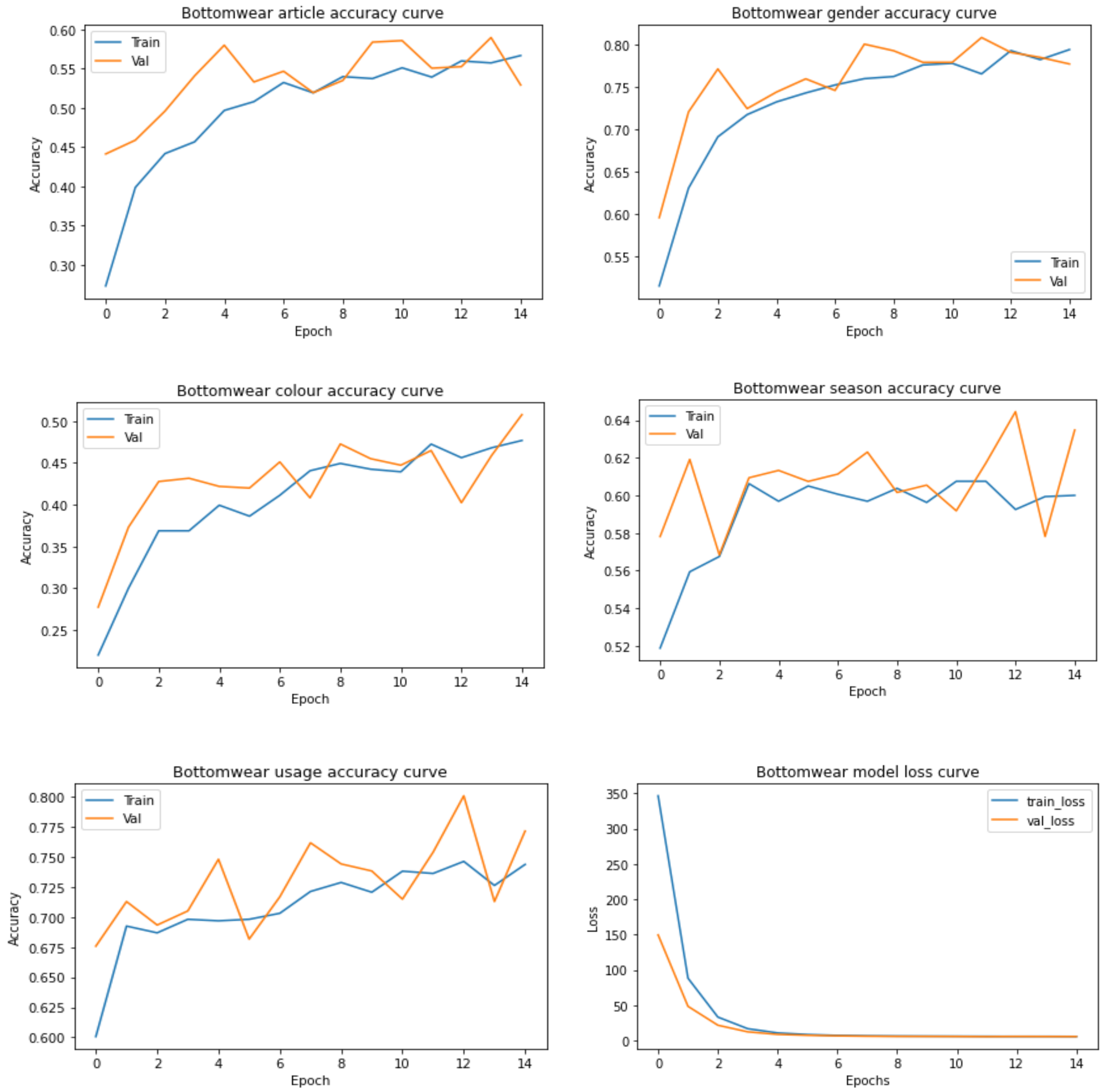
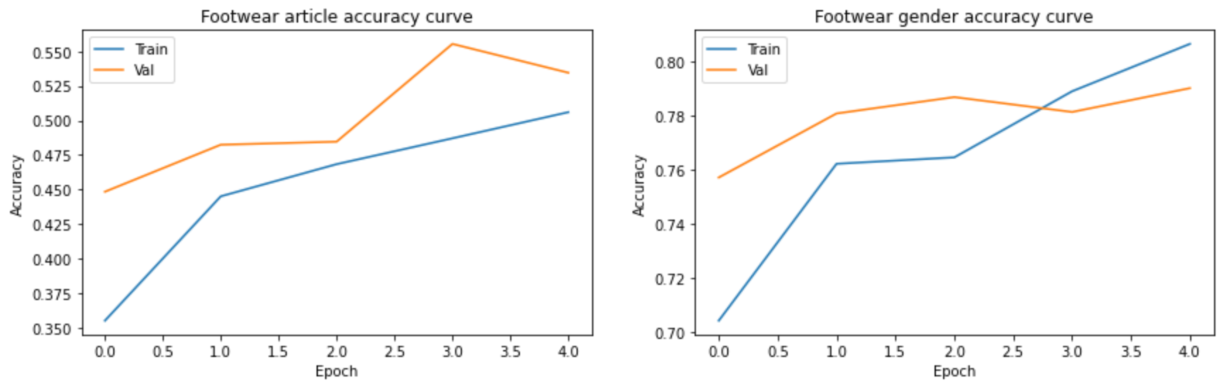


