information

MDPI

Article

Multimodal Sequential Fashion Attribute Prediction

Hasan Sait Arslan [1,2,3,*], Kairit Sirts [1], Mark Fishel [1] and Gholamreza Anbarjafari [2,4]

1 NLP Group, Institute of Computer Science, University of Tartu, 50090 Tartu, Estonia; kairit.sirts@ut.ee (K.S.); fishel@ut.ee (M.F.)
2 iCV Lab, Institute of Technology, University of Tartu, 50090 Tartu, Estonia; shb@ut.ee
3 Rakuten Fits.Me, 50090 Tartu, Estonia
4 Faculty of Engineering, Hasan Kalyoncu University, Gaziantep 27900, Turkey
* Correspondence: hasan@icv.tuit.ut.ee

Received: 15 July 2019; Accepted: 29 September 2019; Published: 3 October 2019

Abstract: We address multimodal product attribute prediction of fashion items based on product images and titles. The product attributes, such as type, sub-type, cut or fit, are in a chain format, with previous attribute values constraining the values of the next attributes. We propose to address this task with a sequential prediction model that can learn to capture the dependencies between the different attribute values in the chain. Our experiments on three product datasets show that the sequential model outperforms two non-sequential baselines on all experimental datasets. Compared to other models, the sequential model is also better able to generate sequences of attribute chains not seen during training. We also measure the contributions of both image and textual input and show that while text-only models always outperform image-only models, only the multimodal sequential model combining both image and text improves over the text-only model on all experimental datasets.

Keywords: Fashion E-Commerce; product attribute prediction; multimodal classification; sequential prediction; CNN; RNN

1. Introduction

Modeling garment data is a theme that has achieved considerable attention during recent years because of the developments in deep-learning methods. There is now an assortment of research on various tasks in garment modeling and redrawing in 2D or 3D environments [1], recognizing the garment [2], parsing the semantic parts of the garment in the image [3], retrieving the garment based on specified attributes [4], and making garment recommendations based on previous choices [5]. An expanding number of papers are concentrating on garment data retrieval [6–8], which is a fundamental task relevant for the customers of the web-based fashion businesses.

To enable sorting, standardizing and searching fashion products on e-commerce retail platforms, garments must be supplied with relevant product attributes, such as product category and type, gender information, etc. However, this index data is often incomplete as the merchants may fail to provide all relevant information together with the product. Considering many fashion products offered on e-commerce platforms, adding the missing attributes manually is not feasible. As a result, merchants can lose potential sales because due to missing attributes, some relevant fashion items may not be returned in response to a potential customer's search query. Thus, systems that can infer the missing attributes of garment products are of practical importance.

Several pieces of information, such as images and textual descriptions, are typically available for fashion items sold on e-commerce platforms and this data can be used to extract missing attribute information. Consider an example in Figure 1 that shows a preview of an item search form of T-shirts and Tank Tops on Rakuten.com, a Japanese e-commerce company similar to Amazon.com. For each product the page shows an image and a title that potentially contain information about the item's missing attributes. For instance, the values of the attributes highlighted in the upper right part of Figure 1, such as *American Casual* (Fashion Taste), *Short Sleeves* (Sleeve Length), and *Medium* (Clothing Length) can be inferred from such multimodal data.

Extracting missing attributes from the textual and image data of fashion products is challenging in several respects. Product images contain visual features of several attributes (for instance, the Sleeve Length is short and the Fashion Taste is American Casual) and the system must learn to distinguish between visual features characterizing these different attributes. The textual data comprises unstructured image titles or product descriptions that may or may not contain relevant attribute information. The system must be able to analyze unstructured textual data and make proper inferences based on it.

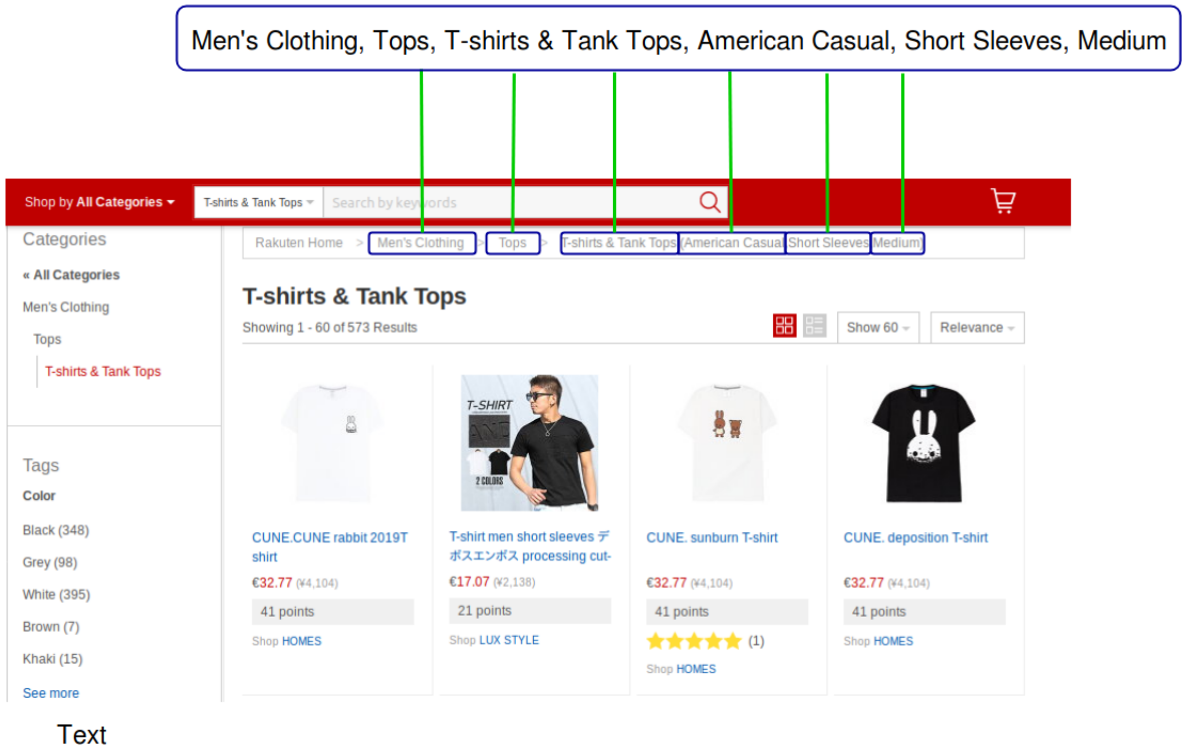


Figure 1. Snapshot of a T-shirt search page taken from Rakuten.com.

Most previous approaches have addressed the attribute extraction as a multitask classification problem where the values of different attributes are predicted independently of each other [9–11]. While such an independence assumption is justified for certain attribute categories, it provides the system too many degrees of freedom in other cases. For instance, the value of the attribute of the product type constrains the set of possible values for the product sub-type attribute: if the product type attribute value is *Top*, then *T-shirt* would be a legal value for the sub-type attribute while *Skirt* would be not as skirts do not belong to the type of *Top* garment.

This observation suggests that if the available attributes are organized in a tree-like structure, where the value of each previous node constrains the possible values in the next branch of the tree, attribute prediction can be formulated as a sequential prediction problem where the value of each next attribute is dependent on the values of the previously predicted attributes. While previous work has explored sequential product attribute prediction based on textual data [12,13], we are not aware of such work on multimodal input. Also, neither of these previous works compared the sequential prediction with the multitask formulation.

In this paper, we propose to formulate the multimodal product attribute prediction as a sequential prediction problem to enable the model to learn to capture the dependencies between different attribute values. We compare our model to three different non-sequential task formulations. The following list summarizes all models:

(1) **MultiClass Prediction:** This is a simple baseline where the values of all attributes of an item are concatenated and predicted as a single united label. This formulation is suitable when the number of different attribute value combinations is not too large; however, it does not allow prediction of value combinations that were not seen during training.

(2) **MultiLabel Prediction:** Here the problem is formalized as a multilabel classification task where the value of each attribute is considered to be a different label and thus, each input instance is assigned multiple labels. We experiment with several multilabel models, starting with a simple binary model and including several multiclass multilabel models similar to the multitask models used in previous work [9–11].

    (a) **Binary MultiLabel Prediction:** This is the most basic multilabel baseline where the model predicts the presence or absence of each attribute value.

    (b) **MultiTask Prediction:** This is the multiclass multilabel model that adopts a multiclass classifier for predicting one value for each attribute.

    (c) **Attentive MultiTask Prediction:** This is the multitask model with an attention component between the image, text and each label.

    (d) **Self-Attentive MultiTask Prediction:** This is the attentive multitask model with an additional self-attention component between the multiclass attribute classifiers.

