Large Scale Generative Multimodal Attribute Extraction for E-commerce Attributes

Anant Khandelwal [*]
Ads Trust
Amazon
anantkha@amazon.com

Happy Mittal
International Machine Learning
Amazon
mithappy@amazon.com

Shreyas Sunil Kulkarni
International Machine Learning
Amazon
kulkshre@amazon.com

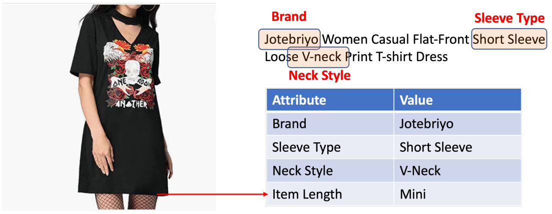
Deepak Gupta
International Machine Learning
Amazon
dgupt@amazon.com

Abstract

E-commerce websites (e.g. Amazon) have a plethora of structured and unstructured information (text and images) present on the product pages. Sellers often either don't label or mislabel values of the attributes (e.g. color, size etc.) for their products. Automatically identifying these attribute values from an eCommerce product page that contains both text and images is a challenging task, especially when the attribute value is not explicitly mentioned in the catalog. In this paper, we present a scalable solution for this problem where we pose attribute extraction problem as a question-answering task, which we solve using **MXT**, consisting of three key components: (i) **M**AG (Multimodal Adaptation Gate), (ii) **X**ception network, and (iii) **T**5 encoder-decoder. Our system consists of a generative model that *generates* attribute-values for a given product by using both textual and visual characteristics (e.g. images) of the product. We show that our system is capable of handling zero-shot attribute prediction (when attribute value is not seen in training data) and value-absent prediction (when attribute value is not mentioned in the text) which are missing in traditional classification-based and NER-based models respectively. We have trained our models using distant supervision, removing dependency on human labeling, thus making them practical for real-world applications. With this framework, we are able to train a single model for 1000s of (product-type, attribute) pairs, thus reducing the overhead of training and maintaining separate models. Extensive experiments on two real world datasets show that our framework improves the absolute recall@90P by 10.16% and 6.9% from the existing state of the art models. In a popular e-commerce store, we have deployed our models for 1000s of (product-type, attribute) pairs.

1 Introduction

E-commerce websites (e.g. Amazon, Alibaba) have a very wide catalog of products. Seller provided catalog of these products contain both textual information and product images. Apart from this unstructured information, they also provide structured information about the products such as color, material, size, etc. This information can be represented in terms of attribute-value pairs (see figure 1). In this paper, we will use the terms attribute and attribute-name interchangeably. The value of attribute will be referred as *attribute-value*. However, while listing the products, sellers rarely specify all attribute values or mistakenly fill incorrect values. These attribute values may or may not be present in the unstructured textual product information. Extracting/inferring the missing attribute values from the unstructured textual product information (and images) can improve the catalog quality, thereby improving the customer experience (again, refer figure 1 for an example of attribute extraction).


Brand
Sleeve Type
Jotebriyo Women Casual Flat-Front Short Sleeve
Loose V-neck Print T-shirt Dress
Neck Style

Attribute	Value
Brand	Jotebriyo
Sleeve Type	Short Sleeve
Neck Style	V-Neck
Item Length	Mini

Figure 1: Illustration of attribute extraction problem

PT-attribute: A PT-attribute is defined as a pair of (product-type, attribute), where product-type (or PT) is a broad category of products (e.g. "shoes",

[*] This work was done while author was in International Machine Learning team.

"dress", "laptops" etc.) and attribute is an attribute-name (e.g. "color", "size" etc.). Typically, attribute-extraction is done at the granularity of PT-attribute (e.g. "extract the value of *color* attribute of *shoe*").

A good attribute extraction system has following desirable properties: (1) **Scalability:** A single model should handle multiple PT-attributes so that there is no need to train a separate model for every PT-attribute combination, (2) **Multi-modality:** Model should be able to extract attributes from multiple modalities like text, image, video etc., (3) **Zero-shot inference:** Model should be able to extract attribute values that were not seen in the training data, and (4) **Value-absent inference:** Model should extract attribute values that are not explicitly mentioned in the text on the product page (but can be inferred from image or some other reasoning).

