TU WIEN Informatics

Comparative Analysis of Fashion Captioning and Multimodal Fashion Recommendation

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Data Science

eingereicht von

Gwendolyn Rippberger, BSc.
Matrikelnummer 01307685

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Assistant Prof. Mag.a rer.nat. Dr.in techn. Julia Neidhardt

Wien, 31. März 2025

Gwendolyn Rippberger	Julia Neidhardt

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Comparative Analysis of Fashion Captioning and Multimodal Fashion Recommendation

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Data Science

by

Gwendolyn Rippberger, BSc.
Registration Number 01307685

to the Faculty of Informatics

at the TU Wien

Advisor: Assistant Prof. Mag.a rer.nat. Dr.in techn. Julia Neidhardt

Vienna, 31st March, 2025

Gwendolyn Rippberger	Julia Neidhardt

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Gwendolyn Rippberger, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 31. März 2025

Gwendolyn Rippberger

Acknowledgements

At this point, I would like to sincerely thank all the people who have accompanied and supported me throughout my master's thesis.

My special thanks go to Julia Neidhardt, who gave me the opportunity to write this thesis and guided me with her expertise and support throughout the entire process. I would also like to thank Thomas Kolb, who was always there to assist me with advice and support—without him, this thesis would not have been possible in its current form.

A big thank you also goes to my mom, who supported me in her own way. Obrigada, mãe, por tudo que você fez por mim durante os meus estudos. Sem você, eu não poderia ter feito isso.

Additionally, I would like to express my heartfelt gratitude to my wonderful boyfriend, Dipl. Ing. Thomas Niedermayer, B.Sc., who has been by my side throughout our shared academic journey. His unwavering encouragement, insightful advice, and constant support have meant the world to me, and I am truly grateful for everything he has done.

Furthermore, I would like to thank Daniele Malitesta and Nicholas Moratelli, who kindly took the time to answer my questions regarding their work. A special thanks also goes to Gábor Recski for his valuable feedback on my evaluation.

Without the support, exchange, and motivation of all these people, this thesis would not have been possible in this form. Thank you very much!

Abstract

This thesis explores two main tasks: (1) fine-tuning image captioning models for fashion datasets and (2) evaluating different feature spaces for personalized fashion recommendations. We fine-tune state-of-the-art vision-language models—BLIP-2 and LLaVA—on two fashion datasets, H&M and FACAD, to generate product descriptions. Our quantitative and qualitative analyses show that fine-tuning can achieve performance levels comparable to fully training a model (SRFC) specifically for generating "fashion captions".

With our qualitative analysis of the captioning results, we take a deep dive into understanding the models' limitations and identify what works well and what does not. We find that working with datasets that have clearly identifiable visual cues for words, e.g., front pocket, can improve the fine-tuning process. The models struggled with non-visual attributes (e.g., material composition, designer names), distinguishing fine-grained differences (e.g., satin vs. velvet), and handling partial or ambiguous product images. These limitations highlight the need for dataset curation that emphasizes visible attributes.

For recommendations, we extract multimodal features (visual, textual, and combined) and evaluate them using the VBPR recommendation algorithm on the H&M dataset. Besides sophisticated models for feature embeddings such as ResNet50 (visual features) or SentenceBERT (textual features), we use our, on the H&M dataset, fine-tuned BLIP-2 model to extract additional features, which we hypothesized to work better. Surprisingly, textual embeddings performed better than visual and multimodal features with VBPR, suggesting that text-based attributes provide better signals for recommendations than image features, in this setup. However, overall performance across different feature spaces remains similar, with ItemKNN outperforming VBPR results.

Our findings demonstrate that fine-tuning is an effective and simpler alternative to complex reward-based training. Additionally, despite fashion being a visual domain, textual descriptions resulted in the best recommendation performance. Future work should focus on exploring the performance of already available models for fashion datasets and refining datasets for better performance.

Contents

CHAPTER 1

Introduction

1.1 Preface

The fashion industry has emerged as a compelling area of research not only because of its economic impact but also because of the diverse range of challenges it confronts. Within the context of online clothing purchases, a multitude of tasks are included, e.g., recommending matching or complementary items, image-based item retrieval, and more.

Recommendations are pivotal in improving the customer's shopping experience by intuitively guiding shoppers towards items aligned with their preferences, fostering a more engaging and personalized shopping journey. Better recommendations can also lead to fewer returns, optimized purchases, better use of resources, and higher revenues. There is a large variety of different algorithms and frameworks using data to enhance the shopping experience for customers in an online setting [SMSC, DNR$^+$, GQLD].

1.2 Motivation and Problem Setting

Navigating an online fashion shopping platform to locate a specific clothing item can be a laborious endeavor, akin to a virtual treasure hunt. Amidst the vast array of offerings, users often find themselves embarking on a time-intensive quest, scrolling through numerous product listings, employing imprecise keyword searches, and toggling filters in pursuit of that one specific item they have in mind. Recommending items based on previous interactions that nearly match their expectations can help refine the options until they arrive at the desired clothing item.

As fashion is a domain focused on the visual appearance of items, several papers proposed an item recommendation based on extracted visual features [CSC$^+$]. Still, there is visual information that is easier expressed through words, e.g. style, cuts, and tailor details. Additionally, well-written descriptions can assist visually impaired individuals in

navigating fashion items while shopping online. However, some websites do not provide descriptions for all clothing items, or sometimes none at all. This gap poses a challenge, specifically in generating textual descriptions for fashion items—a task known as fashion captioning.

The fashion domain has not seen extensive research in image captioning, mainly because of the scarcity of publicly accessible datasets and the constraints posed by existing captioning models. These models are typically designed for natural images and exhibit subpar performance when adapted to the fashion context.

However, in recent years two major datasets were released, one of them being the H&M Personalized Fashion Recommendation dataset [LHR+]. This dataset was provided for a kaggle challenge in February 2022. It includes information about 1,372,980 customers on 105,542 articles with 31,788,324 transactions. Fashion items in this dataset include attributes about their appearance, a detailed description, and an image.

The second dataset is called the fashion captioning dataset (FACAD) created by Yang et al. [Yan] in the year 2020. The dataset originally consisted of 993K high-resolution images (1560x2392) with descriptions having an average length of 21 words. It includes 78 categories, 990 attributes, and multiple images per item. FACAD was specifically created to evaluate the performance of image captioning models on a fashion captioning task [SCB+].

Using those datasets, we propose a methodology that covers the use case of generating a description for a clothing item and then, recommending items using visual and textual features extracted. Big fashion online platforms provide a large collection of product images and descriptions, making it interesting to utilize both visual and textual information to enhance the accuracy and relevance of recommendations.

For the case of item descriptions, we will explore how well those can be generated using pre-trained image captioning models and to what extent fine-tuning can improve results (based on image captioning metrics). Using item descriptions as textual features and corresponding clothing item images as visual features, we want to compare the performance of using different feature spaces for recommendations: visual, textual, and multimodal (visual and textual combined).

1.3 Research Questions

Through our literature review, we identified fashion captioning as an under-researched area. With the rise of large language models (LLMs) with multimodal capabilities, this presents an opportunity to explore the performance of pre-trained image captioning models such as BLIP-2 [LLSH] on domain-specific datasets like FACAD and the H&M dataset. Our goal is to compare the performance of these off-the-shelf models with models specifically trained for fashion captioning, such as the one introduced by Yang et al. [YZJ+]. This comparison is conducted in both, a zero-shot setting and after fine-tuning. Based on this motivation, we define our first research question:

RQ1: To what extent can fine-tuning improve the performance of off-the-shelf image captioning models on domain-specific fashion datasets?

To answer this question, we use two recent fashion datasets: the H&M dataset published in 2022 and the FACAD dataset from 2020 (details in Chapter 3). We focus on open-source models with multimodal capabilities (further details in Section 4.1). The phrase "to what extent" in our research question refers to evaluating the performance of fine-tuned models compared to a task-specific fashion captioning model like the one presented by Yang et al. [YZJ+]. The evaluation is conducted using standard image captioning performance metrics (presented in Section 2.1.3) as well as two additional metrics introduced by Yang et al. (see Section 4.3). Furthermore, we conduct a qualitative analysis of the results (see Section 5.1.3) to provide a deeper understanding of the models' limitations that go beyond quantitative results.

In addition, since each clothing item in the datasets includes both an image and a textual description—and the H&M dataset was originally published for a recommendation challenge—exploring multimodal fashion recommendation is a natural next step. After fine-tuning the models for image captioning, we can leverage their feature embeddings and employ feature extraction models to assess the performance of recommendation systems using visual, textual, and multimodal (visual and textual) embeddings. This leads to our second research question:

RQ2: Which feature embeddings (textual, visual, or multimodal) provide the best recommendations?

For this research question, we use the H&M dataset, as FACAD does not include user-item interactions. As a baseline, we define unpersonalized recommendation approaches, including random recommendations and recommending the most popular items. We use Visual Bayesian Personalized Ranking (VBPR), which allows us to experiment with different feature spaces to answer RQ2. The results are then evaluated using the metrics described in Section 2.2.1 to define the best working setting.

To give a richer context of the results, we additionally run state-of-the-art collaborative filtering algorithms (see Section 4.4.2).

1.4 Methodological Approach

This thesis employs the Design Science Research Framework (DSRF) [HRM+] to develop and evaluate artifacts in models and methods. The previous sections explained the problem of specifically generating descriptive captions for fashion items and recommending items based on different feature spaces. The relevance of the research lies in 1) generating suitable item descriptions for fashion items and 2) improving recommendations using better feature representations.

Following the DSRF approach, the artifacts developed in this thesis are models. We demonstrate research quality by using open-source models and evaluating them with

sophisticated task-specific metrics. The evaluation process involves quantitative and qualitative analysis of the image-captioning models, providing a thorough understanding of their suitability for domain-specific applications.

The methodological approach of this thesis is divided into the following stages:

1. **Literature Review and Tool Exploration**

   The first step involved a thorough review of the existing literature in two areas: fashion image captioning and fashion recommendation systems. The objective was to identify prior work related to domain-specific image captioning, fashion recommendation methods using visual and textual features, and the usage of multimodal data in fashion recommender systems. This stage also included an exploration of existing frameworks, tools, and models. We reviewed various pre-trained image captioning models and evaluated their usefulness for this project. We also explored datasets like the H&M Personalized Fashion Recommendation dataset [LHR+] and FACAD [Yan] to identify appropriate data sources for experiments.

2. **Data Acquisition and Preprocessing**

   After identifying suitable datasets, the next step was to acquire and preprocess the data. The H&M and FACAD datasets were used, including detailed descriptions and images of clothing items. The FACAD dataset was provided in a preprocessed format, including descriptions, processed images, and split datasets. The H&M dataset was provided as an unprocessed dataset; the individual preprocessing steps performed are explained in Section 3.2.

3. **Pre-trained Model Evaluation for Captioning**

   Using the preprocessed data, we first evaluated the performance of existing image captioning models, such as BLIP-2, in a domain-specific context. This step aimed to determine how well general-purpose captioning models perform on fashion-related data and whether these models can effectively generate relevant and descriptive captions for fashion items.