(3) **Sequential Prediction:** This formulation predicts each attribute value in sequence and each next prediction is conditioned on the previously predicted attribute values. We hypothesize that this formulation is the best in cases where the attribute categories are organized into a tree-like structure. According to our knowledge, while the sequential prediction approach is not novel in the attribute prediction domain [12,13], the multimodal sequential prediction has not been explored before.

We tested our models on three multimodal datasets, containing both image and text:

1. Rakuten dataset, which is an in-house proprietary dataset obtained from Rakuten.com,
2. Kaggle Fashion-Product-Images Dataset, where each product item is supplied with the image, title, and several attribute values of the product, and
3. Amazon Product Dataset, which includes Amazon product metadata, such as titles, descriptions, category information, price, brand, image URLs and image features belonging to product data profiles.

For encoding the data, we use the CNN-based Resnet-50 [14] architecture for product images, and RNN-based RNMT+ model [15] for textual data. We use this encoder in combination with all output prediction approaches on our experimental datasets. In Rakuten and Kaggle datasets, the attributes are organized in a chain-like structure and thus we expect the sequential modeling to perform best on those two datasets. For the Amazon dataset, we do not know the category structure of the attribute chains and thus on that dataset, we can only apply the MultiClass and the Sequential prediction approaches. On all datasets, we expect Sequential and MultiLabel approaches to outperform the MultiClass model.

Our contributions in this paper are:

(1) We propose the sequential modeling approach for multimodal fashion attribute prediction and show that it outperforms other methods on datasets where the product attributes are organized in a tree-like structure. You can find the source code for reproducing our experiments at https://github.com/saitarslanboun/MultimodalSequential
(2) Besides multimodal models, we also experiment with text-only and image-only models and show that while text-only models always outperform image-only predictions, only the sequential model always benefits from multimodal input.
(3) We perform extensive experiments in real-life datasets to demonstrate the sequential model's efficacy and show that it is also better than the multilabel models in predicting novel attribute chains.

2. Related Work

All our models are multimodal deep-learning classification models and there is quite a lot of research in this area [16–19]. Here we focus our review of recent work on category/attribute extraction in the e-commerce/fashion domain using both multimodal and only image or only text input.

2.1. Attribute Prediction Based on Multimodal Data

There is little previous work on multimodal attribute extraction for commercial products and none of them has explored sequential modeling of attribute value chains. Cardoso et al. [9] predict attribute values for fashion products using both product images and textual information as input on a dataset driven from the ASOS e-commerce marketplace. They encode both image and with CNNs and predict the value of each attribute independently of each other conditioned on the same input text and image representation, similar to the multitask approach also used in our presented work. However, whereas we predict the values of all attributes of a product item together, which to some extent enables the multitask model to learn the dependencies between different attribute values, they only learn one attribute of each product item at one time, thus diminishing the advantages of the multitask model in capturing the dependencies between different attribute values. Also Logan et al. [10] use a multitask model, predicting the values of several attributes of generic commercial products on a multimodal dataset. Similar to Cardoso et al. [9], they also only predict a single attribute value for each product item at a time. They use an additional input called "query" to encode the category of the predicted attribute that is used to guide the model to predict only those values relevant to that category. Both images, text, and queries are encoded with CNNs, and they are fused in a fusion layer. They experiment with two different fusion techniques; the first technique just concatenates the three encoded representations, and the second technique uses a special gating mechanism.

Zahavy et al. [20] focus on fusing the representations of multimodal inputs for the category prediction on a general-domain commercial product dataset derived from the Walmart.com e-commerce website. Both image and text are encoded with CNN, and their representations are fused to predict the value of the product category in a multiclass setting. They experiment with two different fusion techniques: feature-level fusion and decision-level fusion. The feature-level fusion is based on end-to-end training, where the output representation vectors from image CNN and text CNN are ensembled for the subsequent prediction. In the decision-level fusion, the image and text models are trained separately, and their predictions are fed through a policy layer which decides between image CNN and text CNN prediction.

2.2. Attribute Prediction for Fashion Based on Image Data

Inspired by the recent popularity of CNN-based deep-learning methods for image processing, there have been several approaches proposed for a fashion product classification and attribute prediction based on image data. Chen et al. [21] predict attribute values for fashion products on an image dataset derived from the Taobao e-commerce marketplace. They encode images with a CNN with spatial attention and predict the value of each attribute independent of each other, conditioned on the same input image representation. This is similar to the multitask approach used in our work. Schindler et al. [22] predict the product category of fashion images using a multiclass model. Their dataset is derived from online e-commerce companies such as Asos-EU, Farfetch or Zalando. They analyzed five different CNN architectures pretrained with ImageNet [23].

Liu et al. [24] introduced the DeepFashion dataset with the FashionNet model, which learns fashion features for general usage in the fashion domain. Several subsequent works have used this dataset to train models for various fashion-related prediction tasks. For example, Li et al. [25] perform clothing classification using the multiclass approach. They make use of CNN, HOG [26] and color histograms to learn garment classification from different image representations. Li et al. [11] predict product categories and attributes. They use multiclass learning for category prediction and a binary multilabel model to predict multiple attributes for an item.

2.3. Product Categorization Based on Text

Hiramatsu and Wakabayashi [12] and Li et al. [13] experimented with the encoder-decoder network to predict attribute chains on a dataset sampled from the Rakuten.com product catalog. Their work is very similar to us as they also predict the values of attributes sequentially. However, whereas our work leverages multimodal data, their dataset only contains text. Also, neither of those works compare to the neural multitask baseline as we do in this work. Hiramatsu and Wakabayashi [12] compare their sequential model to a multiclass Random Forest baseline, while Li et al. [13] only experiment with sequential models. On the same dataset Lin et al. [27] proposes various other methods for predicting product attribute chains.

Krishnan and Amarthaluri [28] compare CNN and RNN encoders for categorizing products on a textual dataset driven from Walmart's e-commerce marketplace, where each product data includes both structured product features as well as unstructured textual information. OpenTag [29] casts the problem of attribute prediction as a sequence tagging task, using an end-to-end neural model comprising a bidirectional LSTM, CRF and an attention mechanism for extracting attribute values from the textual data of item profiles such as title and description. The advantage of this approach is that the set of attribute values does not have to be fixed in advance and the model can predict new attribute values that it has never observed during training, allowing also multi-word attribute values and multiple attributes per item if appropriate. The downside lies in the assumption that the relevant attribute values have to be explicitly represented in the text. Hsieh et al. [30] use a more traditional K-means clustering method for generating product hierarchies based on text encoded with word embeddings.

2.4. Fashion Product Categorization Using Noisy Data or Transfer Learning

Several works have tackled the problem of fashion product categorization using semi-supervised learning techniques or noisy data. For example, Inoue et al. [31] predict attribute values for fashion images on the Fashion550k dataset where only a subset of the database is human-verified. Their proposed model learns to clean the noisy labels while simultaneously also predicting the attributes.

Dong et al. [32] classify unconstrained images taken from Google street views using a two-level transfer learning method. The first level is a multitask learning model trained on images from the X-Domain benchmark dataset [33]. On the second level, they fine-tune the trained model for learning the street-view images. They perform transfer learning by matching each garment on street-view images with an equivalent garment from the X-Domain dataset.

Corbiere et al. [8] proposed an unsupervised method for garment category prediction using a dataset crawled from e-commerce web pages. The dataset comprises images and textual descriptions of products, and the textual descriptions serve as noisy labels. Their model encodes images with a CNN and is trained to predict the words in the description that serve as the most likely candidates for representing the garment category.

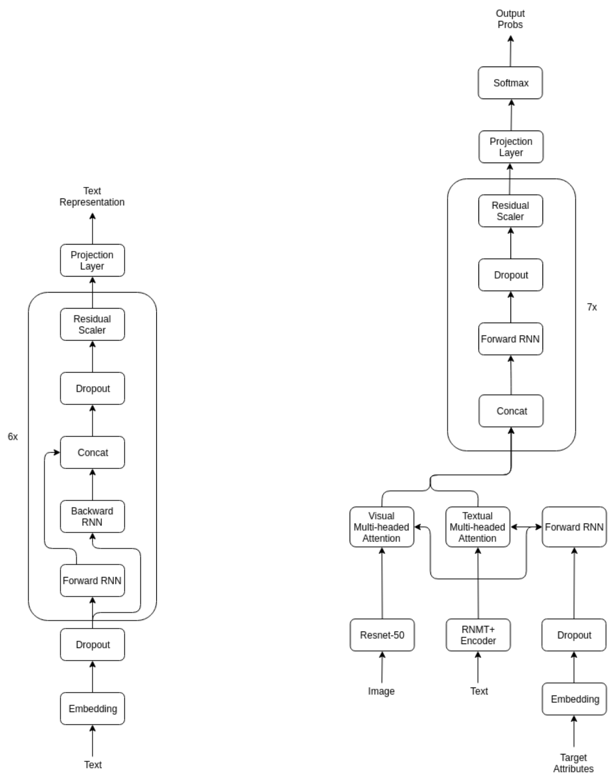
3. The Sequential Attribute Prediction Model

In this section, we describe the sequential architecture used in our experimental work. First, we will describe the image and text encoders that are the same for all models. Then we describe the sequential prediction model.