Related Work: Extensive research has been done to build attribute extraction models, which can be categorized as *extractive*, *predictive*, or *generative*. Extractive models pose this problem as a Named Entity Recognition (NER) problem (Zheng et al., 2018). Some of the recent work in this space include LATEX-numeric (Mehta et al., 2021), and MQMRC (Shrimal et al., 2022b) . However, these models don't do value-absent inference. Moreover, these are text based models and do not use product images. Predictive models are the classifier models that take text (and image) as input and predict the attribute values. CMA-CLIP (Liu et al., 2021) is a recent multi-modal predictive framework for predicting attribute values. However, these models can't do zero-shot inference as the prediction comes from the predefined classes only. Generative models pose this problem as an answer generation task given a question and context. Here, the question is the attribute name, and context is the product data (text and image), and the answer is the attribute value. For example, Roy et. al. (Roy et al., 2021) presented a generative framework to generate attribute values using product's text data. PAM (Lin et al., 2021) introduced a multi-modal generative framework, however their model requires (i) Training encoder and decoder from scratch, (ii) Manually modifying the vocabulary of outputs (attribute-values) for different product-types.

In this paper, we present **MXT**, a multimodal generative framework to solve the attribute extraction problem, that consists of three key components: (i) **M**AG (Multimodal Adaptation Gate) (Rahman et al., 2020b): a fusion framework to combine textual and visual embeddings, that enables generating image-aware textual embeddings, (ii) **X**ception network (Chollet, 2017): an image encoder that generates attribute-aware visual embeddings, and (iii) **T**5 encoder-decoder (Raffel et al., 2020). The models trained by our generative framework are scalable as a single model is trained on multiple PT-attributes, thus reducing the overhead of training and maintaining separate models. We remove the disadvantages of PAM model by (i) finetuning a strong pre-trained language model (T5 (Raffel et al., 2020)) and thus leveraging its text generation ability, (ii) providing product-type in the input itself so that output distribution is automatically conditioned on the PT. Moreover, our trained model satisfies all of the 4 desirable properties that were mentioned previously.

Our system formulates the attribute extraction problem as a question-answering problem, where (a) question is the attribute name (e.g. "color"), (b) textual context comprises of a concatenation of product-type (e.g. "shirt"), and textual description of the product, (c) visual context comprises product image, and (d) answer is the attribute value for the attribute specified in the question. Our model architecture consists of (i) a T5 encoder to encode the question and textual context, (ii) encoding visual context into product specific embeddings through a pre-trained ResNet-152 model (He et al., 2016) and fusing them with T5's textual embeddings using a multimodal adaptation gate (MAG) (Rahman et al., 2020a), (iii) encoding visual context into attribute (e.g. "sleeves", "collar" etc.) specific embeddings through Xception model (Chollet, 2017) and fusing them with previously fused embeddings through a dot product attention layer (Yu et al., 2021), and finally (iv) generating the attribute values through T5 decoder. The detailed architecture of our system is shown in figure 2.

In section 2, we explain our proposed model MXT. In section 3, we compare our model's performance with NER-Based MQMRC (Shrimal et al., 2022a) along with a popular multi-modal model CMA-CLIP (Liu et al., 2021) and show that on same precision, we outperform them (on recall) for a majority of the attributes. We also show an ablation study justifying the proposal of different components in MXT. Finally, we also show that our model is able to perform zero-shot and value-absent inference. Our trained models using MXT framework are being used to extract attributes for over 12000 PT-attributes in a popular e-commerce store, and have extracted more than 150MM attribute values.

2 MXT Framework

Given a set of product-types (PTs) $\mathcal{P} = \{p_1, p_2, \ldots, p_m\}$ and attribute-names $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$, we define MXT$_{\mathcal{P},\mathcal{A}}$ as a multi-PT, multi-attribute, and multi-modal generative model that is trained on PT-attributes from $(\mathcal{P}, \mathcal{A})$, and can be used to generate attribute value for any product in the trained PT-attribute set. The overall architecture of our model is described in figure 2.

2.1 Problem Formulation

We formulate the problem of attribute extraction as the problem of answer generation given a question and a context. Here question is the attribute-name $a \in \mathcal{A}$, and context consists of textual description, product type $p \in \mathcal{P}$ and image of the product. All of these are used to extract attribute values. The answer generated from the model is the attribute value for a. As shown in figure 2, our model architecture mainly consists of 3 components: (a) Image-aware Text encoder, (b) Attribute-aware Text-Image Fusion, and (c) Text decoder. Below, we describe each component in detail.