Figure 10: Bottomwear model curves (left to right) - article, gender, colour, season, usage, and loss



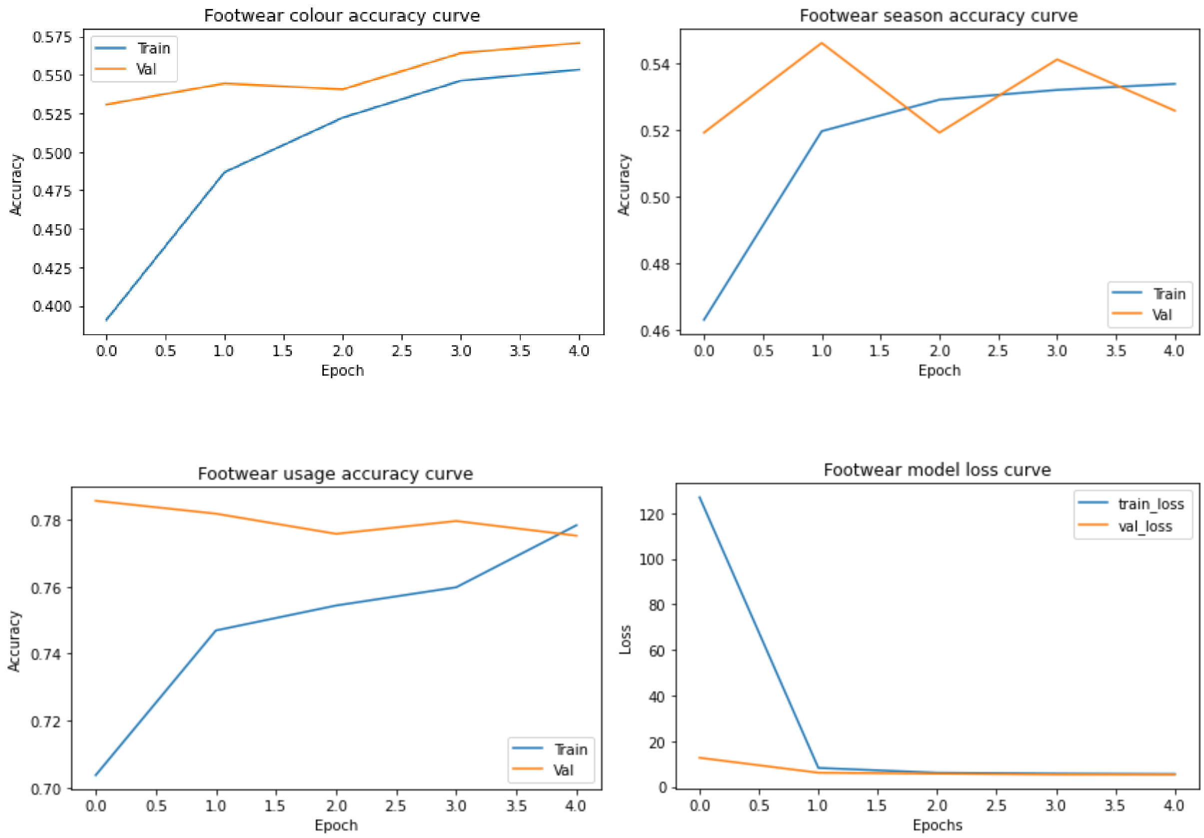


Figure 11: Footwear model curves (left to right) - article, gender, colour, season, usage, and loss

Declaration of Authorship

I affirm that I have produced the work independently, that I have not used any aids other than those specified and that I have clearly marked all literal or analogous reproductions as such.

Saarbrücken, 22.07.2022

Location, Date

Ahmad, Zeba Karin

Singh, Vikram

Burly, Keerthana Krishnamurthy

Athreya, Advait Ganesh