3.1. Encoding Image and Text

For decoding images, we use the Resnet-50 CNN model [14], which consists of 50 CNN layers that all extract latent feature vectors from the image segments of increasing size. The last layer outputs the representation of the whole image concerning different object labels. Because our models might benefit from a more fine-grained image representation, we extract the image features from a lower convolutional layer of Resnet. In this way, we obtain a separate feature vector for different image segments. In particular, we extract the feature vectors from the 47th layer where the feature vector size is 1024 and the image is divided into 14×14 segments. These feature vectors are organized into an image representation matrix I.

For encoding text, we use the recurrent encoder (see Figure 2a) from the RNMT+ model [15]. RNMT+ is a state-of-the-art encoder-decoder architecture designed for machine translation and neural sequence generation. In the RNMT+ encoder, there are six bidirectional recurrent layers. (We use GRU cells). For each bidirectional layer, we concatenate the outputs of the forward layer and the backward layer before feeding into the following layer. Starting from the third layer, the concatenated outputs are passed through the residual connections. After each bidirectional layer (including residual connections), we apply layer normalization [34], and dropout. The output of the text encoder is a matrix containing a feature vector for each input word. This matrix is subsequently called a text representation matrix, denoted by T.



(**a**) RNMT+ encoder (**b**) Sequential architecture

Figure 2. RNMT+ encoder and the sequential architecture.

3.2. Sequential Model

The Sequential model (see Figure 2b) is an encoder-decoder generation model that predicts the output attribute values one by one. It uses as input the encoders for image and text described previously in Section 3.1 and has a very similar architecture to the RNMT+ sequential decoder [15].

Denote an attribute chain by $Y = y_1, y_1, ..., y_n$, where y_t are the values of each attribute category in the chain and n is the total number of attribute values in the chain. The joint probability of the attribute chain can be decomposed into the conditionals:

$$p(Y) = \prod_{t=1}^{n} p(y_t | y_1, ..., y_{t-1}, I, T) \tag{1}$$

$I \in \mathbb{R}^{194 \times d}$ is the image representation matrix computed by the image encoder containing a $d = 1024$-dimensional vector for each of the $14 \times 14 = 196$ image segments. $T \in \mathbb{R}^{m \times d}$ is the text representation matrix computed by the text encoder, where m is the number of words in the textual input. In the following subsections we describe the components of the sequential decoder in more detail.

3.2.1. Input and Output Formats of the Labels

The input and output format of the attributes follows the standard language modeling input-output format (see Figure 3). The generation of an attribute sequence starts with a beginning-of-sequence symbol <BOS>, and ends with an end-of-sequence symbol <EOS>. Similar to language modeling, the output sequence is equivalent to the input but offset by one, the input sequence lacks the last <EOS> item while the output sequence lacks the first <BOS> item.

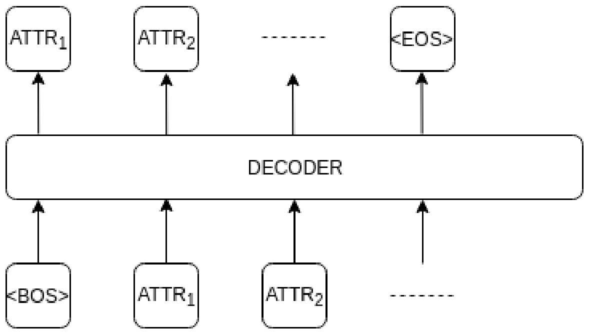


Figure 3. Input format of the sequential decoder.

3.2.2. Embedding Layer

The attribute embedding layer captures the context of an attribute value within the set of all possible attribute values. The context information captures semantic similarities and relations with other attribute values. Each label is represented with a one-hot encoded vector. The embedding layer projects the one-hot encoded vectors into d-dimensional attribute embedding vectors. We use $d = 1024$-dimensional embedding vectors.

3.2.3. Decoder Layers

The decoder consists of 8 GRU [35] layers and the output of each decoder layer is fed as input to the next decoder layer. Attribute embeddings are input to the first decoder layer. Starting from the third layer, the output of the decoder layers is additionally fed through the residual layers. Both dropout and layer normalization are applied after each decoder layer. After the first decoder layer, the attentional context vectors are computed. They are concatenated with the decoder layer outputs and fed through all the decoder layers, except the first.

In each GRU layer, the next hidden state is computed as:

$$\mathbf{s}_t = \text{GRUCell}(\mathbf{x}_t, \mathbf{s}^l_{t-1}) \tag{2}$$

where $\mathbf{x}_t$ is input to that GRU layer, and $\mathbf{s}^l_{t-1}$ is the previous hidden state of the lth GRU layer. We initialize the initial hidden state vector $\mathbf{s}^l_0$ with $\mathbf{0}$.

At each prediction step, the embedding of the last predicted attribute value y_{t-1} is fed as input to the first GRU layer to compute the current hidden state vector $\mathbf{s}^1_t$. Then, two attentional context vectors are calculated using multi-headed attention, $\mathbf{c}^I_t$ and $\mathbf{c}^T_t$, where $\mathbf{c}^I_t$ is the alignment information between $\mathbf{s}^1_t$ and the input image representation I, and $\mathbf{c}^T_t$ is the alignment information between $\mathbf{s}^1_t$ and the input text representation T. The alignment vectors $\mathbf{c}^I_t$ and $\mathbf{c}^T_t$ are concatenated with the hidden state vector $\mathbf{s}^1_t$ and fed as an input to the next GRU layer. The rest of the GRU layers $l = 2 \ldots L$ use the concatenation of the output hidden state of the previous GRU layer $\mathbf{s}^{l-1}_t$, $\mathbf{c}^I_t$, and $\mathbf{c}^T_t$ as input. The context vectors $\mathbf{c}^I_t$ and $\mathbf{c}^T_t$ are calculated only after the first GRU layer.

3.2.4. Multi-Headed Attention

The multi-headed attention is used to compute attention over the source text representation T and the source image representation I. The multi-headed attention component has several attention heads, each of which can focus on different part of the input representation.

Following Vaswani et al. [36], the context vector of each attention head is computed with the scaled dot-product attention:

$$\text{head} = \text{Attention}(\mathbf{q}, K, V) = \text{softmax}\left(\frac{\mathbf{q}K'}{\sqrt{d_K}}\right)V, \tag{3}$$

where $\mathbf{q}$ is the query vector, K is a set of key vectors and V denotes the values corresponding to the keys. K' denotes the transpose of the keys matrix and d_K is dimensionality of the key vectors. In our setting, $\mathbf{q}$ is the hidden state vector output by the first GRU layer $\mathbf{s}_t^1$, K and V are both either image representation matrix I in the visual multi-headed attention layer or textual representation matrix T in the textual multi-headed attention layer.

The final context vector of the multi-headed attention is formed by concatenating the outputs of the different attention heads and applying a linear projection with weights W^O and bias b^O on top of it:

$$\text{MultiHead}(\mathbf{q}, K, V) = \text{concat}(\text{head}_1, ..., \text{head}_h)(W^O)' + b^O \tag{4}$$

To enable the multi-head attention to focus on different parts of the input, for each head, the query, keys and values are projected into a d'-dimensional subspace, where $d' = d/h$ with d being the original dimensionality of $\mathbf{q}$, K and V, and h is the number of heads. Thus, each attention head is computed as:

$$\text{head}_i = \text{Attention}(\mathbf{q}(W_i^q)' + b_i^q, K(W_i^K)' + b_i^K, V(W_i^V)' + b_i^V), \quad i = 1 \dots h, \tag{5}$$

where W_i^q, W_i^K and W_i^V are the projection matrices, and b_i^q, b_i^K and b_i^V are the bias vectors for the query, keys and values, respectively, for each head $i = 1 \dots h$, and W' denotes the transpose of the matrix W.

The visual multi-headed attention context vector $\mathbf{c}_I$, and the textual multi-headed attention context vector $\mathbf{c}_T$ are concatenated to the inputs for all subsequent GRU layers. All weight matrices and bias vectors of the multi-headed attention are trainable parameters and are learned together with the rest of the model parameters.