2.2 Image-aware Text encoder

We use T5 (Raffel et al., 2020), which is a transformer (Vaswani et al., 2017) based text only Seq2Seq pretrained language model. It includes a bidirectional encoder and a unidirectional (forward only) decoder. In this section, we give an overview of T5's encoder and details of its usage for our task. Our text input consists of (i) attribute-name (e.g. "color"), (ii) product-type (e.g. "dress"), and (iii) textual description of product. In our QnA format, the question consists of attribute-name, and context consists of concatenation of product-type and textual description of the product. We tokenize both question and context and create a single input sequence of tokens. This input sequence x is then fed to an embedding and positional encoding layer to create input features $T_{emb} \in \mathbb{R}^{N \times d}$, where N is the sequence length and d is the feature dimension. These input text embeddings are then fused with Multimodal Adaptation Gate (MAG) as described in Rehman et. al. (Rahman et al., 2020b) to generate image aware text embeddings. Due to MAG, the internal representation of words (at any transformer layer) is shifted conditioned on visual modalities. This attachment essentially puts words into a different semantic space, which is conditioned on the visual inputs. For e.g., the meaning of the word "ripple" changes according to the visual input soap image or paper image. With soap, the meaning is "free and clear", while with paper, the meaning is "wavy pattern" as shown in figure 3. This module shifts the meaning of "ripple" according to visual modality. Since T5 is pretrained model and can understand only text embeddings it is required to fuse the visual embeddings ($V_R \in \mathbb{R}^d$) with text before feeding it to T5 Encoder rather than feeding the visual embeddings along with text. Specifically, in MAG, for each input token i of the sequence, we first learn a gating vector g_i using concatenated embeddings of T^i_{emb} and V_R: $g_i = RELU(W_g[T^i_{emb}; V_R] + b_g)$. This gating vector highlights the relevant information in visual modality conditioned on the input textual vector. We then create an image displacement vector H_i by multiplying V_R with each token's gating vector g_i: $H_i = g_i \cdot (W_H V_R) + b_H$. Finally, we shift the embedding T^i_{emb} by the weighted displacement vector H_i to get the multimodal vector $\hat{T}^i_{emb} = T^i_{emb} + \alpha * H_i$. In this equation, $\alpha = min(\frac{||T^i_{emb}||_2}{||H_i||_2} * \beta, 1)$, where β is a hyper-parameter whose value is taken as it is from the paper (Rahman et al., 2020b). This is then passed through a layer normalization followed by a dropout layer to get the final fused embedding F_{MAG} from MAG module, where $F^i_{MAG} = \text{dropout}(\text{LN}(\hat{T}^i_{emb}))$. This fused output is then fed to the T5 encoder. The encoder consists of L encoder-layers. It takes F_{MAG} as input gives T_{enc} as output. Equation 1 shows the encoding done by k^{th} layer. Here SA is the multi-head self attention layer, Res is the residual connection, LN is the layer normalization, and FC is a fully connected layer.

$$T^k_{enc} = \text{LN}(\text{Res}(\text{FC}(\text{LN}(\text{Res}(\text{SA}(T^{k-1}_{enc})))))) \tag{1}$$

2.3 Attribute-aware Text-Image Fusion

Xception(Chollet, 2017) model performs depth-wise (or channel-wise) separable convolutions, i.e., it applies separate filters for different color channels. We propose another fusion layer based on the Xception network. The advantage of using this is that it can readily learn the visual features conditioned on the attribute type. For example, for the attribute "sleeve type" of a dress, it can iden-