   The evaluation was conducted using standard image captioning metrics such as BLEU, ROUGE, CIDEr, and METEOR (see Section 2.1.3) and additional metrics described in Section 4.3. These metrics provided insight into the accuracy, fluency, and coverage of the generated captions.

4. **Model Fine-tuning for Improved Captioning**

   Based on the initial evaluations, we moved on to fine-tune the pre-trained models for improved performance in the fashion domain. We also compared the performance of the fine-tuned models on the FACAD dataset against the model specifically trained for fashion captioning, presented by Yang et al. [YZJ+].

5. **Feature Extraction and Recommendations**

   For our recommendation task, we extracted feature representations from the fine-tuned models trained on the H&M dataset. These features were then used to generate recommendations, which we evaluated using NDCG and MAP (further detailed in Section 2.2.1).

   As baselines, we included random recommendations and a popularity-based approach that recommends the most frequently purchased item. Additionally, we compared results to state-of-the-art collaborative filtering techniques based on item and user similarity. We employed Visual Bayesian Personalized Ranking (VBPR), a statistically-driven recommendation algorithm that leverages different feature vectors to improve item recommendations.

   The objective of this analysis was two-fold: first, to assess how different feature representations influence the performance of the recommendation algorithm, and second, to contextualize its effectiveness by comparing it to other recommendation approaches. This allowed us to determine which feature space yielded the best recommendation results.

1.5 Structure of the Work

Following the steps previously presented, we provide an overview of the structure of this thesis and its chapters. We begin by presenting the state-of-the-art methods in Chapter 2, covering both image captioning (2.1) and fashion recommendation systems (2.2), including the metrics used for evaluation. In Chapter 3, we introduce the two datasets used and show their differences. This is followed by Chapter 4, where we describe the development of our solution in detail, including the selection of models for image captioning, the fine-tuning process, additional evaluation metrics, and the methods used for recommendation and feature extraction. The main contribution of this thesis is presented in Chapter 5, where we report the results in relation to the research questions introduced in this chapter. This section also includes multiple illustrative examples to help understanding. Finally, in Chapter 6, we conclude with a summary of the work, highlight our contributions, and discuss limitations and directions for future research.

CHAPTER 2

State of the Art

This chapter provides an overview of state-of-the-art methods used for image captioning (2.1.3). We present the general architecture of combining a visual encoder and a language model to generate captions. Both parts can be individually combined and provide a variety of options (see Figure 2.1). Then we present different image captioning datasets (2.1.2) and their attributes. Finally, we explain the details of the metrics used for evaluating the captions (2.1.3).

Finally, we give a comprehensive overview of fashion recommendation algorithms (2.2) that include multimodal aspects as well as the metrics (2.2.1) used in this work for evaluating the recommendations.

2.1 Image Captioning

2.1.1 Methodology

Image captioning is an interdisciplinary field that combines computer vision and natural language processing. It focuses on teaching machines to understand images and generate coherent textual descriptions. As the field is large and fast developing, several review papers were published [GPM, YSR+, SCB+, HSSL, BA, BCE+].

In 2014, Sutskever et al. [SVL] introduced an encoder-decoder model framework, which takes an original input sequence (such as an image, text, or video) and converts it into a fixed-sized vector. Then, the decoder part of the model translates this vector into the desired output sequence. This type of architecture is also known as Sequence-to-Sequence architecture (or short Seq2Seq). Since then the common approach for image captioning has involved using a combination of a visual encoder and a language model for generating textual content (see summary of taxonomy in Figure 2.1).

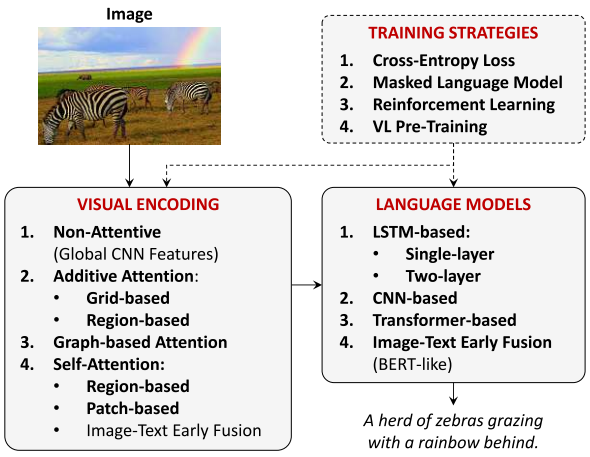


Figure 2.1: Overview of the image captioning task methodology and taxonomy of the most relevant approaches. Source:[SCB+], page 2

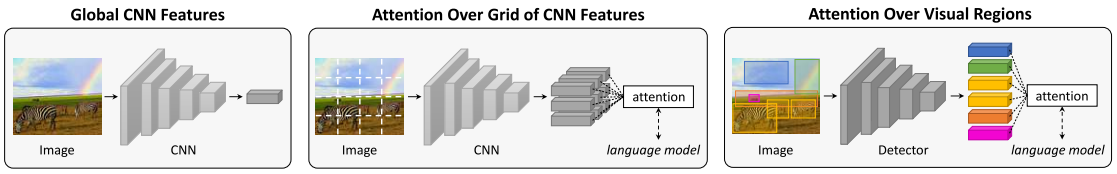
Visual Encoding

Vinyals et al. [VTBE] enhanced the model from Sutskever et al. for image description generation by using a convolutional neural network (CNN) for encoding the images (see Figure 2.2a) and the decoder using a long short-term memory network (LSTM) for text generation.

Because CNNs tend to lead to information loss due to excessive compression and lack of granularity, the image descriptions would be general and capture the overall essence of the image but not the details. To improve the granularity level of visual encoding, Xu et al. [XBK+] tried to imitate the focusing mechanism of the human eye by incorporating an additional spatial attention mechanism across the spatial grid produced by the convolutional layer (see Figure 2.2b). Each word then reflects relevant regions of the image. Similar is the approach presented by Dai et al. [DYL] using a 2D activation map instead of 1D global feature vectors to connect spatial information directly to the language model. The principle of additive attention is to use weights to emphasize important parts of a sequence. Initially designed to model relationships between two sequences, this concept was adapted to connect visual representations with hidden states of a language model.

From a top-down perspective, additive attention enables the language model to generate the next word by focusing on a predefined grid of features, which remains unaffected by the actual content of the image. Region-based attention proposed by Anderson et al. [AHB+] improves this by preselecting regions based on, e.g., proposed regions by an image detector model. Those are then used to create feature vectors for the attention mechanism (see Figure 2.2c).

A different approach is to use graphs to encode image regions and their relationships (see



(a) (b) (c)

Figure 2.2: Different methods of visual encoding: (a) CNN to extract global features, (b) grid-based attention mechanism, and (c) region-based attention mechanism. Source:[SCB+], page 3

Figure 2.3a). Yao et al. [YPLM] first presented the use of a graph convolutional network (GCN) to combine semantic and spatial relationships between objects. Compared to neural networks, graph neural networks can handle non-euclidean data, e.g., graph-based relationships between objects. Using a classifier that was previously trained on Visual Genome [KZG+] the interactions between object pairs are predicted. The spatial relationship is extracted from geometry measures, e.g., i.e. intersection over union, relative distance, and angle between bounding boxes of object pairs.

What is seen as the breakthrough in this area, is the 2017 proposed transformer architecture by Vaswani et al. [VSP+] that further developed the concept of the attentive mechanism using self-attention. Self-attention establishes connections between all elements within a set. Residual connections can be used in this mechanism to repeatedly improve the representation of the same elements, as shown in Figure 2.3b. The transformer architecture provided a foundational element for other breakthroughs in Natural Language Processing (NLP) but has also found significant applications in Computer Vision tasks.

One of the early applications of self-attention was introduced by Yang et al.[YZC], who employed a self-attentive module to capture relationships between features extracted by an object detector. Building upon this idea, Li et al.[LZLY] proposed a model that integrates both region-based features and semantic features obtained from an external tagger. In both approaches, self-attention and feed-forward layers are used to encode the representations.

Newer papers [TCD+, DBK+] propose skipping convolutional layers completely and using image patches directly as input for transformer-like architectures.

Language Models

Given the sequential structure of language, RNNs are a natural choice for sentence generation, with LSTM [HS] being the most widely adopted variant for language modeling. LSTM, compared to the traditional RNNs for sequential data, can better preserve information of long-term sequences tackling the vanishing/exploding gradient problem [Hoc]. In image captioning, the core idea is to replace textual input with a visual encoding of an image, allowing the model to generate descriptive sentences. Given a sequence of

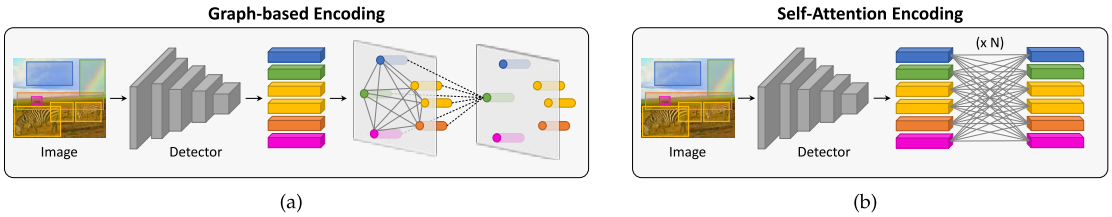


Figure 2.3: Different methods of visual encoding: (a) graph-based, and (b) self-attention-based. Source:[SCB+], page 4

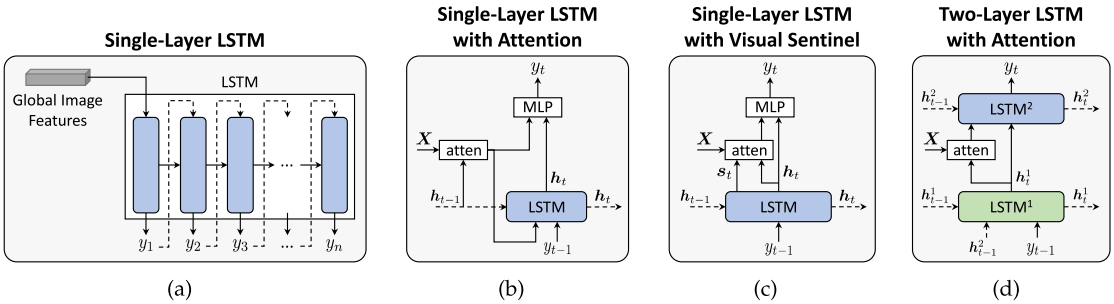


Figure 2.4: Different models based on LSTM architecture: **(a)** Using extracted visual features as a hidden state for one single LSTM model **(b)** LSTM-based model enhanced by adding attention, proposed by Xu et al. [XBK+] **(c)** adding visual sentinel as learnable vector as proposed by Lu et al. [LXPS] **(d)** stacked two-layer LSTM with attention presented by Anderson et al. [AHB+]. In all figures, $\boldsymbol{X}$ represents previously extracted image features by e.g. a CNN or object detector. Source:[SCB+], page 6

n words and a visual representation $\boldsymbol{X}$ of the image, the model assigns a probability $P(y_1, y_2, \ldots y_n \mid \boldsymbol{X})$ to the sequence as:

$$P(y_1, y_2, \ldots y_n \mid \boldsymbol{X}) = \prod_{i=1}^{n} P(y_i \mid y_1, y_2, \ldots, y_{i-1}, \boldsymbol{X})$$

Vinyals et al. [VTBE] introduced one of the simplest LSTM-based decoder architectures, which consists of a single-layer LSTM. Figure 2.4a shows the main idea of using visual features extracted by, e.g., a CNN and providing them as the initial hidden state to the LSTM.