3.2.5. Attribute Value Prediction

The output of the last decoder layer s_t^L is fed through projection and softmax layers to compute the probability distribution over all possible attribute values. Then, the value with the highest probability is picked as the next predicted attribute value y_t. The following equations summarize these operations:

$$\begin{aligned} \mathbf{p}_t &= \text{softmax}(\mathbf{s}_t^L(W^P)') \\ y_t &= \text{argmax}(\mathbf{p}_t) \end{aligned} \tag{6}$$

where W^P is the weight matrix of the projection layer.

4. Data and Experimental Setup

In this section, we first describe the three datasets—Rakuten, Kaggle, and Amazon—used for the experiments. Then we will describe the baseline models and detail the experimental setup that was used to train the classification models.

4.1. Data

All three datasets are multimodal, including both image and textual data and are labeled with attribute values. For each dataset, we fetch 5000 samples for validation and test sets at random and use the rest for training. The statistics of the datasets are presented in Table 1.

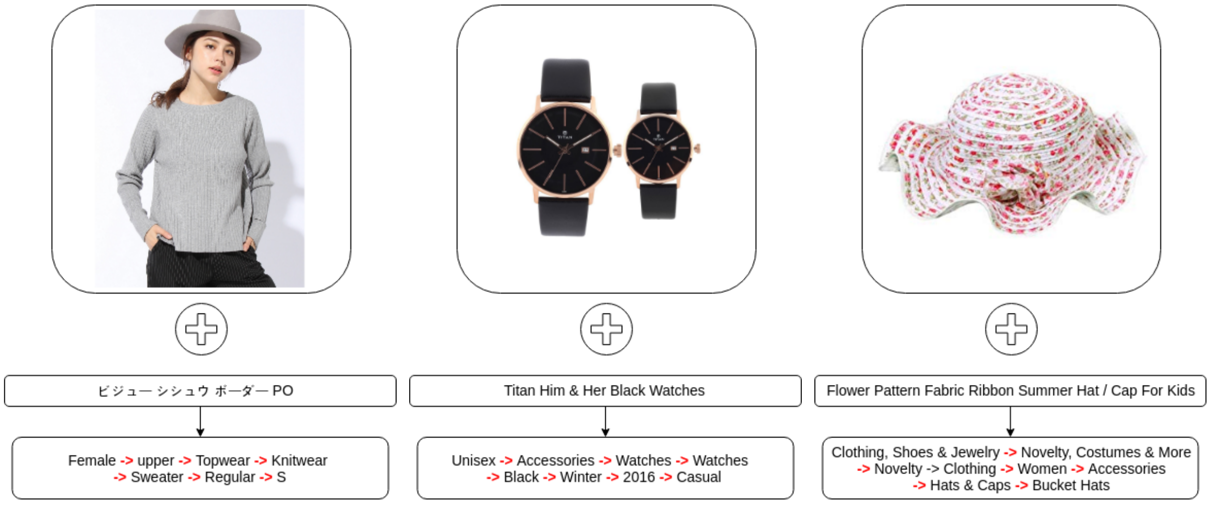
Table 1. Statistics of the experimental datasets.

	Train	**Dev**	**Test**
Rakuten	206,104	5000	5000
Kaggle	34,446	5000	5000
Amazon	1,492,475	5000	5000

4.1.1. Rakuten Dataset

The Rakuten data used in this paper is a confidential dataset prepared by Rakuten Fits.Me (https://fits.me/) annotators. In this dataset, each garment item is manually annotated with 7 hierarchical attributes for detailed classification. Each sample in the Rakuten dataset (Figure 4a) comprises the image and the title of the garment paired with a chain of attribute values. Each value in the category chain is an attribute which belongs to one attribute class or category, and is in a hierarchical relationship with previous and next attributes in the chain. The attribute chain involves the values for the following categories:

(1) GENDER: Male, Female;
(2) CATEGORY: Upper, Trousers, Coverall;
(3) TYPE: Outerwear, Topwear, Skirts, Trousers, Dresses, Jumpsuits;
(4) SUB-TYPE: Knitwear, Vest, Cardigan, Coat, Top, Suit, Underwear, 1st layer, Jacket, Twinset, Trousers, Dress, Skirt, Playsuit, Sportswear, Jumpsuit;
(5) CUT: 78 different values such as for instance Blouse, Sweatpants, Shirt, Anorak, Rugby, etc.;
(6) FIT: Extra Slim, Slim, Regular, Relaxed, Oversized, Extra Oversized, Over-oversized;
(7) STRETCH-FACTOR: Large, Normal, Stretchy, Very Stretchy.


ビジュー シシュウ ボーダー PO
Female -> upper -> Topwear -> Knitwear -> Sweater -> Regular -> S
Titan Him & Her Black Watches
Unisex -> Accessories -> Watches -> Watches -> Black -> Winter -> 2016 -> Casual
Flower Pattern Fabric Ribbon Summer Hat / Cap For Kids
Clothing, Shoes & Jewelry -> Novelty, Costumes & More -> Novelty -> Clothing -> Women -> Accessories -> Hats & Caps -> Bucket Hats

(**a**) Rakuten (**b**) Kaggle (**c**) Amazon

Figure 4. The examples of fashion product items for each of the three datasets.

4.1.2. Kaggle Dataset

This dataset was obtained from the Kaggle platform (https://www.kaggle.com/paramaggarwal/fashion-product-images-dataset). In the dataset, each item sample involves images, 8 different category values, and the product title (Figure 4b). The attribute categories are:

(1) GENDER: Men, Women, Unisex, Boys, Girls;
(2) MASTER CATEGORY: Apparel, Accessories, Footwear, Personal Care, Free Items, Sporting Goods, Home;
(3) SUB-CATEGORY: 45 different values such as for instance Topwear, Shoes, Bags, Bottomwear, Watches, Innerwear, Eyewear, Jewelry, Fragrance, Sandals, etc.;
(4) ARTICLE TYPE: 141 different values such as for instance T-shirts, Shirts, Casual Shoes, Watches, Sports Shoes, Kurtas, Tops, Handbags, Heels, Sunglasses, Wallets, Flip Flops, etc.;
(5) BASE COLOR: 46 different colors or None;
(6) SEASON: Winter, Spring, Summer, Fall, or None;
(7) YEAR: years from 2008 to 2019, or None;
(8) USAGE: Casual, Smart Casual, Sports, Ethnic, Formal, Party, Travel, Home, and None.

Amazon Dataset

We obtained this data from public datasets of amazon web services (https://registry.opendata.aws/). This dataset has been used for several natural language processing and information retrieval tasks [37,38]. It comprises Amazon product metadata including product image URL, title, textual description, category information, price, brand, image, and recommendation links (also viewed / also bought info). It contains 24 main categories, such as BOOKS, CLOTHING, SHOES AND JEWELRY, SPORTS AND OUTDOORS etc. We only use for our experiments the items belonging to the CLOTHING, SHOES AND JEWELRY category, since it this category shares the same domain with the Rakuten and Kaggle datasets. As the textual description is available only for a small part of the whole dataset, we use only product titles as textual input. The attribute chain to be predicted is a list of sub-page names, where each sub-page indicates a sub-category where the item belongs to. For instance, consider the following two example attribute value chains:

(a) *Clothing, Shoes & Jewelry* $\rightarrow$ *Shoes & Accessories: International Shipping Available*
(b) *Clothing, Shoes & Jewelry* $\rightarrow$ *Women* $\rightarrow$ *Shoes* $\rightarrow$ *Sandals*

In the example (a) the first part *Clothing, Shoes & Jewelry* is the main page indicating the main category while *Shoes & Accessories: International Shipping Available* is the sub-page indicating sub-category. In the second example (b), there are three sub-categories *Women*, *Shoes*, and *Sandals*, following each other hierarchically. Since the structure of the category chain is dynamic, unlike in Rakuten and Kaggle datasets, and since the attribute category structure is not specified in the dataset description, we do not know which sub-attributes belong to which category class. For training our models, we use images downloaded from provided URLs, titles and category chains (see Figure 4c for an example).

4.2. Data Preprocessing

The titles of the Kaggle and Amazon datasets are in English, whereas the titles in the Rakuten dataset are in Japanese but may also contain words in Latin characters. To handle this mixture of languages, we process the titles of all datasets with the sentence-piece tokenizer [39]. In the sequential model, we feed the predicted attributes to the model in the textual format as well but here we do not apply the sentence-piecing but treat each attribute value as a single token.

The images are preprocessed to convert them to the format compatible with the Resnet image encoder. The images are cropped between 0.08 and 1 of the original size at random, and a random aspect ratio between 0.75 and 1.33 of the original aspect ratio is kept. Finally, the crop is resized to 224 $\times$ 224 dimensions. The resized image is normalized over 3 channels. For data augmentation, the normalized image is flipped at random with 0.5 probability. The image preprocessing procedures are integrated into the training pipeline.