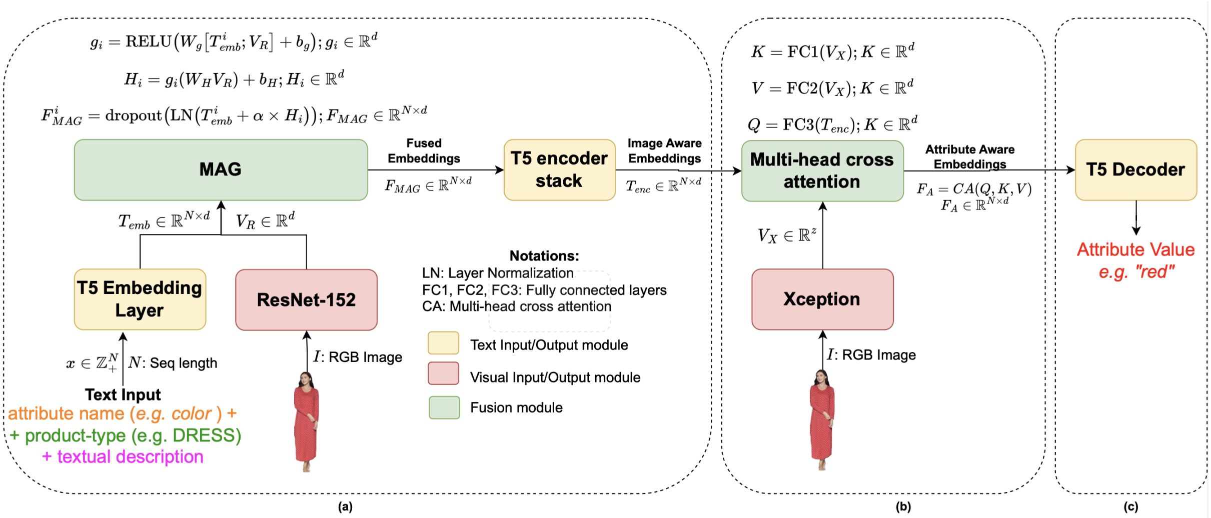

$g_i = \text{RELU}(W_g[T^i_{emb}; V_R] + b_g); g_i \in \mathbb{R}^d$
$H_i = g_i(W_H V_R) + b_H; H_i \in \mathbb{R}^d$
$F^i_{MAG} = \text{dropout}(\text{LN}(T^i_{emb} + \alpha \times H_i)); F_{MAG} \in \mathbb{R}^{N \times d}$
MAG
Fused Embeddings
$F_{MAG} \in \mathbb{R}^{N \times d}$
T5 encoder stack
Image Aware Embeddings
$T_{enc} \in \mathbb{R}^{N \times d}$
$T_{emb} \in \mathbb{R}^{N \times d}$
$V_R \in \mathbb{R}^d$
T5 Embedding Layer
ResNet-152
$x \in \mathbb{Z}^N_+$
N: Seq length
Text Input
attribute name (e.g. color) +
+ product-type (e.g. DRESS)
+ textual description
I: RGB Image
Notations:
LN: Layer Normalization
FC1, FC2, FC3: Fully connected layers
CA: Multi-head cross attention
Text Input/Output module
Visual Input/Output module
Fusion module
$K = \text{FC1}(V_X); K \in \mathbb{R}^d$
$V = \text{FC2}(V_X); K \in \mathbb{R}^d$
$Q = \text{FC3}(T_{enc}); K \in \mathbb{R}^d$
Multi-head cross attention
Attribute Aware Embeddings
$F_A = CA(Q, K, V)$
$F_A \in \mathbb{R}^{N \times d}$
$V_X \in \mathbb{R}^z$
Xception
I: RGB Image
T5 Decoder
Attribute Value
e.g. "red"
(a)
(b)
(c)

Figure 2: Architecture of MXT. (a) Generates image-aware text embeddings by fusing image embeddings (obtained from ResNet-152) and text embeddings of the input text (concatenation of *attribute name*, *product type*, and textual description of the product), (b) Image-aware text embeddings are then attended with region specific visual embeddings obtained from separable convolution of Xception Network, which in turn passes only the attribute specific embeddings to the decoder (c) Fused embeddings are passed through T5 decoder to generate attribute value.

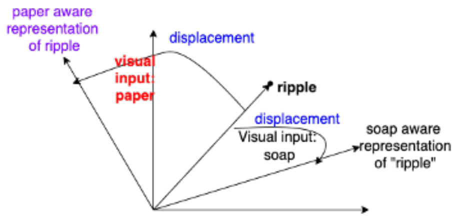

paper aware representation of ripple
displacement
visual input: paper
ripple
displacement
Visual input: soap
soap aware representation of "ripple"