Building on efforts to enhance the visual encoding of images, Xu et al. [XBK+] improved the LSTM-based model by incorporating an attention mechanism. As shown in Figure 2.4b, this mechanism uses the previous hidden state to attend over the visual features $\boldsymbol{X}$, resulting in a context vector. This context vector is then passed to a Multilayer Perceptron (MLP), which is used to predict the next word. The MLP, a fully connected layer followed by a softmax function, outputs a vector with the same dimensionality as the vocabulary, assigning a probability to each possible word.

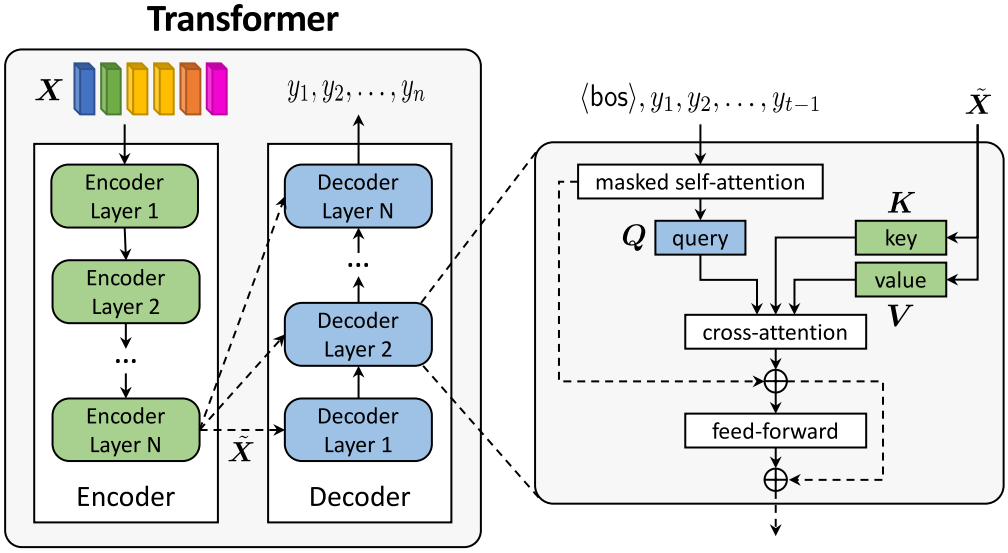


Figure 2.5: Simplified transformer architecture. Source:[SCB+], page 7

Lu et al. [LXPS] enhanced spatial image features by incorporating a supplementary learnable vector, called visual sentinel. This vector can be attended to by the decoder instead of visual features when generating "non-visual" words (such as "the", "of", and "on"), where visual features are unnecessary (see Figure 2.4c). The visual sentinel vector is derived from the previous hidden state and the previously generated word. Combined with the image features, it is used to compute the context vector.

LSTM layers can be stacked to improve the model's ability to capture higher-order relations. Donahue et al. [DHR+] introduced the first two-layer LSTM architecture, in which the hidden states of the first layer serve as input to the second layer. Similar to the single-layer LSTM, this model was later improved by adding visual attention [AHB+].

In Figure 2.4d, the first LSTM layer functions as a top-down visual attention module. It considers the previously generated word, the prior hidden state, and the mean-pooled image features. Using an additive attention mechanism, it computes a probability distribution over the image regions. The resulting attended image feature vector is then passed to the second LSTM layer, where it is combined with the hidden state from the first layer to generate a probability distribution over the vocabulary.

Aneja et al. [ADS] proposed an unconventional approach by feeding a combination of global image feature vectors and word embeddings into a CNN. To prevent the model from accessing information from future word tokens, right-masked convolutions are employed—ensuring that only past and present context is used during prediction. Although this method allows for parallel training, CNN-based language models have not gained widespread adoption, primarily due to their subpar performance and the growing dominance of transformer-based architectures.

As mentioned previously, the transformer-based architecture presented by Vaswani et al.

[VSP+] and its various adaptations (cited 97,372 times since 2017[1]) have revolutionized the field of Natural Language Processing (NLP) being the building block for important developments in NLP, like BERT [DCLT] and GPT [RN]. Transformers are based on an encoder and decoder structure (see Figure 2.5). Each decoder layer applies masked self-attention to the input words and uses cross-attention to capture the relationships between the image features and the words.

Vaswani et al. [VSP+] defined the terms *query*, *key* and *value* analogous to retrieval systems. The query (the sentence) will be mapped against a set of keys (image features) and associated with values (also image features). The similarity between query and key returns weights that are used to weight the corresponding value vectors. This gives the formal definition of attention to the scaled dot-product of a set of n_q query vectors $\boldsymbol{Q}$, a set of key vectors $\boldsymbol{K}$, and a set of value vectors $\boldsymbol{V}$, both containing n_k elements coming together as the following formula

$$\text{Attention}\left(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\right) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V}$$

In this case, d_k is a scaling factor that depends on the dimension of the key vectors. As this is computed individually for each word and the order of words is added via positional encoding this allows transformers to be parallelized.

Other works, leveraging self-attention for encoding visual features, have demonstrated impressive performance, using vision-and-language pre-training [LBPL, TB] and early-fusion strategies [LYL+, ZPZ+]. Vision-and-language pre-training used by, e.g., CLIP [RKH+] are based on learning a text encoder and an image encoder jointly. The goal is to minimize the distance between the embeddings of matching image-text pairs and maximize it for non-matching image-text pairs. On the other hand, early fusion combines both image and text embedding into a multimodal representation. Using self-attention, semantic alignments can be learned, and inter-model correlations can be leveraged.

Large Language Models with Multimodal Capabilities

Based on the concept of Transformers (Figure 2.5 and Section 2.1.1), the term Large Language Models (LLMs) evolved, describing powerful artificial intelligence models, like GPT-3 [BMR+], that utilize attention mechanisms and vast amounts of pre-existing text data to understand and generate human-like language across a wide range of tasks.

Inspired by natural intelligence that is not limited to a single modality, additional modalities were incorporated into LLMs[2]. In the last years, major research labs introduced various LLMs with multimodal capabilities, e.g. DeepMind's Flamingo [ADL+], Salesforce's BLIP [LLXH], Microsoft's KOSMOS-1 [HDW+], Google's PaLM-E [DXS+], and

[1] Google Scholar article, last accessed 20.11.2023, 13:49

[2] `https://huyenchip.com/2023/10/10/multimodal.html`, last accessed 29.03.2025, 11:39

Tencent's Macaw-LLM [TC]. Chatbots such as ChatGPT [BMR+, Ope] and Gemini [SP] are also able to understand and use different types of input, such as text and images.

Multimodal can describe multiple scenarios:

- Input and output are of different modalities (e.g. text-to-image, image-to-text)
- Inputs are multimodal (e.g. a system that can process both text and images)
- Outputs are multimodal (e.g. a system that can generate both text and images)

In the case of image captioning as covered in Section 2.1 the initial framework has images encoded and then decoded into text. Another method can also be giving images, as well as a prompt (an explicit instruction or question), to the model to specify a task, e.g., identifying the style of the dress depicted in this image.

On a high level, a multimodal system comprises the following key elements:

1. An encoder dedicated to each data modality is responsible for producing embeddings specific to the data of that modality.
2. Mechanisms for aligning embeddings from diverse modalities into a unified multimodal embedding space.
3. In the case of generative models, the inclusion of a language model to create text responses (or a vision model to create images). Given that inputs can encompass both textual and visual elements, new techniques are essential to enable the language model to base its responses not only on text but also on visual inputs.

Ideally, many of these components should be pre-trained and reusable to improve efficiency.

Fashion Captioning

Image captioning is a computer vision task that involves generating descriptive textual explanations for the content of an image using machine learning. The task of fashion captioning was first mentioned by Yang et al. [YZJ+]. Although both tasks cover generating image descriptions, fashion captioning is done specifically for fashion items and differs from the "conventional" image captioning problem in the following ways [YZJ+]:

- Fashion captioning focuses on describing fine-grained attributes of a single item, while traditional image captioning typically highlights the entities within an image and their relationships—for example, a person wearing a dress.

Table 2.1: Results of current state-of-the-art models on the FACAD test split. Source: [MBM+], page 10

	B-1	B-4	M	R	C	mAP
Show, Attend and Tell [XBK+]	-	4.3	9.5	19.1	35.2	0.056
Up-Down [AHB+]	-	4.4	9.7	19.6	36.9	0.058
LBPF [QDZL]	-	4.5	9.5	19.1	36.4	0.055
ORT [HKBS]	-	4.2	10.2	19.9	36.7	0.061
SRFC [YZJ+]	-	4.4	9.8	20.2	35.6	0.058
SCST [RMM+]	-	5.6	11.8	22.0	39.7	0.080
SRFC (RL-fine-tuned) [YZJ+]	-	6.8	**13.2**	**24.2**	42.1	0.095
Knowledge Retrieval based [MBM+]	**27.3**	**10.6**	11.5	22.3	**84.5**	**0.248**
Transformer	24.5	6.8	10.1	19.7	53.0	0.238
$\mathcal{M}^2$ Transformer [CSBC]	24.7	6.8	10.4	19.9	53.3	0.237
CaMEL [BSC+]	25.0	7.0	10.7	20.4	55.0	0.241

- For fashion items the descriptions tend to be long due to "fancy" expressions and detailed descriptions. The average caption length in the Fashion Captioning Dataset provided by Yang et al. is 21 compared to the average caption length in MS COCO [LMB+] which is 10.4 words (see Table 2.2).

- Descriptions for fashion items tend to be "more enchanting" to sound more attractive to the customer. Sentences like "so-simply yet so-chic" are preferred over straightforward words like "plain" or "undecorated" used in MS COCO.

For the task of fashion captioning, Yang et al. [YZJ+] proposed two reward functions, one related to the generation of single attributes and one that covers the semantics of the entire sentence, to train an LSTM model for captioning fashion items. Additionally, they created the biggest existing fashion-specific dataset for image captioning tasks, the fashion captioning dataset (FACAD). A more recent paper by Moratelli and Barraco et al. [MBM+] improved those results with a transformer-based captioning model with the integration of external textual memory that can be accessed through k-nearest neighbor (kNN) searches.

Using the performance metrics described in Section 2.1.3 the authors of [YZJ+] and [MBM+] compared their models to the current state-of-the-art models using the test dataset from FACAD. The results can be seen in Table 2.1, where models are grouped by (i) models trained with cross-entropy loss, (ii) models trained with reinforcement learning, (iii) transformer-based models.