While preprocessing the Kaggle and Rakuten datasets is straightforward, there is an additional preprocessing step necessary for the Amazon dataset. Each sample in the Amazon dataset involves one image, one title, and might involve more than one category chains. For example, the item with the title "14K Yellow Gold Cameo Stud Earrings Polished Jewelry" there are two different category chains:

(a) *Clothing, Shoes & Jewelry* $\rightarrow$ *Women* $\rightarrow$ *Jewelry* $\rightarrow$ *Fine* $\rightarrow$ *Earrings* $\rightarrow$ *Stud*
(b) *Clothing, Shoes & Jewelry* $\rightarrow$ *Novelty, Costumes & More Novelty*

Thus, we extend our training and validation sets by creating separate data items for each chain. For instance, we create two samples from the above item with both samples sharing the same image and title, but each of them with a different category chain as label.

4.3. Baseline Models

As baselines, we adopt several models that perform product attribute prediction using either multiclass or multilabel formulation. All baseline models incorporate image and text as input, encoded as described in Section 3.1. The outputs of both encoders are average-pooled to obtain fixed-length final feature vectors for both image and text and then concatenated to form the input to the model.

4.3.1. MultiClass Baseline

The first baseline is the standard multiclass classification model that predicts the whole attribute chain as a single label. The finite set of possible attribute chains is constructed during training from the training set and while making new predictions, the model is constrained to choose the attribute chains only from this fixed finite set.

The problem with the multiclass setting in our case is that the label set of attribute chains is not finite. Although not all attribute combinations make sense, there is no guarantee that all valid and sensible attribute chains are present in the training set.

4.3.2. MultiLabel Baselines

Multilabel prediction solves the problem of fixed label set related to the multiclass formulation. We experiment with several different multilabel baselines with increasing complexity.

Binary MultiLabel Model

The first multilabel baseline is the binary multilabel classifier that predicts the presence or absence of each attribute value independently. The problem with this formulations is that although each category (e.g., CATEGORY) can have only one value (e.g., Upper, Trousers, or Coverall), the Binary MultiLabel model can predict multiple values for a single attribute category, as it makes the decision for the presence or absence of each attribute value and does not take into account the values predicted for other attribute values.

For each attribute class, we create a binary vector with the size of the number of labels in the class. In this vector, the index of the annotated label is 1, and the rest is 0. We concatenate all attribute-specific vectors and use a label vector for training. In this way, during training, we learn the binary probability of each attribute value for each sample. During the inference, we de-concatenate the vector of binary probabilities of attributes back into attribute-specific vectors and choose for each attribute class the value with the highest probability.

MultiTask Model

The second multilabel model resembles multitask prediction models used in previous work, i.e., the prediction of each attribute category is treated as a separate task solved with a multiclass classifier. The difference between the Binary MultiLabel and MultiTask models is that while the former makes binary predictions for each attribute value, the latter makes multiclass predictions over attribute categories thus ensuring that each category will be predicted only a single value.

Attentive MultiTask Model

The Attentive MultiTask model [40] extends the regular MultiTask model with a set of label-specific attention module, one for each attribute category. Each of those modules computes an attentional context vector between the input text and image, and its respective attribute category, scanning the input for words or image segments relevant for predicting the value for that attribute category. As attention function the same multi-head attention described in Section 3.2.4 is used:

$$\mathbf{c}_{IT} = \text{MultiHead}(\mathbf{a}_i, \text{concat}(I, T), \text{concat}(I, T)) \tag{7}$$

where the query $\mathbf{a}_i \in \mathbb{R}^{1024}$ is the category-specific representation vector that is obtained by applying a category-specific linear transformation layer to the input representation vector. Both the keys K and values V are concat(I, T), which is the concatenation of the image and text representation matrices. The attention uses $h = 8$ heads. The context vectors obtained from all heads are concatenated with the representation vector $\mathbf{a}_i$, and then fed through the linear output layer of the category classifier.

Self-Attentive MultiTask Model

Finally, the Self-Attentive MultiTask model extends the Attentive MultiTask model by also computing self-attention between attribute categories. The attention function is again the multi-head attention:

$$\mathbf{c}_{AIT} = \text{MultiHead}(\mathbf{a}_i, \text{concat}(A, I, T), \text{concat}(A, I, T)), \tag{8}$$

where $\mathbf{a}_i$ is again the category-specific representation vector. However, here the keys and values are extended with the matrix A that consists of the representation vectors $\mathbf{a}_i$ for all categories $i = 1 \ldots n$.

4.4. Training

We trained our models on P100 GPUs. Our minibatch size is 32, and each training step took about 1 s. On Rakuten and Kaggle datasets, we applied early stopping by monitoring the validation accuracy. The training was stopped when the validation accuracy did not improve within three epochs. It took about 30 min to train an epoch on these datasets. The Amazon dataset is much larger compared to the other two and it took around 22 h to train an epoch. Therefore, we did not use early stopping there and trained each model for 7 epochs, choosing the model obtained after the last 7th epoch.

We use categorical cross-entropy as loss function that minimizes the negative log-likelihood of the attribute chains. We employ the same optimizer, learning rate scheduler, and the same regularization methods as used by Vaswani et al. [36]. The optimizer is Adam [41], with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. Learning rate scheduler is performed according to the following formula:

$$\text{lrate} = d^{-0.5} \times \min(\text{step}^{-0.5}, \text{step} \times \text{warmup\_steps}^{-1.5})$$

where d is the embedding size, step is the current iteration number, and warmup_steps $= 4000$. Attribute embedding size and encoder and decoder hidden layer sizes are 1024. We use eight heads for the multi-head attention for both image and text.

While we update all the layers for text encoder in all models and decoders in sequential models, we update only the last dense layer in Resnet-50 architecture, which is pretrained by ImageNet.

5. Results and Discussion

Table 2 presents the full-chain accuracy for all models on each dataset on both validation and test set. An attribute chain is counted as correct if all attribute values in the chain are predicted correctly. On Rakuten and Kaggle dataset, the sequential model performs the best as predicted, although on the Rakuten dataset, the difference with the MultiClass model is not large. Within multilabel models, the Binary MultiLabel model performance is the worst on the Rakuten Dataset and the basic MultiTask model is the worst on the Kaggle dataset. On both datasets, the multilabel models using attention are the best. However, we can see that the MultiTask model does not benefit from self-attention, as the Self-Attentive MultiTask model performs worse than the Attentive MultiTask model on both datasets.

Table 2. Full-chain accuracy of all models on all datasets.

	Rakuten		**Kaggle**		**Amazon**	
	Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**
MultiClass	26.82	25.98	52.08	51.54	**82.82**	**82.98**
Binary MultiLabel	19.98	20.84	55.00	54.70	-	-
MultiTask	22.34	22.66	53.58	52.40	-	-
Attentive MultiTask	26.28	26.36	56.10	54.70	-	-
Self-Attentive MultiTask	25.06	25.14	55.42	54.60	-	-
Sequence	**27.34**	**27.62**	**68.86**	**67.70**	80.36	80.32

On the Amazon dataset, we could not train the multilabel models as we do not know the category structure. Also, on the Amazon dataset, the MultiClass model performs better on both validation and test sets than the sequential model. There are about 4K unique attribute chains in the Amazon dataset. Because the Amazon training set is very large containing around 4M items, it might be easier for the model to learn to classify items into 4K classes, rather than learn to generate the exact combinations of attributes. Overall, the differences between the validation and test set results are very small, suggesting that all our models generalize well.

Tables 3 and 4 show the prediction accuracies on the attribute level for the Rakuten and Kaggle test sets, respectively. We could not perform the same analysis for the Amazon dataset as we do not know the category structure of that dataset.

Table 3. Attribute level accuracies on the Rakuten test set for all models. Binary ML stands for Binary MultiLabel model, Attn. MT stands for Attentive MultiTask model, Self-Attn. MT denotes the Self-Attentive MultiTask model.

Attribute	**# Values**	**MultiClass**	**Binary ML**	**MultiTask**	**Attn. MT**	**Self-Attn. MT**	**Sequence**
GENDER	2	94.40	94.08	94.46	**96.86**	96.70	96.26
CATEGORY	3	98.58	98.76	98.90	**99.20**	98.66	98.24
TYPE	6	94.20	94.52	94.30	**94.96**	94.92	93.74
SUB-TYPE	16	87.68	88.20	87.62	89.36	**89.42**	88.02
CUT	78	60.62	57.90	58.10	62.06	61.94	**62.58**
FIT	7	49.72	48.06	49.90	51.94	51.80	**52.50**
STRETCH-FACTOR	4	79.38	78.60	77.20	79.64	**80.12**	78.80
AVERAGE	-	80.65	80.02	80.07	**82.00**	81.94	81.45

Table 4. Attribute level accuracies on the Kaggle test set for all models. Binary ML stands for Binary MultiLabel model, Attn. MT stands for Attentive MultiTask model, Self-Attn. MT denotes the Self-Attentive MultiTask model.