Figure 3: Shift in text embeddings (e.g. "ripple") after applying MAG with visual embeddings

tify the channel/color difference between sleeves of dress and skin of the person, thus identifying whether sleeve is half or full. We then fuse the text and image embeddings using multi-head cross attention. As shown in figure 2(b), a product image has several regions of interest, for different attributes like "neck style" and "sleeve type". This region specific embeddings are learnt by separable convolutions in Xception which is then attended with text embeddings to arrive at attribute aware text embeddings. Now given text embedding $T_{enc} \in \mathbb{R}^{N \times d}$ and image embedding $V_X \in \mathbb{R}^{1 \times x}$ (from MXT), we create an attribute-aware fused embedding $F_A \in \mathbb{R}^{N \times d}$ (having same dimension as of text embedding). This fused embedding is created through a multi-head cross attention module, that applies cross attention between textual and visual embeddings as shown in figure 2. This fusion has an advantage that for an attribute, different attention scores can be learned for each object of an image, allowing attending to specific portions of the product image conditioned on the attribute name in the question. For example, for the product type "shirt" and attribute "sleeve-type", we may want to concentrate only on the portions of the image where sleeves are visible.

2.4 Text Decoder

We use T5's unidirectional decoder to output the attribute values. The input to the decoder is the fused embedding vector $F_A = < F^1_A, F^2_A, \ldots, F^N_A >$. The decoder iteratively attends to previously generated tokens $y_{<j}$ (via self-attention) and F_A (via cross-attention), then predicts the probability of future text tokens $P_\theta(y_j|y_{<j}, x, I) = \text{Dec}(y_{<j}, F_A)$. For attribute generation, we finetune our model parameters θ by minimizing the negative log-likelihood of label text y tokens given input text x and image I: $L^{GEN}_\theta = -\sum_{j=1}^{|y|} log P_\theta(y_j|y_{<j}, x, I)$.

3 Experimental Setup & Results

30PT Dataset: We picked 30 product types (PTs) consisting of total 38 unique attributes from a popular e-commerce store. For each product in the dataset, we have textual information and image. The dataset has 569k and 84k products in train and validation data across 30 PTs. Our test data consists of products from two product types with a total of 73k products.

PT	#top attributes	CMA-CLIP	MXT			
			Multi-PT	Single PT	Without-Xception	Without-MAG
A	K=5	+6.16%	**+22.33%**	+19.58%	+21.82%	+20.93%
	K=10	+6.70%	**+16.89%**	+15.50%	+15.60%	+15.19%
	K=15	+1.81%	**+13.23%**	+11.64%	+10.67%	+10.45%
B	K=5	+8.34%	**+16.63%**	+12.86%	+13.94%	+13.58%
	K=10	+18.46%	**+24.98%**	+22.46%	+22.81%	+22.55%
	K=15	+11.72%	**+18.51%**	+15.50%	+16.28%	+15.68%

PT	MXT
A	**+15.56%**
B	-1.47%
C	-7.89%
D	**+9.98%**
E	**+13.23%**

Table 1: Left: Improvement in Recall@90P% of CMA-CLIP and MXT (with different ablation studies) over NER-MQMRC on 30PT datasetE-commerce5PT dataset. Right: Improvement in F1-score of MXT over NER-MQMRC on E-commerce5PT dataset

We evaluated MXT against two state of the art methods on attribute extraction: (1) **CMA-CLIP:** A multi-task classifier that uses CLIP (Radford et al., 2021) for learning multi-modal embeddings of products followed by using two types of cross-modality attentions: (a) sequence-wise attention to capture relation between individual text tokens and image features, and (b) modality-wise attention to capture weightage of text and image features relative to each downstream task, (2) **NER-MQMRC:** This framework (Shrimal et al., 2022b) poses Named Entity Recognition (NER) problem as Multi Question Machine Reading Comprehension (MQMRC) task. This is the state of the art model for the text-based attribute extraction task. In this model, given the text description of a product (*context*), they give attribute names as *multiple questions* to their BERT based MRC architecture, which finds span of each attribute value *answer* from the context.

Left table in the figure 1 compares the recall@90P% of the three models. We show the performance on top-5, top-10 and top-15 attributes (by number of products in which they are present. We can see that MXT outperforms MQMRC and CMA-CLIP on both product types.

E-commerce5PT: This is a benchmark dataset from NER-MQMRC paper (Shrimal et al., 2022b). We take a subset of this dataset (removing numerical attributes) consisting of 22 product-attributes across 5 product types. This is a benchmark dataset for NER based models since all attribute values are present in the text in this dataset. The dataset has 273,345 and 4,259 products in train and test data respectively. We compare average F1 scores (averaged across attributes for each product type) of MXT model with NER-MQMRC on this dataset where our model outperforms NER-MQMRC on 16/22 attributes. Right table in the figure 1 shows the average F1-scores (across attributes in each product type) of MXT and NER-MQMRC models.