2.1.2 Datasets

Generally, the trend and focus of most papers is to improve the general architecture of image captioning models and to tweak training and data preprocessing methods to

improve performance. This is done using popular generic datasets such as MS COCO [LMB+, CFL+] or Flickr [YLHH, HYH]. This is also because otherwise, it is difficult to compare performance between models, and the MS COCO dataset provides one of the largest collections of image and caption pairs (330K images with 5 captions per image, see Table 2.2).

The initial MS COCO dataset contained more than 120,00 images of complex scenes with people, animals, and common everyday objects. For easier comparability, an official split into training (82,783 images), validation (40,504), and test set (40,775) was provided. Nevertheless, most of the literature used a split definition provided by Karpathy et al. [KFF] which proposes using 5,000 of the original validation set for validation and 5,000 for test and the rest for training. Currently[3] the MS COCO dataset contains 330K images and different splits were provided over the years [4].

The Flickr datasets (31K and 8K, respectively called Flickr30K and Flickr8K) were used by the early image captioning architectures [KFF, DHR+] and consist of images collected from the Flickr website.

Only a limited number of works focus on domain-specific image captioning tasks, such as CUB-200 (birds) [WBM+], Oxford-102 (flowers) [NZ], or FACAD (fashion) [YZJ+]. These datasets address both visual challenges—such as variations in image types and styles—and semantic challenges. As discussed in Section 2.1.1, such challenges include the use of domain-specific vocabulary and stylistic expressions. This distinction is clearly visible when examining the 50 most frequently used words in the datasets (see Figure 2.6). While the fashion captioning dataset emphasizes clothing items and descriptive adjectives, MS COCO is centered around general-purpose image descriptions.

Another example of semantic challenges is the BreakingNews [RYMNM] and GoodNews [BGRK] datasets. Those promote the utilization of a more extensive vocabulary which is attributed to the fact that their images, sourced from news articles, are accompanied by detailed captions authored by expert journalists. This can be seen, for example, by the average caption length of BreakingNews (see Table 2.2) being the longest with 28.1 words. Other fashion datasets are mentioned in [YZJ+] but are targeted towards different tasks, e.g., item retrieval, segmentation, or fashion classification. FACAD was specifically created for the fashion captioning task containing 993K high-resolution images with 6 to 7 images per clothing item which also resembles the online shopping environment.

2.1.3 Evaluation Metrics

The following section describes state-of-the-art evaluation metrics for generated text. They are chronologically ordered from the oldest metric, BLEU (2002), to the newest one, SPICE (2016).

[3] https://cocodataset.org/#home, last accessed 20.11.2023, 13:49
[4] https://cocodataset.org/#download, last accessed 13.03.2025, 14:57

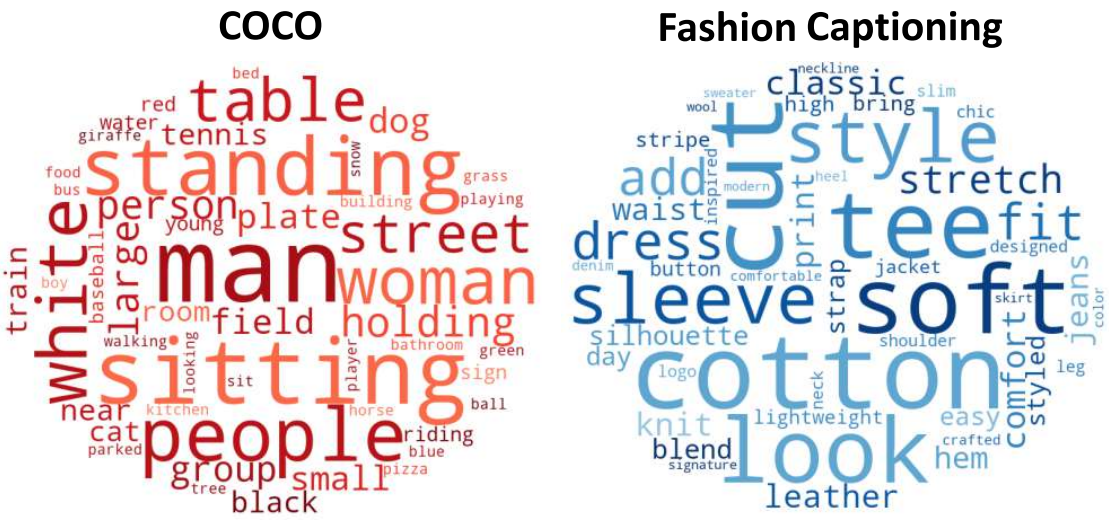


Figure 2.6: Two word clouds representing the 50 most used visual words in the image captions from MS COCO (red) and FACAD (blue). Source:[SCB+], page 10

Table 2.2: Examples of different popular image captioning datasets and domain-specific datasets.

	Domain	Nb. Images	Nb. Caps (per Image)	Vocab Size	Nb. Words (per Cap.)
COCO[LMB+]	Generic	330K	5	27K	10.5
Flickr30K[YLHH]	Generic	31K	5	18K	12.4
Flickr8K[HYH]	Generic	8K	5	8K	10.9
CUB-200[WBM+]	Birds	12K	10	6K	15.2
Oxford-102[NZ]	Flowers	8K	10	5K	14.1
FACAD[YZJ+]	Fashion	130K	1	17K	21.0
BreakingNews[RYMNM]	News	115K	1	85K	28.1
GoodNews[BGRK]	News	466K	1	192K	18.2

BLEU

Bilingual Evaluation Understudy (BLEU)[5] [PRWZ] is a metric to evaluate machine translation systems. It is based on *modified n-gram precision* and *best match length*. N-gram describes n consecutive words in a sentence. Precision is a metric used to measure the number of words that overlap in the candidate and the reference sentence.

To avoid inflating scores due to repeated words, BLEU uses *modified n-gram precision*, which matches each n-gram only once in the candidate sentence. The n-gram precision scores are then combined using their *geometric mean*. This approach accounts for the fact that precision decreases exponentially as n increases. Logarithmic averaging is applied to

[5] https://www1.cs.columbia.edu/nlp/sgd/bleu.pdf, last accessed 29.03.2025, 11:51

represent the scores more fairly:

$$\text{Precision} = \exp\left(\sum_{n=1}^{N} w_n \log p_n\right), \quad \text{where } w_n = 1/n$$

Here, p_n denotes the modified precision for n-grams up to length N, and w_n are positive weights summing to one. This mechanism inherently penalizes sentences longer than their reference by considering each n-gram only once. Additionally, weights can be adjusted to prioritize specific n-gram overlaps if desired.

To prevent candidates from being excessively short, BLEU introduces a *brevity penalty*. This penalty is set to 1.0 when the candidate text matches the reference length, referred to as the *"best match length"*. In the original paper, the reference length r is calculated for a corpus by summing the best match lengths for each candidate sentence. The brevity penalty is then computed as a decaying exponential of r/c, where c is the total length of the candidate translations in the corpus.

$$\text{Brevity Penalty} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

Then both values are combined into one score $BLEU = BP \cdot Precision$, which ranges from 0 to 1. Only sentences that are identical to the reference sentence achieve a score of 1. Generally, we can interpret the BLEU score as how similar the reference sentences and the generated sentences are based on the respective n-grams.

The BLEU score is a widely used metric because it is quick to calculate and easy to understand. Still, it has been criticized for its weaknesses. It does not consider the similar meaning of words or synonyms. When computing the score those words are marked as incorrect. It also does not consider variations of the same word, e.g. "run" and "running". Irrelevant filler words, e.g., "an", "the", etc., are penalized equally to important words that contain meaning. Lastly, the change of order in words that would change the content of the sentence completely could still result in a high score, e.g., "The cat eats the pizza" vs. "The pizza eats the cat".

ROUGE

Recall Oriented Understudy for Gisting Evaluation (ROUGE) [Lina] is based on the *Recall*, *Precision*, and the *F1-Score* of unigrams and n-grams. There are different nuances to the ROUGE metric, e.g., ROUGE-1 (unigrams), ROUGE-2 (bigrams), ROUGE-N (overlap of n-grams), and more. We will use the ROUGE-L metric which focuses on the longest common subsequence (LCS) of words in both sentences.

Incorporating LCS within the context of evaluating captions involves perceiving a generated sentence as a sequential arrangement of words. The underlying concept

revolves around the notion that an increased length of the LCS shared by two distinct sentences corresponds to a heightened level of similarity between the two sentences. In this regard, we introduce the application of an LCS-centered F-measure to measure the degree of similarity between two sentences, denoted as X, Y, r, and c. X, representative of a reference sentence with a length of r, and Y, the candidate sentence having a length of c. $LCS(X, Y)$ represents the length of the longest common subsequence between X and Y. The ROUGE-L is calculated as follows:

$$R_{lcs} = \frac{LCS(X, Y)}{r}$$

$$P_{lcs} = \frac{LCS(X, Y)}{c}$$

$$F_{lcs} = \frac{(1 + \beta^2)\, R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

The beta factor controls the relative weight assigned to recall and precision:

- When $\beta > 1$, recall (R_{LCS}) is given greater importance than precision (P_{LCS}).
- When $\beta < 1$, precision is prioritized over recall.
- When $\beta = 1$, recall and precision are weighted equally, resulting in the balanced F-measure.

The implementation we use sets β to 1.2.

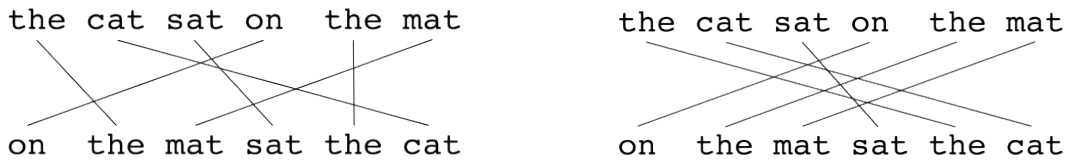
ROUGE-L captures sentence-level structure effectively but is limited by its reliance on sequential word order, which can overlook alternative word arrangements in longer or more flexible sentences — naturally, the values for ROUGE-L range between 0 and 1.

METEOR

The Metric for Evaluation for Translation with Explicit Ordering (METEOR) [DL] employs a weighted F-score that considers the matching of individual words along with a penalty mechanism to address inaccuracies in word order.

Initially, the objective is to match each word in the candidate sentence to a word in the reference sentence. This is achieved by first examining exact matches, then exploring matches after applying Porter stemming [Por], then utilizing synonymy from WordNet[6] and lastly, match phrases if they are listed as paraphrases in a language appropriate paraphrase table (see [DL], 3.2). These approaches create so-called “alignments” (for example, see Figure 2.7).

[6] `https://wordnet.princeton.edu/`, last accessed 13.03.2025, 15:09



(a) Alignment A (b) Alignment B

Figure 2.7: Example of possible alignments for the reference sentence "the cat sat on the mat" and the candidate sentence "on the mat sat the cat". Source: [Wik]

Once all possible alignments are established, the final alignment is determined as the largest subset of all matches that satisfies the following criteria, listed in order of importance:

1. Ensure that each word in both sentences is aligned to at most one word.
2. Maximize the total number of words covered across both sentences.
3. Minimize the number of chunks, where a chunk is defined as a sequence of matches that are contiguous and identically ordered in both sentences.
4. Minimize the sum of absolute differences between the starting indices of matched words in the two sentences. Looking back at our example in Figure 2.7, this means that alignment A is preferred over alignment B because the sum of the absolute difference between the starting indices in the sentences is minimized. In the case of ties, prioritize aligning phrases that occur in similar positions within both sentences.