Attribute	**# Values**	**MultiClass**	**Binary ML**	**MultiTask**	**Attn. MT**	**Self-Attn. MT**	**Sequence**
GENDER	5	96.56	99.12	99.12	98.94	**99.26**	99.18
MASTER CATEGORY	7	95.96	97.84	97.24	98.76	98.36	**99.48**
SUB-CATEGORY	45	91.64	94.60	94.00	96.10	96.30	**98.70**
ARTICLE TYPE	141	84.62	88.72	87.80	91.26	90.96	**96.50**
BASE COLOR	47	79.10	83.76	82.76	85.86	85.02	**90.80**
SEASON	5	79.92	82.44	81.00	81.42	80.84	**85.06**
YEAR	14	81.32	83.60	83.18	83.54	84.22	**86.64**
USAGE	10	92.10	93.18	92.50	92.90	92.82	**94.06**
AVERAGE	-	87.65	90.41	89.70	91.10	90.97	**93.80**

On the Rakuten dataset (in Table 3), the Attentional MultiTask model has the highest accuracy on three attributes out of seven, while both the Self-Attentional and the Sequential model achieve the highest accuracy on two attribute categories. Although based on average accuracy, the Attentional MultiTask model has the best performance, the difference with Self-Attentional MultiTask model and the Sequential Model is not large. We also note that the Sequential model obtains the highest accuracy on the attribute categories (CUT and FIT) that in general have the lowest accuracy, suggesting that these are the attributes that are hardest to predict.

On the Kaggle dataset, the Sequential model performs the best on seven attribute categories, and it also achieves competitive accuracy on the remaining GENDER attribute. One can observe the largest differences between the Sequential and the other models on those attributes that have more values and that are thus more difficult to predict. For example, the BASE COLOR attribute category has 47 different color attribute values. Many color attribute values are very similar to each other, like for instance *Blue* and *Navy Blue*, or *Pink*, *Rose* and *Red*, or *Grey* and *Grey Melange*. If the text specifically does not mention the name of the color, it is non-trivial to extract the exact tone and intensity of the main color from product images which might involve several color combinations. The MultiClass model achieves the lowest accuracy on average on the Kaggle dataset. While all multilabel models achieve the average accuracy in similar range, the Attentive MultiTask is again the best.

When looking at the average attribute accuracy of the four different multilabel models in Tables 3 and 4, we see that the Attentive MultiTask model performs the best. This is the same pattern as observed in Table 2 for the full-chain accuracy. Therefore, we choose the Attentive MultiTask model as the best baseline multilabel model and omit the other multilabel models in the following analyses.

5.1. Predicting Unseen Attribute Chains

While technically the MultiClass model cannot predict the chains that it has not seen during training, other models can. In the Rakuten test set, there were only three items with novel attribute chains in the test set and these were predicted incorrectly by all models. In the Amazon test set, there were no items with unseen attribute chains. However, in the Kaggle test set, there were 453 items with novel attribute chains. We present results that show the accuracy of the Kaggle test set for both known and new attribute chains in Table 5. The total column is copied from Table 2.

The accuracy of the MultiClass model on the unseen chains is 0.00 as expected. The Attentive MultiTask model can correctly generate novel attribute chains, but the Sequential model does it better. The overall higher performance of the Sequential model stems from higher accuracy on both seen and unseen attribute chains.

Table 5. The accuracy of the different models on the Kaggle test set comparing the performance of seen and unseen attribute chains.

Model	**Seen**	**Unseen**	**Total**
MultiClass	56.67	0.00	51.54
Attentive MultiTask	58.04	21.19	54.70
Sequential	**71.04**	**34.22**	**67.70**

5.2. Contributions of Image and Text

Next, we will look at how image and text contribute to the attribute prediction accuracy. For that purpose, we train all models with only image or only text as input. The results of these ablation experiments are shown in Table 6.

Table 6. Full-chain attribute accuracy on the test sets of all datasets comparing only text, image-only and multimodal models.

	Rakuten			**Kaggle**			**Amazon**		
	Text	**Image**	**Both**	**Text**	**Image**	**Both**	**Text**	**Image**	**Both**
MultiClass	23.62	13.72	**25.98**	**61.24**	20.62	51.54	**83.10**	45.08	82.98
Attentive MultiTask	21.94	19.04	**26.36**	**63.00**	29.97	54.70	-	-	-
Sequential	23.24	19.52	**27.62**	67.48	35.20	**67.70**	80.02	47.20	**80.32**

The general trend visible from Table 6 is that the models using only images perform much worse than the models using either text or both text and image. For the Rakuten data, all models with multimodal input perform better than the models using only text. On the Amazon dataset, the multimodal and text models perform on the same level. However, on the Kaggle dataset, except for the Sequential model, all multimodal models perform significantly worse compared to the text-only models. With the Sequential model, both text-only and multimodal model settings perform on the same level on the Kaggle dataset with the multimodal model being slightly better.

To shed more light on these observations on the Kaggle dataset, we present in Table 7 the attribute accuracies of all models using text and multimodal inputs. The numbers in **Diff** columns show the difference between the performance of the respective multimodal and the text-only models. Inspecting these differences reveals that the largest gaps between the text-only and multimodal models are due to three attribute categories: SUB-CATEGORY, ARTICLE TYPE and BASE COLOR. These were the categories with the largest number of attribute values and thus are the most challenging for the models to predict correctly.

However, in the case of the Sequential model, the differences between the text-only and multimodal model are the smallest for all attribute categories.

One possible reason for such differences between the MultiClass and the Attentive MultiTask models compared to the Sequential model might be due to the different handling input representations by these models. While all baseline models fuse the text and image representations via simple concatenation, the Sequential model uses image and text representations through attention. It is possible that integrating more complex fusion mechanisms into the non-sequential models would help to diminish the gap between the text-only and multimodal settings for these models as well on the Kaggle dataset.

Overall, in two datasets out of three, the multimodal models perform as good or better than text-only or image-only models and the Sequential model either performs better or at least does not degrade the performance.

Table 7. Attribute accuracies on the Kaggle dataset for all models using both text and multimodal inputs. The **Diff** column shows the difference between the multimodal and text-only model.

	MultiClass			**Attentive MultiTask**			**Sequential**		
	Text	**Multi-Modal**	**Diff**	**Text**	**Multi-Modal**	**Diff**	**Text**	**Multi-Modal**	**Diff**
GENDER	98.22	96.56	−1.66	99.44	98.94	−0.5	99.50	99.18	−0.32
MASTER CATEGORY	98.14	95.96	−2.18	99.66	98.76	−0.9	99.46	99.48	+0.02
SUB-CATEGORY	96.94	91.64	−5.30	98.78	96.10	−2.68	98.78	98.70	−0.08
ARTICLE TYPE	94.38	84.62	−9.76	96.64	91.26	−5.38	97.14	96.50	−0.64
BASE COLOR	87.10	79.10	−8.00	88.92	85.86	−3.06	90.40	90.80	+0.40
SEASON	81.58	79.92	−1.66	84.06	81.42	−2.64	84.98	85.06	+0.08
YEAR	83.10	81.32	−1.78	85.92	83.54	−2.38	86.58	86.64	+0.06
USAGE	93.18	92.10	−1.08	94.46	92.90	−1.56	95.00	94.96	−0.04
AVERAGE	91.58	87.65	−3.93	93.48	91.10	−2.38	93.98	93.91	−0.07

5.3. Qualitative Analysis

Next, we will show some qualitative examples of correctly and incorrectly predicted attribute chains. Figure 5 shows a Rakuten test item together with the predictions from three different multimodal models. The original attribute combination of the sample does not exist in the training set. As the Figure shows, none of the models made a correct prediction. However, when inspecting the image and the predicted chains, the predictions of the MultiClass and Sequential models make sense. In the annotation, the TYPE attribute is labeled as *Trousers*. The MultiClass and Sequential models predicted it as *Jumpsuits*. According to the image, it can be easily considered a jumpsuit instead of trousers. By looking at the text, no strong indication about the *Trousers* attribute can be found. The predictions of the Attentive MultiTask model, on the other hand, are inconsistent as hierarchically, the value *Jumpsuit* of the category SUB-TYPE cannot follow the value *Trousers* of the category TYPE.

Figure 6 shows an example from the Rakuten test set, where the title does not give any reliable information, and the image consists of more than one garment. Although the title contains information that might refer to any top or bottom dress, the image features the bottom garment more than the top. Figure 6 shows the prediction generated by the Sequential model. The generated attribute chain is sequentially consistent and the attributes make sense with respect to the image representation.