3.1 Ablation Study

We show three ablation studies on 30PT dataset that justify our choices in the MXT architecture. Left table in the figure 1 shows the results of these studies. **(a) Scalability:** We show that our proposed framework is highly scalable. For that, we compute Recall@90P% of the MXT model trained on individual PTs. The results show that (i) our model leverages cross-PT information during training, (ii) we don't need to train separate model for each PT, which makes model monitoring and refreshing easier in the production, **(b) Xception network:** We show that Xception network helps concentrating on certain attribute features. For this, we removed the Xception network from our architecture and trained and evaluated the model, **(c) MAG:** We replaced MAG with simple concatenation of text and image embeddings in MXT. We can see in the table that each of our ablation model under-performs the MXT model trained on 30PTs, thus justifying our design choices.

3.2 Zero-shot Inference and Value-absent Inference

Most existing methods for attribute extraction face two challenges: **(i) Zero-shot inference:** All the predictive models (classification-based models) can predict attribute values only from a predefined set of values that are seen in the training data. They are unable to do zero-shot inference i.e. they can't predict an attribute value if it is not seen in the training data, **(ii) Value-absent inference:** All NER-based models can extract values only which are mentioned in the text data i.e. if an attribute value is absent in the input text, they can't extract that value. Our generative model solves both of these challenges. For example, in the E-commerce5PT dataset, there are a total of 8289 product-attribute pairs in the test data, out of which 970 product-attribute pairs were not seen in the training data, from which our model correctly generated 124 product-attribute pairs. For example, given a product of product-type *"dress"* with title *"Tahari ASL Women's Sleeveless Ruched Neck Dress with Hi Lo Skirt"*, our model generated the value *"Ruched Neck"* for the attribute *"neck style"*. Here the value *"Ruched Neck"* was absent from the training data. Similarly, for the *"dress"* product shown in figure 1 , our model generated the value *"mini"* for the attribute *"item length"* (by inferring it from the image) even when this value is not mentioned in the product text(thus solving the second challenge).

3.3 Training & Inference Details

We conducted training for each model over a span of 20 epochs, employing a batch size of 4. The training process was performed using distributed multi-GPU training across 8 V-100 Nvidia GPUs, each equipped with 16GB of memory. For text encoder and decoder, we finetune the pretrained `t5-base` [1] checkpoint. We obtained ResNet-based image embeddings using a pretrained ResNet-152, specifically with one embedding assigned to each image. [2]. During training, we employed the Adam optimizer with learning rate of $5e^{-5}$ and warmup ratio of 0.1. We chose the checkpoint having best validation loss. For inference, we used greedy search to generate attribute values.

4 Deployment

In a popular e-commerce store, we have deployed MXT for 6 English speaking markets covering >10K PT-attributes and have extracted >150MM attribute values.

Design Choices: In popular e-commerce stores, usually there are more than 100K PT-attributes across various markets. Earlier models like NER-MQMRC or CMA-CLIP could be trained only for few 100s of PT-attributes. NER-MQMRC (Shrimal et al., 2022a) architecture only allowed one product type in one model training, while CMA-CLIP couldn't scale beyond few 100s of PT-attribute pairs due to network explosion (as they had to create an output layer for each of the different attribute value). This had serious issues of monitoring, refreshing and maintaining the quality of models. Our prompt-based approach in MXT allows us to train a single model checkpoint for any number of PT-attribute pairs.

Practical Challenges: We faced several challenges during building and deploying the model. One of the biggest challenge was lack of normalized attribute values. Since we were relying on the distantly supervised training data from the catalog, there were multiple junk values. Normalizing these values is challenging without the support of annotations. To overcome this problem, we used some heuristic matches to merge similar values. We also trimmed the tail attribute values to remove the junk values further. The second major challenge was to evaluate the model and find the threshold for every PAC to achieve the desired precision. Since we had >10K PT-attributes, even if we annotate 300 samples per PT-attribute, it leads to 3MM annotations, which is not feasible. For that, we evaluated the model automatically using the catalog data. Since the catalog data can be noisy, we checked other things like whether the predicted value is present in text, whether the attribute should allow zero-shot prediction etc. Based on these checks, we decided the required precision accordingly.