Once this alignment is established, the number of mapped unigrams between the two texts is denoted as m. Subsequently, precision and recall are computed as ratios of m over the lengths of the candidate c and reference r sentence, respectively. F_{mean} is calculated as

$$Recall = \frac{m}{r}$$

$$Precision = \frac{m}{c}$$

$$F_{\text{mean}} = \frac{P \cdot R}{\alpha \cdot P + (1-\alpha) \cdot R}$$

To account for the word order in the candidate, we introduce a penalty function as

$$\text{Pen} \ = \gamma \left(\frac{ch}{m} \right)^{\beta}, \text{ where } 0 \leq \gamma \leq 1$$

In this context, ch represents the number of matching chunks, and m is the number of overall matches. Consequently, when a majority of matches are consecutive, the count of chunks decreases, resulting in a reduction in the penalty. The parameters α, β, and γ are tuned to maximize correlation with human judgment in a certain language. Ultimately, the METEOR score is determined as

$$Score = (1 - Pen) \cdot F_{mean}, \text{ where } 0 \leq \text{Score} \leq 1$$

Our captions are in English; we, therefore, use the setting from the original paper [DL] for the English language, setting α to 0.85, β to 0.2, and γ to 0.6. As METEOR is based on Recall and Precision in ranges between 0 and 1.

METEOR offers a more comprehensive evaluation of translation quality compared to metrics like BLEU or ROUGE, as it considers factors such as recall, stemming, synonym matching, and word order, enabling it to better capture fluency and semantic similarity.

Additionally, its strong correlation with human judgment highlights its effectiveness in evaluating machine translation tasks (and might directly relate to comparing image captions). However, METEOR's algorithm is computationally intensive and complex, which can make it slower and more resource-demanding than simpler metrics. The weights assigned to various measures may not always align with the desired evaluation criteria, potentially affecting its representativeness for specific tasks.

CIDEr

The Consensus-based Image Description Evaluation (CIDEr) [VZP] is based on term frequency-inverse document frequency (TF–IDF) computation. Written as a formula, we define Ω as the vocabulary of all n-grams, and I is the set of all images in the dataset. The goal is to evaluate how well a candidate sentence c_i matches a set of image captions $S_i = \{s_{i1}, ..., s_{im}\}$. The number of times a n-gram ω_k occurs in a reference sentence s_{ij} is denoted by $h_k(s_{ik})$ or $h_k(c_i)$ for the candidate sentence c_i. The TF-IDF weighting $g_k(s_{ij})$ for each n-gram ω_k is computed using:

$$g_k\left(s_{ij}\right) = \frac{h_k\left(s_{ij}\right)}{\sum_{\omega_l \in \Omega} h_l\left(s_{ij}\right)} \log \left(\frac{|I|}{\sum_{I_p \in I} \min\left(1, \sum_q h_k\left(s_{pq}\right)\right)} \right)$$

The first term in this equation measures the term frequency (TF) and the second term measures the rarity based on the inverse document frequency (IDF). TF tends to assign greater importance to n-grams that are frequently present in the reference sentence describing an image. In contrast, IDF diminishes the significance of n-grams

that commonly appear across all images in the dataset. Essentially, IDF offers a metric for assessing word saliency by reducing the weight of popular words that are likely to convey less visual information. The computation of IDF involves taking the logarithm of the ratio of the total number of images in the dataset $|I|$ to the number of images where the specific n-gram ω_k occurs in any of their reference sentences.

The computation of CIDEr_n score for n-grams of length n involves determining the average cosine similarity between the candidate sentence and the reference sentences. This calculation takes into consideration both precision and recall:

$$\text{CIDEr}_n\left(c_i, S_i\right) = \frac{1}{m} \sum_j \frac{\boldsymbol{g^n}\left(c_i\right) \cdot \boldsymbol{g^n}\left(s_{ij}\right)}{\left\|\boldsymbol{g^n}\left(c_i\right)\right\| \left\|\boldsymbol{g^n}\left(s_{ij}\right)\right\|}$$

where $\boldsymbol{g^n}(c_i)$ is a vector formed by $g_k(c_i)$ corresponding to all n-grams of length n, and $\|\boldsymbol{g^n}(c_i)\|$ is the magnitude of the vector $\boldsymbol{g^n}(c_i)$. Similarly, for $\boldsymbol{g^n}(s_{ij})$.

To combine short n-grams with n-grams of higher order which capture richer semantics as well as grammatical properties, the scores are combined by a weighted sum which results in the final score:

$$\text{CIDEr}\left(c_i, S_i\right) = \sum_{n=1}^{N} w_n \, \text{CIDEr}_n\left(c_i, S_i\right)$$

In some cases, the basic CIDEr metric produces higher scores when words of higher confidence are repeated over long sentences. The authors introduce a Gaussian penalty based on the difference between candidate and reference sentence lengths to reduce this effect. The authors use $\sigma = 6$.

$$\text{Penalty} = e^{-\frac{\left(\text{Length}_{\text{hyp}} - \text{Length}_{\text{ref}}\right)^2}{2\sigma^2}}$$

The sentence length penalty can be manipulated by repeatedly using confident words or phrases to reach the target sentence length. To address this, the authors introduce clipping to the n-gram counts in the CIDEr numerator. Specifically, for each n-gram, the number of occurrences in the candidate is limited to match the number in the reference. This discourages excessive repetition of n-grams beyond their occurrence in the reference sentence.

The CIDEr Score has by nature, a range of 0 to 1, as it is based on the cosine similarity of positive vectors. The authors multiply the score by 10. Shifting the values to a range of 0 to 10. The authors of Yang et al. [YZJ+] and Moratelli et al. [MBM+] proceed to additionally multiply final values with 100, leading to a final range of 0 to 1000. We also implemented this approach to maintain comparability across experiments.

Because CIDEr is based on the consensus using multiple reference captions, it is used with datasets with more than 1 image caption per image, e.g., PASCAL-50S [VZP] and ABSTRACT-50S [VZP], which both have 50 captions per image.

SPICE

Semantic Propositional Image Caption Evaluation (SPICE) [AFJG] is the newest metric compared to the previously presented. The original paper explained that the previous metrics primarily focus on n-gram overlap, which can lead to sentences with a high score but two very different meanings. They propose an approach where a graph-based semantic representation is created for an image called a *scene graph*. The scene graph directly represents the objects, attributes, and relationships present in image captions. This representation removes many of the complexities and unique language patterns found in natural language, simplifying the visual information analysis.

To create the scene graph a dependency parser [KM] that has been pre-trained on a vast dataset is employed to establish the syntactic relationships among words within the caption. Then they use a rule-based system [SKC+] to map from the dependency trees to scene graphs. The metric is calculated using candidate and reference scene graphs, as well as an F-score based on the intersection of logical tuples representing semantic statements within the scene graphs.

2.2 Fashion Item Recommendation

In the context of fashion, recommendation systems play a crucial role in helping users navigate large item collections and discover relevant products. Unlike traditional recommendation tasks, fashion recommendation is inherently multimodal, relying on visual content, textual descriptions, and structured metadata [DNR+]. The combination of these modalities and their methods allows for a richer understanding of items and user preferences, addressing challenges such as subjective style perception [HG, BHV] and dynamic fashion trends [MMH+, MBS]. The primary goal of recommendation systems is to present users with relevant items by predicting their preferences and ranking these items in order of relevance.

He et al. [HMb] proposed Visual Bayesian Personalized Ranking (VBPR). Bayesian Personalized Ranking is a recommendation algorithm that utilizes Bayesian inference to estimate the ranking preferences of users for items, enabling personalized recommendations based on the user's historical interactions. VPBR extends it by incorporating a content-based preference factor which is based on the visual signal of an item. A pre-trained CNN is used to extract the latent feature. Kang et al. [KFWM] extend this idea to using a full end-to-end trained model instead of a pre-trained CNN. He and McAuley [HMa] combined the VBPR with seasonality and temporal changes.

While VBPR has proven to be useful, it often learns an item's visual qualities based on its category rather than its specific style (such as informal, aesthetic, or formal). To

connect clothing items to a certain style, labeled data is needed. Liu et al. [LWW] presented a way to learn the style of items by using user-item matrices to model the style as the difference between item and category. User-item matrices are commonly used in recommendation systems and collaborative filtering techniques to represent the interactions or preferences of users for various items.

Previously mentioned approaches [HMb, HMa, KFWM, LWW] combine two main components for recommendation: visual features of fashion items and user-item preference signals derived from implicit or explicit feedback. These methods extend traditional collaborative filtering by incorporating visual information.

In contrast, traditional item-item [LSY] collaborative filtering determines item similarity based on historical user interactions. The core idea is that if two items are frequently co-interacted with by the same users, they are likely to be similar or associated in some way. The similarity is then quantified using measures such as cosine similarity or Pearson correlation coefficient.

Several papers presented algorithms that are able to detect visual information (sleeve length, color, pattern) from clothing photographs [YR, YLL, YKB]. Still, fashion items do not only benefit from visual features but also from textual information provided in the description, e.g., material used, cut, and more. Laenen et al. [LM] propose an attention-based fusion method for outfit recommendation which fuses the information in the product image and description focusing on outfit recommendation.

2.2.1 Evaluation Metrics

To assess the performance of recommendation systems, one measures the relevance and ranking quality of recommended items. Two commonly used metrics are Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP). Both NDCG and MAP offer valuable insights into recommendation performance, with NDCG focusing on the ranking order and MAP emphasizing precision at relevant positions.

Normalized Discounted Cumulative Gain (NDCG)

Normalized Discounted Cumulative Gain (NDCG) [JK] is a ranking-based evaluation metric that takes into account the position of relevant items in the ranked list. The Discounted Cumulative Gain (DCG) at rank k is computed as:

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

Here, rel_i denotes the relevance score of the item at position i. NDCG can handle both binary relevance (relevant vs. non-relevant) and graded relevance (e.g., highly relevant, moderately relevant, irrelevant). In the case of binary relevance, the relevance score is typically set to 1 for relevant items and 0 for non-relevant items, which simplifies the

Discounted Cumulative Gain (DCG) calculation. The DCG value is then normalized by the Ideal DCG (IDCG), which represents the maximum possible DCG for the given list:

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}}$$

NDCG ranges from 0 to 1, where a value closer to 1 indicates a more accurate ranking with relevant items appearing at the top of the list.