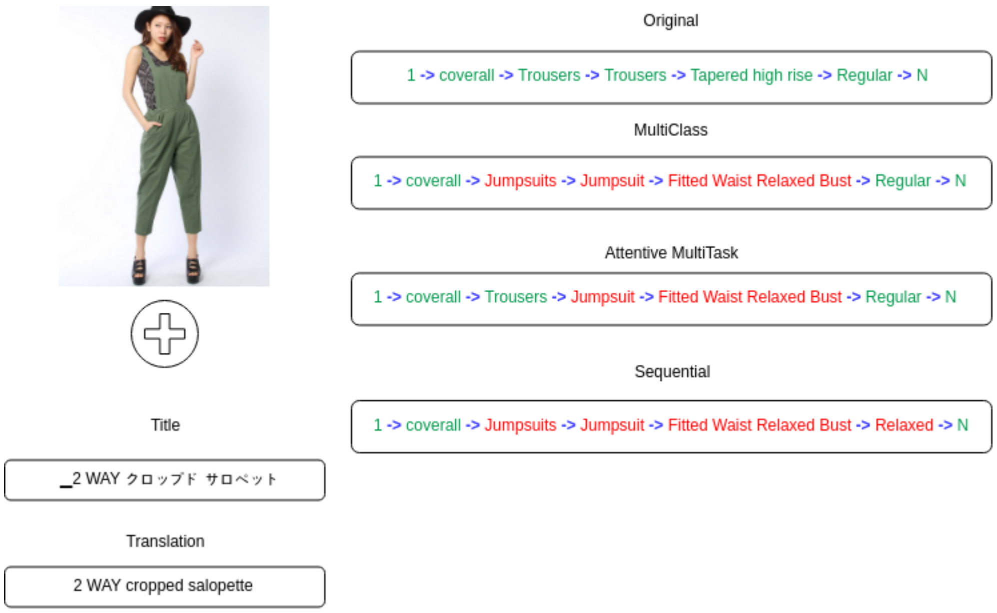

Original
1 -> coverall -> Trousers -> Trousers -> Tapered high rise -> Regular -> N
MultiClass
1 -> coverall -> Jumpsuits -> Jumpsuit -> Fitted Waist Relaxed Bust -> Regular -> N
Attentive MultiTask
1 -> coverall -> Trousers -> Jumpsuit -> Fitted Waist Relaxed Bust -> Regular -> N
Sequential
1 -> coverall -> Jumpsuits -> Jumpsuit -> Fitted Waist Relaxed Bust -> Relaxed -> N
Title
_2 WAY クロップド サロペット
Translation
2 WAY cropped salopette

Figure 5. A sample item from the Rakuten test set with predicted attribute chains. Correctly predicted attributes are in green, the wrongly predicted attributes are in red.

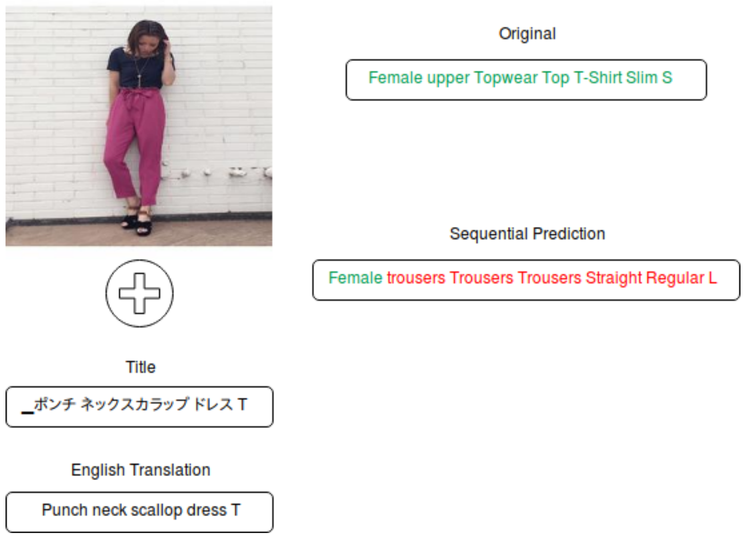

Original
Female upper Topwear Top T-Shirt Slim S
Sequential Prediction
Female trousers Trousers Trousers Straight Regular L
Title
_ポンチ ネックスカラップ ドレス T
English Translation
Punch neck scallop dress T

Figure 6. A sample item from the Rakuten test set, where the original attribute chain is annotated for top dress, but the model focuses on bottom dress attributes. Correctly predicted attributes are in green, the wrongly predicted attributes are in red.

Finally, Figure 7 shows the predictions of all three models for a Kaggle test instance. The prediction of the Sequential model is fully correct while the Attentive MultiTask model makes errors in predicting the chain. Although the chain predicted by the MultiClass model is incorrect, it is nevertheless consistent. However, the prediction of the Attentive MultiTask model is inconsistent—the *Handbags* attribute cannot appear under the hierarchy of *Watches*.

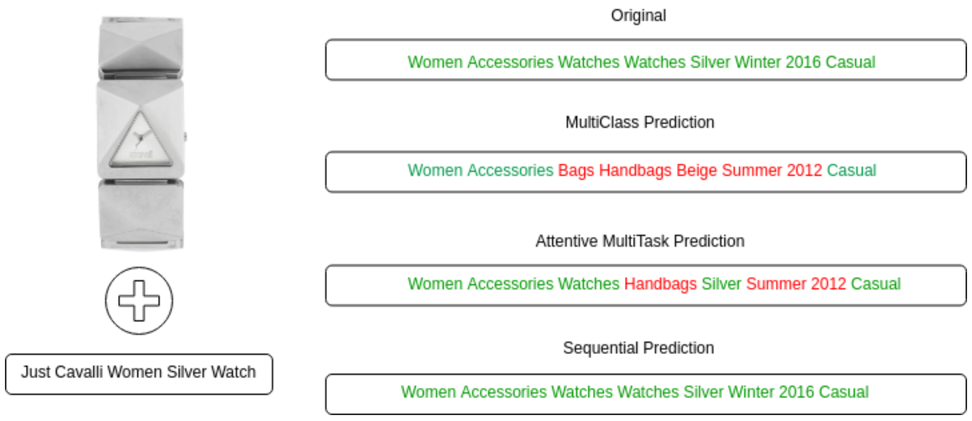

Original
Women Accessories Watches Watches Silver Winter 2016 Casual
MultiClass Prediction
Women Accessories Bags Handbags Beige Summer 2012 Casual
Attentive MultiTask Prediction
Women Accessories Watches Handbags Silver Summer 2012 Casual
Sequential Prediction
Women Accessories Watches Watches Silver Winter 2016 Casual
Just Cavalli Women Silver Watch

Figure 7. A sample item from the Kaggle test set with predicted attribute chains. Correctly predicted attributes are in green, the wrongly predicted attributes are in red.

6. Conclusions and Future work

In this work, we have proposed to use sequential modeling for multimodal fashion product attribute prediction. We demonstrated that the sequential model based on the RNMT+ decoder outperforms all the other baselines on three experimental datasets. Moreover, we showed that although both multilabel and sequential models can generate novel attribute chains, the sequential model does it more accurately. Finally, we also compared multimodal models to text-only and image-only models. While text-only models always outperform image-only models, only the sequential multimodal model demonstrated as good or better performance than the text-only model on all datasets.

We have used RNN and CNN architectures to encode the text and images. Additionally, we have used attention layers for the model to learn relations between the predicted attribute values and the specific areas of text and image. However, these attention layers lack the self-relation information between different parts of the image and text, and between the predicted attribute values themselves. We believe that this information of self-relation can help to better predict the attribute values for noisy samples, where the self-relations can encode information important for inference. Self-attention has been successful for encoding such information in images [42] and text [43]. Although according to our experimental results, the Self-Attentive MultiTask model did not perform as well as the Attentive MultiTask model, we believe that Self-Attentive model can be better formulated. Moreover, we only experimented with the self-attention in the multilabel setting, but it can be beneficial in the sequential model as well. Based on this motivation, in the future, we are planning to use self-attention for a similar performing model architecture for the same task.

Author Contributions: Methodology, H.S.A., K.S., M.F. and G.A.; Supervision, K.S., M.F. and G.A.; Validation, H.S.A.; Writing—original draft, H.S.A.; Writing—review&editing, K.S., M.F. and G.A.

Funding: This work has been partially supported by an R&D Contract between University of Tartu and Rakuten Fits.Me (LLTTI18221), the Estonian Centre of Excellence in IT (EXCITE) funded by the European Regional Development Fund and the Scientific and Technological Research Council of Turkey (TÜBITAK) (116E097).