5 Conclusion & Future Work

In this paper, we presented MXT, a large scale multi-modal product attribute generation system to extract product attributes from the products listed in eCommerce stores. Our model infers the attribute values using both textual and visual information present on the product pages. We introduced a novel architecture comprising a T5 based encoder and decoder along with two fusion layers to fuse text and image embeddings. We showed our model can beat the existing state of the art extractive as well as predictive models on the benchmark datasets. Our model is scalable to multiple product types and countries by just specifying them in the input text prompt. We further showed that our model is able to perform zero-shot inference, as well as it can generate attribute values not present in the text. There are several future directions to ex-

[1] The `t5-base` checkpoint is available at https://huggingface.co/transformers/model_doc/t5.html

[2] https://download.pytorch.org/models/resnet152-b121ed2d.pth

plore which can further improve the performance of our model. First, we would like to create an ensemble of NER-based and generative models so that we can leverage the power of extraction based models which work very well for numerical attributes (e.g. size, length etc.). Second, our current approach does not use relational information among the products. Since similar products can have common attribute values, we can use graph based approaches to capture that relational information. Specifically, we can approach the attribute extraction problem through either link prediction or node classification. In the former method, we aim to predict missing links between products and their attributes. Alternatively, the latter approach involves using similarity between product features, including text, images, and co-viewing information, to determine graph edges for classification of product nodes.

6 Limitations

In this section, we discuss some of the limitations of our current model architecture: (1) **Non-English locales:** Currently in our experiments, we have trained and evaluated models only on English datasets. Building models on non-English locales is the direction for future work, (2) **Use of pre-trained tokenizer:** The T5's tokenizer in our models has been pre-trained on open-domain datasets, and its vocabulary misses out on e-commerce specific terms. For example, the current tokenizer of T5 tokenizes the phrase "skater midi dress" as ["sk", "a", "ter", "mid", "I", "dress"]. Here, the meaning of words "skater" and "midi" is not captured in the tokenized text. We believe that we can overcome this limitation by pre-training T5 on e-commerce data which would help tokenizer understanding and tokenizing the e-commerce specific terms more correctly.

7 Acknowledgements

We thank all the anonymous reviewers for providing their valuable comments that helped us improve the quality of our paper. We also thank our colleagues in the science, product, and engineering teams at Amazon for their valuable inputs.

References

François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition supplementary materials. In *IEEE conference on computer vision and pattern recognition*, pages 770–778.

Rongmei Lin, Xiang He, Jie Feng, Nasser Zalmout, Yan Liang, Li Xiong, and Xin Dong. 2021. Pam: Understanding product images in cross product category attribute extraction. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Huidong Liu, Shaoyuan Xu, Jinmiao Fu, Yang Liu, Ning Xie, Chien Wang, Bryan Wang, and Yi Sun. 2021. Cma-clip: Cross-modality attention clip for image-text classification. *ArXiv*, abs/2112.03562.

Kartik Mehta, Ioana Oprea, and Nikhil Rasiwasia. 2021. Latex-numeric: Language agnostic text attribute extraction for numeric attributes. In *NAACL*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Wasifur Rahman, M. Hasan, Sangwu Lee, Amir Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. 2020a. Integrating multimodal information in large pretrained transformers. *Proceedings of the conference. Association for Computational Linguistics. Meeting*, 2020:2359–2369.

Wasifur Rahman, Md Kamrul Hasan, Sangwu Lee, AmirAli Bagher Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. 2020b. Integrating multimodal information in large pretrained transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2359–2369, Online. Association for Computational Linguistics.

Kalyani Roy, Pawan Goyal, and Manish Pandey. 2021. Attribute value generation from product title using language models. In *Proceedings of the 4th Workshop on e-Commerce and NLP*, pages 13–17, Online. Association for Computational Linguistics.

Anubhav Shrimal, Avi Jain, Kartik Mehta, and Promod Yenigalla. 2022a. NER-MQMRC: Formulating named entity recognition as multi question machine reading comprehension. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies: Industry Track, pages 230–238, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Anubhav Shrimal, Avi Rajesh Jain, Kartik Mehta, and Promod Yenigalla. 2022b. Ner-mqmrc: Formulating named entity recognition as multi question machine reading compmrehension. *ArXiv*, abs/2205.05904.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Tiezheng Yu, Wenliang Dai, Zihan Liu, and Pascale Fung. 2021. Vision guided generative pre-trained language models for multimodal abstractive summarization. *ArXiv*, abs/2109.02401.

Guineng Zheng, Subhabrata Mukherjee, Xin Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.