Mean Average Precision (MAP)

Mean Average Precision (MAP) [BV] is a precision-based evaluation metric that measures the overall precision of a recommendation system across multiple queries or users. It is well-suited for tasks with binary relevance labels (i.e., relevant or not relevant). The Average Precision (AP) for a single user is calculated as the mean of the precision scores at each rank where a relevant item appears:

$$\text{AP} = \frac{1}{m} \sum_{k=1}^{n} P(k) \cdot \text{rel}(k)$$

where $P(k)$ is the precision at position k, $\text{rel}(k)$ is an indicator function that equals 1 if the item at position k is relevant, and m is the total number of relevant items. MAP is obtained by averaging the AP over all users:

$$\text{MAP} = \frac{1}{|U|} \sum_{u \in U} \text{AP}_u$$

MAP also ranges between 0 and 1, with higher values indicating better overall precision and recommendation performance. It provides an aggregated measure of ranking quality by rewarding systems that consistently rank relevant items higher across multiple queries.

CHAPTER 3

Data

In this chapter, we present the two datasets used in this thesis: the H&M dataset, released as part of a Kaggle challenge in February 2022, and the Fashion Captioning Dataset (FACAD)[1], created by Yang et al. [YZJ+].

For FACAD (3.1), we describe the procedure used by the authors to extract attributes and categories. In contrast, the H&M dataset provides predefined categories, while attributes were manually extracted following the procedure from Yang et al. (see Section 3.2 for details). Finally, we explore the transactional data of the H&M dataset and outline the procedure used to split it into training and test sets.

In Section 3.3 we highlight the differences between the two datasets for the fashion captioning task.

3.1 FACAD

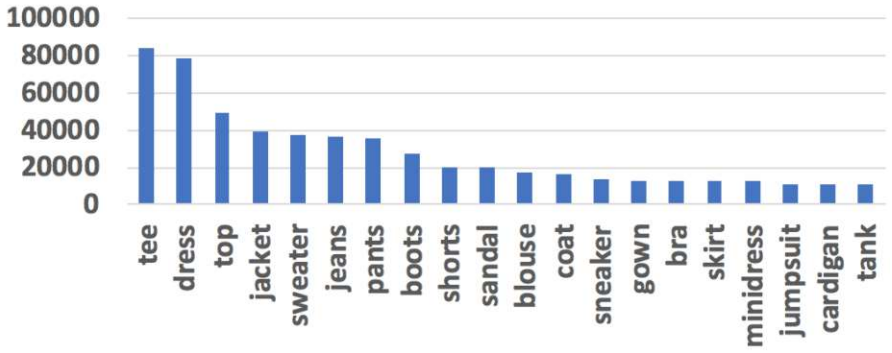
We already presented the dataset in Section 2.1.2 but we will cover details here.

The dataset contains 993K images and 130K captions that were split into 794K (∼80%) image-description pairs for training, 99K (∼10%) for validation, and the remaining 100K (∼10%) for testing.

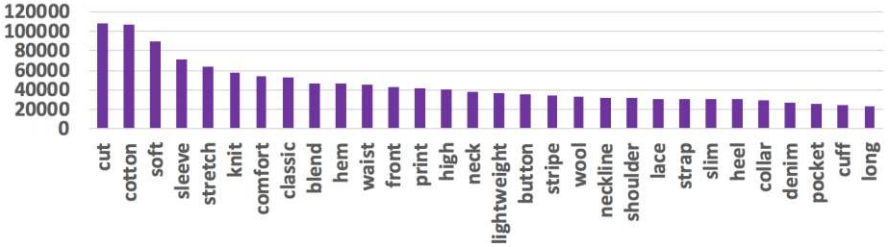
However, the dataset that is currently provided by the authors has a different distribution, with 888,293 pairs designated for training, 19,915 for validation, and 101,225 for testing (a total of 1,009,463 samples). As per the authors, this was done so that validation does not take as long[2]. Therefore the charts in Figure 3.1 do not represent the dataset used in this thesis anymore, but they still give an idea about the general distribution of the

[1] https://github.com/xuewyang/Fashion_Captioning, last accessed 13.03.2025, 15:40

[2] https://github.com/xuewyang/Fashion_Captioning/issues/5, last accessed 13.03.2025, 15:40



(a) Number of items in the top-20 categories of the FACAD dataset.



(b) Number of items in the top-30 attributes of the FACAD dataset.

Figure 3.1: Distributions of the categories and attributes of the FACAD dataset. Source: [Yan], page 5

dataset. It should be mentioned, that the authors do not explicitly say if the plots show train data or the whole dataset.

On average, each clothing item has 6∼7 images, and the authors reported sizes of 1560x2392, which, to our best knowledge, cannot be recreated anymore as the images provided by the authors are rescaled to 256x256, and the image URLs provided lead to thumbnails of the images of sizes 60x90.

The dataset includes 78 categories, most of which are upper garments, e.g., tee, top, and jacket (see Figure 3.1a). The authors extracted the categories by taking the last word of the item title and manually selecting, filtering, and merging similar categories. Then, only categories were kept that contained over 200 items.

The authors extracted the attributes from the item's title, description, and metadata. Specifically, nouns and adjectives were extracted from the title using the Stanford Parser [SBMN], and a word was selected as an attribute if it also appeared in both the caption and metadata.

To ensure that captions were clean, the descriptions were tokenized using the NLTK tokenizer, and non-alphanumeric characters were removed. Additionally, all caption

words were converted to lowercase. Although not specifically mentioned in the paper, the attributes were lemmatized, and words that were connected by hyphens were split, e.g., t-shirt turns into t and shirt and words like sleeves are lemmatized to sleeve.

This approach initially identified over 3,000 attributes, but only those associated with more than 10 items were retained, resulting in a refined list of 990 attributes. On average, each item is linked to approximately 7.3 attributes. The distribution of items associated with the top 30 attributes is presented in Figure 3.1b. The attributes were provided by the authors.

3.2 H&M

3.2.1 Image Captioning

The H&M Personalized Fashion Recommendation dataset[3] [Linb] was provided for a recommendation kaggle challenge in February 2022.

It includes information about 1,372,980 customers on 105,542 articles with 31,788,324 transactions. Fashion items in this dataset include attributes about their appearance, a detailed description, and an image.

Preprocessing. The data preprocessing involved multiple steps to clean and prepare the dataset for further use in the fashion captioning task. The primary focus was on preparing the article dataset and associated images for creating an image captioning dataset.

The first step was to clean the dataset by removing any articles that did not have a detailed description (`detail_desc`) or a corresponding image file. This reduced the initial dataset size from 105,542 to 104,696 articles.

The articles dataset included a hierarchical product category information, specifically `product group`, `product type`, and `product name`. The `product type` level was chosen for filtering due to its appropriate level of granularity. The initial analysis showed:

- 19 unique values in the `product group` column.
- 131 unique values in the `product type` column.
- 45,567 unique values in the `product name` column.

We decided to filter based on the number of articles per product type and kept only those categories with at least 7 items (because 25% of the product categories have less than 7 items, keeping 75% of the initial items). This was done to keep articles that provide

[3] `https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations`, last accessed 26.03.2025, 16:36

enough captions to be trained by the model. This step reduced the number of product types from 131 to 100. Non-fashion-related categories, such as *Dog Wear* and *Sleeping Sack*, were removed manually to retain only fashion and accessory items. This further refined the dataset to 89 relevant product types.

The cleaned dataset was split into training, validation, and test set, maintaining a distribution of 80% training, 10% validation, and 10% test data. The final count of articles after preprocessing was 104,232, distributed as follows:

- Training set: 83,385 articles
- Validation set: 10,423 articles
- Test set: 10,424 articles

The preprocessing steps led to the following reductions in dataset size:

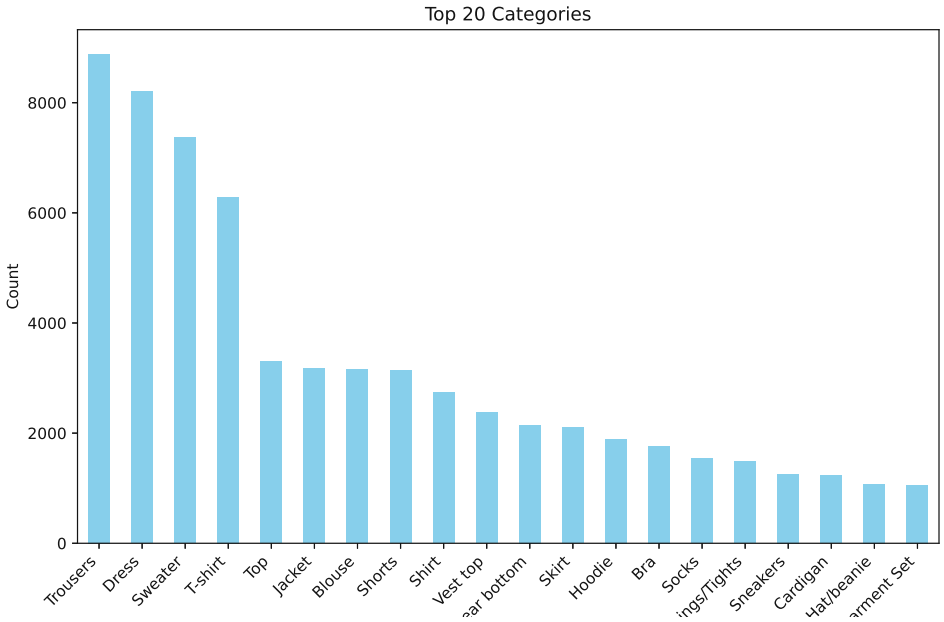
- Initial number of items: 105,542
- After removing items without descriptions: 105,126
- After removing items without corresponding images: 104,696
- After filtering categories with fewer than 7 items: 104,232

To compare the two datasets we plotted the top 20 categories and 30 top attributes as seen in Figure 3.2. Besides the differences mentioned previously in Section 3.1, we see that the attributes are also very similar and differ by frequency.

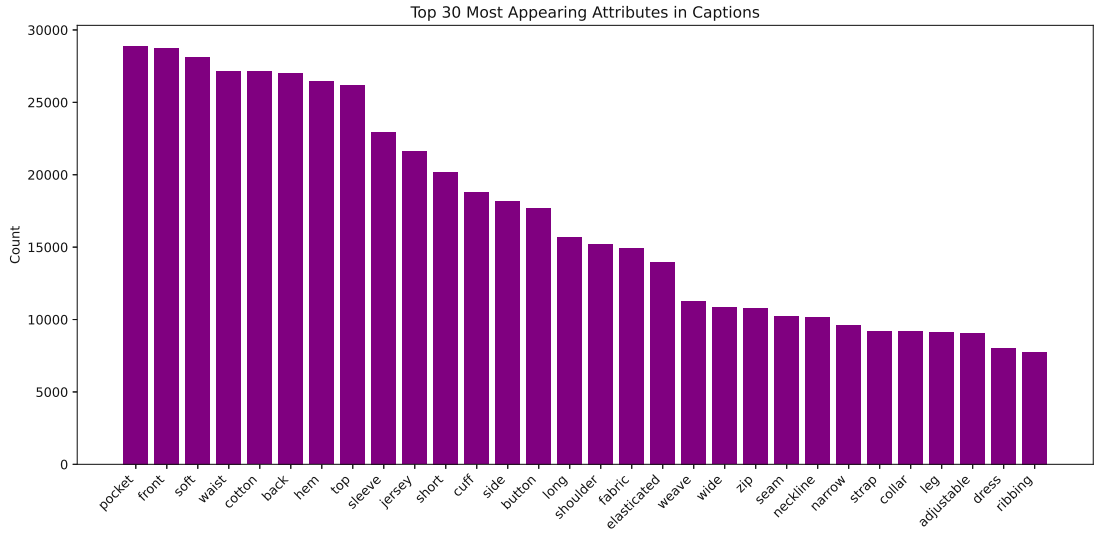
Attributes. To compute meaningful statistics such as the average precision (see Section 4.3.1), attributes within the product descriptions need to be extracted. The `detail_desc` column was used to extract nouns, adjectives, and proper nouns based on Universal POS tagging definitions[4], namely the stages `ADJ`, `NOUN`, `PROPN`. The extraction process used the Stanza NLP library [QZZ+] to identify and filter these attributes. Before tagging, the descriptions are lowercase, and hyphen-connected words, e.g., t-shirt, are split. Then, we extract the lemmatized attributes and filter, keeping only attributes appearing at least 10 times. This is done to ensure the significance and relevance of attributes kept.