Acknowledgments: We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research and University of Tartu High Performance Computing Center (HPC). We also thank the anonymous reviewers for their helpful comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Reed, W.B.; Ritchie, C.C.; Akleman, E. Garment Modeling Simulation System and Process. U.S. Patent 10311508, 4 June 2019.
2. Saxena, K.; Shibata, T. Garment Recognition and Grasping Point Detection for Clothing Assistance Task using Deep Learning. In Proceedings of the 2019 IEEE/SICE International Symposium on System Integration, Paris, France, 14–16 January 2019; pp. 632–637. [CrossRef]
3. Yang, S.; Pan, Z.; Amert, T.; Wang, K.; Yu, L.; Berg, T.; Lin, M.C. Physics-inspired garment recovery from a single-view image. *ACM Trans. Graphics* **2018**, *37*, 170.10.1145/3026479. [CrossRef]
4. Wen, J.J.; Wong, W.K. Fundamentals of common computer vision techniques for fashion textile modeling, recognition, and retrieval. In *Applications of Computer Vision in Fashion and Textiles*; Woodhead Publishing: Duxford, UK, 2017; pp. 17–44. [CrossRef]
5. Hao, L.; Hao, M. Design of intelligent clothing selection system based on neural network. In Proceedings of the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC, Chengdu, China, 15–17 March 2019; pp. 1789–1792. [CrossRef]
6. Takatera, M.; Yoshida, R.; Peiffer, J.; Yamazaki, M.; Yashima, K.; Kim, K.O.; Miyatake, K. Fabric retrieval system for apparel e-commerce considering Kansei information. *Int. J. Cloth. Sci. Technol.* **2019**. [CrossRef]
7. Xiang, J.; Zhang, N.; Pan, R.; Gao, W. Fabric Image Retrieval System Using Hierarchical Search Based on Deep Convolutional Neural Network. *IEEE Access* **2019**, *7*, 35405–35417. [CrossRef]
8. Corbiere, C.; Ben-Younes, H.; Rame, A.; Ollion, C. Leveraging Weakly Annotated Data for Fashion Image Retrieval and Label Prediction. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017, Venice, Italy, 22–29 October 2017; pp. 2268–2274. [CrossRef]
9. Cardoso, A.; Daolio, F.; Vargas, S. Product characterisation towards personalisation: Learning attributes from unstructured data to recommend fashion products. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, London, UK, 19–23 August 2018; pp. 80–89. [CrossRef]
10. Logan, R.L.; Humeau, S.; Singh, S. Multimodal Attribute Extraction. *arXiv* **2017**, arXiv:1711.11118.
11. Li, P.; Li, Y.; Jiang, X.; Zhen, X. Two-Stream Multi-Task Network for Fashion Recognition. *arXiv* **2019**, arXiv:1901.10172.
12. Hiramatsu, M.; Wakabayashi, K. Encoder-Decoder neural networks for taxonomy classification. In *CEUR Workshop Proceedings*; CEUR Workshop Proceedings: Ann Arbor, MI, USA, 2018; Volume 2319.
13. Li, Y.M.; Tan, L.; Kok, S.; Szymanska, E. *Unconstrained Production Categorization with Sequence-to-Sequence Models*; eCOM@ SIGIR: Ann Arbor, MI, USA, 2018.
14. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
15. Chen, M.X.; Firat, O.; Bapna, A.; Johnson, M.; Macherey, W.; Foster, G.; Jones, L.; Parmar, N.; Shazeer, N.; Vaswani, A.; et al. The best of both worlds: Combining recent advances in neural machine translation. In Proceedings of the ACL 2018—56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; Volume 1, pp. 76–86.
16. Barrault, L.; Bougares, F.; Specia, L.; Lala, C.; Elliott, D.; Frank, S. *Findings of the Third Shared Task on Multimodal Machine Translation*; Shared Task Papers: Belgium, Brussels, 2019; pp. 304–323. [CrossRef]
17. Druzhkov, P.N.; Kustikova, V.D. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognit. Image Anal.* **2016**, *26*, 9–15. [CrossRef]
18. Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; Alsaadi, F.E. A survey of deep neural network architectures and their applications. *Neurocomputing* **2017**, *234*, 11–26. [CrossRef]
19. Altınel, B.; Ganiz, M.C. Semantic text classification: A survey of past and recent advances. *Inf. Process. Manag.* **2018**, *54*, 1129–1153. [CrossRef]

20. Zahavy, T.; Krishnan, A.; Magnani, A.; Mannor, S. Is a picture worth a thousand words? A deep multi-modal architecture for product classification in e-commerce. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, AAAI, New Orleans, LA, USA, 2–7 February 2018; pp. 7873–7880.
21. Chen, Z.; Ai, S.; Jia, C. Structure-aware deep learning for product image classification. *ACM Trans. Multimed. Comput. Commun. Appl.* **2019**, *15*, 4. [CrossRef]
22. Schindler, A.; Lidy, T.; Karner, S.; Hecker, M. Fashion and apparel classification using convolutional neural networks. *CEUR Worksh. Proc.* **2017**, *2009*, 24–27.
23. Jia, D.; Wei, D.; Socher, R.; Li-Jia, L.; Kai, L.; Li, F.-F. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE conference on computer vision and pattern recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255. [CrossRef]
24. Liu, Z.; Luo, P.; Qiu, S.; Wang, X.; Tang, X. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1096–1104. [CrossRef]
25. Li, R.; Lu, W.; Liang, H.; Mao, Y.; Wang, X. Multiple features with extreme learning machines for clothing image recognition. *IEEE Access* **2018**, *6*, 36283–36294. [CrossRef]
26. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893. [CrossRef]
27. Lin, Y.C.; Das, P.; Datta, A. *Overview of the SIGIR 2018 eCom Rakuten Data Challenge*; CEUR Workshop Proceedings: Ann Arbor, MI, USA, 2018; Volume 2319.
28. Krishnan, A.; Amarthaluri, A. Large Scale Product Categorization using Structured and Unstructured Attributes. *arXiv* **2019**, arXiv:1903.04254.
29. Zheng, G.; Mukherjee, S.; Dong, X.L.; Li, F. OpenTag: Open aribute value extraction from product profiles. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, London, UK, 19–23 August 2018; pp. 1049–1058. [CrossRef]
30. Hsieh, Y.H.; Wu, S.H.; Chen, L.P.; Yang, P.C. Constructing hierarchical product categories for E-commerce by word embedding and clustering. In Proceedings of the 2017 IEEE International Conference on Information Reuse and Integration, San Diego, CA, USA, 4–6 August 2017; pp. 397–402. [CrossRef]
31. Inoue, N.; Simo-Serra, E.; Yamasaki, T.; Ishikawa, H. Multi-label Fashion Image Classification with Minimal Human Supervision. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops, Venice, Italy, 22–29 October 2017; pp. 2261–2267. [CrossRef]
32. Dong, Q.; Gong, S.; Zhu, X. Multi-Task curriculum transfer deep learning of clothing attributes. In Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017, Santa Rosa, CA, USA, 27–29 March 2017; pp. 520–529. [CrossRef]
33. Chen, Q.; Huang, J.; Feris, R.; Brown, L.M.; Dong, J.; Yan, S. Deep domain adaptation for describing people based on fine-grained clothing attributes. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5315–5324. [CrossRef]
34. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer Normalization. *arXiv* **2016**, arXiv:1607.06450.
35. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
36. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Long Beach, CA, USA, 2017; pp. 5999–6009.
37. He, R.; McAuley, J. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In Proceedings of the 25th International World Wide Web Conferences Steering Committee, Montreal, QC, Canada, 11–15 April 2016; pp. 507–517. [CrossRef]
38. McAuley, J.; Targett, C.; Shi, Q.; Van Den Hengel, A. Image-based recommendations on styles and substitutes. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, 9–13 August 2015; pp. 43–52. [CrossRef]

39. Kudo, T.; Richardson, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *arXiv* **2019**, 66–71, arXiv:1808.06226. [CrossRef]
40. Barbieri, F.; Espinosa-Anke, L.; Camacho-Collados, J.; Schockaert, S.; Saggion, H. Interpretable Emoji Prediction via Label-Wise Attention LSTMs. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 31 October–4 November 2018; pp. 4766–4771. [CrossRef]
41. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
42. Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A. Self-Attention Generative Adversarial Networks. *arXiv* **2018**, arXiv:1805.08318.
43. Vaswani, A.; Bengio, S.; Brevdo, E.; Chollet, F.; Gomez, A.N.; Gouws, S.; Jones, L.; Kaiser, Ł.; Kalchbrenner, N.; Parmar, N.; et al. Tensor2Tensor for Neural Machine Translation. *arXiv* **2018**, arXiv:1803.07416.