Using the train set items, we then collect all extracted attributes (1014 in total). These are then used as a pool to select attributes from a generated caption and to generate the ground truth for the test set. However, due to the threshold of 10, this leads to one item (`0512988022.jpg`, a lint roller) in the test set not having a ground truth, which we exclude in our computations for the average precision described in Section 4.3.1.

[4] `https://universaldependencies.org/u/pos/`, last accessed 13.03.2025, 15:47



(a) Number of items in the top-20 categories of the H&M dataset train set.



(b) Number of items in the top-30 attributes of the H&M dataset train set.

Figure 3.2: Distributions of the categories and attributes of the H&M dataset train set.

3.2.2 Recommendations

The dataset includes information on customers' purchase history across time (about 2 years, from 2018 to 2020), along with supporting metadata. The goal of the kaggle challenge was to predict what articles each customer would purchase in the 7 days immediately after the training data ends.

A submission sample was provided with all customer ids. This was the test set for this challenge with 1,371,980 customers in total. For the challenge, participants submitted a file including customer ids and up to 12 article ids representing the items the customer would buy in the next 7 days. The submissions are evaluated according to the Mean Average Precision with a cutoff of 12 items (MAP@12) calculated as:

$$\text{MAP@12} = \frac{1}{U}\sum_{u=1}^{U}\frac{1}{\min(m,12)}\sum_{k=1}^{\min(n,12)} P(k)\times \text{rel}(k)$$

where U is the number of customers, $P(k)$ is the precision at cutoff k, n is the number of predictions per customer, m is the number of ground truth values per customer and $rel(k)$ is an indicator function equaling 1 if the item at rank k is a relevant (correct) label, zero otherwise.

For evaluation, participants submitted predictions for customers who were not included in the train set (9699 in total). Since there is no penalty for predicting up to 12 items, even for customers who ordered fewer, it was beneficial to always provide 12 predictions per customer. Customers who made no purchases during the test period were excluded from scoring.

The best scores reported on the leaderboard are between 0.038 and 0.035 MAP@12 with a total of 2954 submissions.

For a better understanding of the data, we visualized the number of transactions per month (see Figure 3.3). The plot shows a seasonal shopping behavior which we assume is caused by different sales. The largest number of purchases happen during summer, probably due to the Summer Sale starting mid-June. Summer clothing items are usually cheaper than winter clothing items because there is less material. This leads to extremely low prices, pushing customers to buy beyond their needs, and increasing transaction numbers for this month beyond average. We note that September 2018 and 2020 are only partially available in the data, therefore, not fully representable.

Also, with the average number of purchases per customer being 23.3 and the median being 9 purchases this indicates a right-skewed data, meaning that there is a small number of customers that purchase a large number of items. One can say, that the median represents an "average customer". To give an idea about how much the top customers buy, we visualized the transaction counts of the top 100 users as a box plot (see Figure 3.4). The customer with the biggest number of transactions has a total count

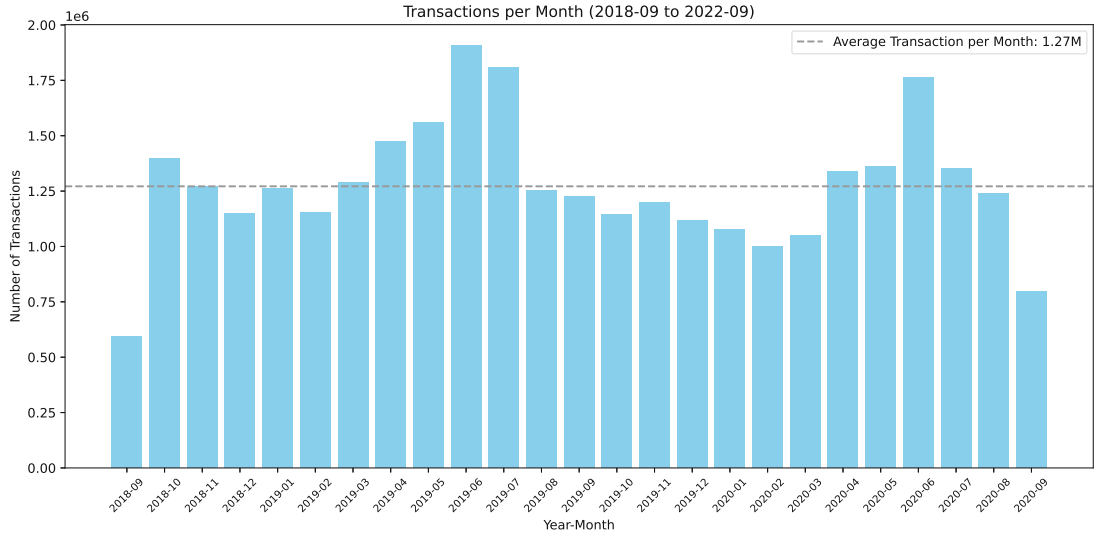


Figure 3.3: Bar plot showing the total number of transactions (purchases) per month. Note: September 2018 and 2020 are only partially available in the dataset.

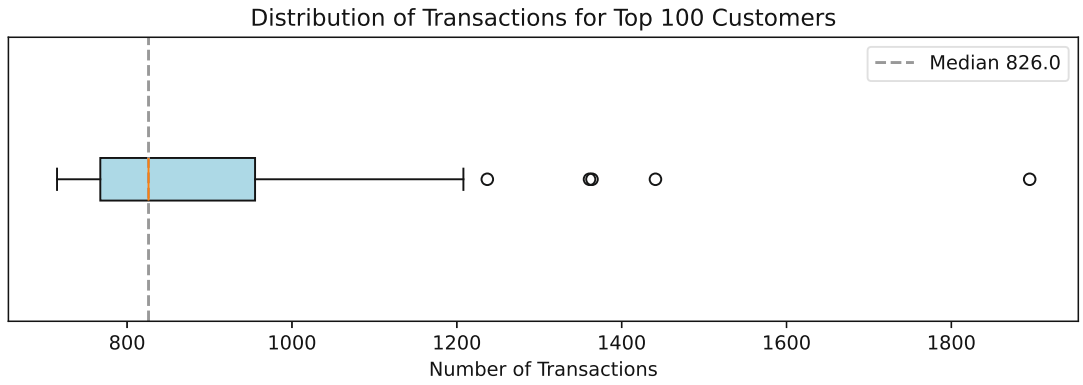


Figure 3.4: Boxplot depicting the 100 customers with the most transactions. There the Median lies with 826 transactions over a span of 2 years.

of 1,895 over a time span of 2 years. This means buying about 18 clothing items each week. However, the data only reports transactions but not returning items.

Data Split. Given the large size of the dataset—over 31 million transactions from more than 1 million customers—we opted to omit cross-validation. Instead, we applied an 80/20 temporal split to divide the transactions into training and test sets. In this temporal split, earlier transactions for each user are assigned to the training set, while later ones are included in the test set. Users with only a single transaction are kept entirely in the test set. Prior to splitting, we removed articles missing either an image or a description, consistent with the preprocessing for image captioning. The final dataset

Table 3.1: Comparison of both datasets. CAT: category, AT: attribute, CAP: caption, *The H&M dataset includes original images of different sizes, but a majority are around 1166x1750 pixels large. **The original images for FACAD are not provided by the authors; they are only preprocessed versions downsized to 256x256.

Dataset	#img	img size	#img (per cap.)	vocab size	#CAP avg len	#CAT	#AT
H&M	104K	*∼1166x1750	1	7.6K	23.9	89	1014
FACAD	993K	**256x256	$7 \sim 8$	15.8K	21.0	78	990

consists of 31,400,864 transactions, with 24,516,873 in the training set and 6,883,991 in the test set. Due to the removal of the transactions based on the items missing images or descriptions, we removed 2,103 users, leaving a total of 1,360,178 users and 103,251 unique items. This leaves 981 that are not officially bought in the dataset. Due to the splitting of single-transaction users, we have more unique users in the test set.

3.3 Comparison of Datasets

We use this section to emphasize the differences between both datasets to also provide a better understanding of the results presented in Chapter 5. Both datasets are domain-specific fashion datasets but differ in many aspects (see Table 3.1). One of the biggest differences is the size, the FACAD dataset has almost 10 times the size of the H&M dataset in terms of images. Providing more variety of item perspectives, including different angle shots, with and without a model and a material shot (see Figure 3.5a). Compared, the H&M dataset only offers single-item images without a model, sometimes only showing part of the item.

Furthermore, the H&M dataset shows a 1-to-1 relationship between captions and images, whereas the FACAD dataset sometimes includes multiple items with the same description but differing in color or single items with many images.

Comparing the captions, the H&M dataset has more concise captions describing the item's "tailor" details, e.g., "frill-trimmed shoulder straps" or "tapered legs with ribbed hems", whereas FACAD captions tend to be more "enchanting" as the authors comment in their paper. It includes expressions made for selling, e.g., "this neutral hued cotton sweater you'll wear everywhere" or "holiday red get some seasonal sparkle in a dazzling dress that is ready to make you and everyone at the party very merry indeed". Interestingly, the H&M captions do not include colors explicitly.

Comparing categories, the H&M dataset has more trouser items (see Figure 3.2a). This could be due to the H&M dataset only having one category for trousers and not specifying between jeans and pants, leading to all pants-like garments being merged into this category. Otherwise, both datasets show similar category distributions.

this **shawl collar jersey** blazer **bloom** with **black** and **white floral** patterning inspired by the **work** of rising spanish photographer coco capit n

(a) Item sample from FACAD test dataset. Source: created by the author

Short, **sleeveless dress** in an **airy cotton weave** that is **open** at the **back** with a **tie**. **Adjustable** frill-trimmed **shoulder straps**, a concealed **zip** in the **side**, **seam** at the **waist** with **elastication** at the **back** and a flared **skirt**. Unlined.

(b) Item sample from H&M test dataset. Source: created by the author

Figure 3.5: Item samples from both datasets showing the image(s) and corresponding captions. Bold words are defined attributes. Source: created by the author

CHAPTER 4

Development of the Solution

In this section, we present the steps that were followed to develop the solution, as well as the decisions taken. We first explain our choice of models used for image captioning, BLIP-2, and LLaVA (see Section 4.1) and their architecture. As one of the goals is to compare results from the fine-tuned models with the model from Yang et al. [YZJ+], we present their model in Section 4.1.3.

We then continue to explain the methods and setup for fine-tuning in Section 4.2 including the experiment done to determine the hyperparameters (4.2.2). Section 4.3 describes the additional metrics we used to evaluate the captions (based on Yang et al.).

Finally, Section 4.4 explains the recommendations experiment setup including the different methods used for feature extraction and the algorithms used for comparison.

4.1 Models for Image Captioning

For the task of image captioning, we required encoder-decoder-based or multimodal-capable models, as covered in Section 2.1. The main challenge we encountered when working with machine learning models (and specifically trending LLMs) was that they were not provided in an open-source manner. As a result, we were unable to investigate their performance on different datasets or reproduce results. To answer RQ1 in Section 1.3, we focused on open-source models to promote transparency and reproducibility of the results achieved in this master thesis. Therefore, we did not use powerful models such as ChatGPT-4 [Ope] or Gemini [SP].

We used the open-source platform HuggingFace[1], which offers access to more than 900,000 models. Beyond being open-source, Hugging Face provides various interfaces that simplify the use of these models, a key factor in our decision-making process. Given the scope

[1] https://huggingface.co/, last accessed 13.03.2025, 16:08

limitations of a master's thesis, a comprehensive comparison of models across different platforms was not feasible. Therefore, we focused on models supported by Hugging Face's Transformers library due to their ease of integration for fine-tuning and inference.

BLIP (Bootstrapping Language-Image Pre-training) [LLXH] is the most popular model for the "image-to-text" task available on Hugging Face. However, Li et al. recently introduced BLIP-2 [LLSH], a successor model. We chose to use BLIP-2, specifically designed for image-to-text generation and demonstrating notable performance improvements, outperforming Flamingo80B by 8.7% on zero-shot VQAv2 while requiring 54 times fewer trainable parameters. More broadly, BLIP-2 achieves state-of-the-art performance while being more compute-efficient compared to existing methods, including its predecessor BLIP, across a range of vision-language tasks such as visual question answering, image-text retrieval, and image captioning. Additionally, BLIP-2's open-source nature and the possibility of fine-tuning using custom datasets made it particularly suitable for our use case.

For comparative analysis, we experimented with different variants of BLIP-2, specifically those that leverage different language models for text generation, including OPT (Open Pre-trained Transformer) [ZRG+] in its 2.7B and 6.7B parameter versions, as well as FLAN-T5 [CHL+] in its XL and XXL versions. More details of the architecture can be found in Section 4.1.1.

Besides other different encoder-decoder models, LLaVA-1.5 (an improved version of LLaVA [LLWL]) is among the most popular models for image captioning on Hugging Face. LLaVA, short for Large Language and Vision Assistant, is a multimodal model that integrates a vision encoder with a large language model, trained on machine-generated language-image instruction-following data from GPT-4. In contrast to BLIP-2, LLaVA aims to function as a conversational assistant that can interactively respond to visual inputs, demonstrating chat-like capabilities. It is optimized for generating detailed, instruction-following responses, similar to an AI like GPT-4 but in a multimodal context. LLaVA requires a specific prompt to generate meaningful multimodal responses, whereas BLIP-2 does not rely on prompts, instead focusing on automatically understanding and aligning image-text pairs. A detailed description of the architecture can be found in Section 4.1.2.

As a baseline for the results using the FACAD dataset, we use the results presented by Yang et al. [YZJ+], which we also used as a reference for the experiment setup. A more detailed description of their model can be read in Section 4.1.3.

4.1.1 BLIP-2

The BLIP-2 architecture can be seen in Figure 4.1. The proposed method uses frozen vision and language models, effectively bridging the modality gap between the unimodal models. Frozen, in the context of machine learning models, describes a model that was stopped at one point during training, and the parameters were no longer adjusted. It is

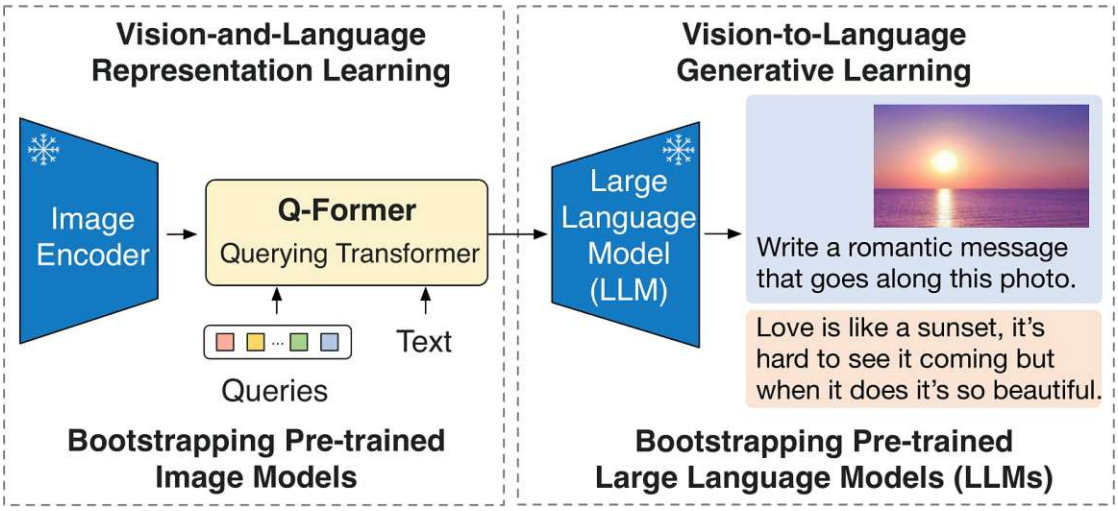


Figure 4.1: In the framework of BLIP-2, a lightweight Querying Transformer is pre-trained using a two-stage strategy to address the modality gap. In the initial stage, vision-language representation learning is initiated with a frozen image encoder. Subsequently, the second stage involves vision-to-language generative learning with a frozen LLM. Source:[LLSH], page 1

computationally efficient, as it requires training fewer parameters compared to previous end-to-end training approaches.

The authors employed a lightweight Querying Transformer (Q-Former) as a bottleneck between the frozen image and text encoders. Initially, the image undergoes processing by the image encoder to extract visual features, and the resulting outputs are then fed to the language model for comprehension. However, a challenge arises, as the frozen language model lacks training in image data and struggles to interpret the extracted visual representations effectively. To address this issue, the Q-Former employs a set of learnable querying vectors and undergoes pre-training in two stages: (1) vision-language representation learning with a frozen image encoder and (2) vision-to-language generative learning with a frozen text encoder.

Training. The Q-Former comprises two sub-modules (see Figure 4.2, left): (1) an image transformer, which engages with visual features derived from the frozen image encoder, and (2) a text transformer responsible for encoding and decoding texts. Employing a set of learnable querying vectors, the Q-Former extracts relevant visual features that contain the most informative aspects of the text associated with the image. In the first phase, the vision-language representative learning, these are jointly optimized for three pre-training objectives where each objective uses a different attention mask between queries and text (see Figure 4.2, right).

Image-Text Contrastive Learning (ITC) tries to adjust the model to make correct pairs more similar while making incorrect pairs less similar using the similarity between

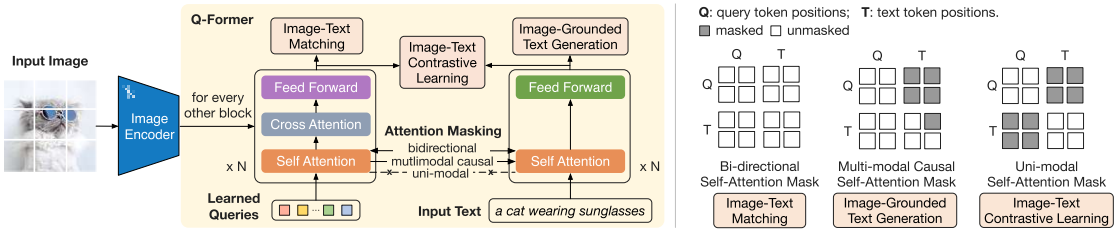


Figure 4.2: (**Left**) The model architecture of the Q-Former and the first-stage vision-language representation learning objectives of BLIP-2. Three objectives are jointly optimized to ensure that the queries (a set of learnable embeddings) effectively extract the visual features most relevant to the text. (**Right**) The self-attention masking strategies are used for each objective to regulate the interaction between the queries and the text. Source:[LLSH], page 3

the queries and the text representation. To prevent information leakage, a unimodal self-attention mask is applied, ensuring that queries and text cannot directly interact (see right of Figure 4.2).

Image-grounded Text Generation (ITG) trains the Q-Former to generate text conditioned on input images. Since the Q-Former architecture does not allow direct interaction between the image encoder and text tokens, the queries extract visual features, passing this information to the text tokens via self-attention layers. To achieve this, a multimodal causal self-attention mask (see right of Figure 4.2) ensures that queries interact only with each other, while text tokens attend to queries and earlier text tokens, facilitating effective image-to-text generation.

Image-Text Matching (ITM) aims to learn fine-grained alignment between image and text representations through a binary classification task, predicting whether an image-text pair is matched or unmatched. A bi-directional self-attention mask (see right of Figure 4.2) allows all queries and text tokens to interact, enabling the query embeddings to capture multimodal information. Each query embedding is processed through a two-class linear classifier to produce logits, which are averaged across queries to determine the matching score. Hard negative mining is employed to generate challenging negative pairs for training.

After the representative learning, the model is trained to generate text based on the generated queries. The authors explored two types of large language models (LLMs): decoder-based and encoder-decoder-based. For decoder-based LLMs, language modeling loss is employed, requiring the LLM to generate text conditioned on the visual representation provided by the Q-Former. For encoder-decoder-based LLMs, prefix language modeling loss is utilized, where the text is split into a prefix (combined with the visual representation as input to the encoder) and a suffix (used as the target for the decoder). The models used in the paper are the unsupervised-trained OPT model family [ZRG+] for decoder-based LLMs (in their 2.7B and 6.7B variation), and the instruction-trained FLAN-T5 model family [CHL+] for encoder-decoder-based LLMs (in their XL and XXL

variation).

The model is trained with the same train data as BLIP [LLXH] with 129M images in total.

4.1.2 LLaVA-1.5

LLaVA (Large Language and Vision Assistant) [LLWL] is a multimodal model designed for general-purpose visual and language understanding. It integrates a vision encoder (CLIP ViT-L/14, [RKH+]) with a large language model (Vicuna, [CLL+]) through end-to-end training.

The architecture is depicted in Figure 4.3. For an input image X_v, they use the pre-trained visual encoder g to extract visual features $Z_v = g(X_v)$. To integrate these visual features into the language model, a simple linear transformation is applied. This transformation uses a projection matrix W to convert Z_v into H_v, which matches the dimensions of word embeddings in the language model:

$$H_v = W \cdot Z_v, \quad \text{where} \quad Z_v = g(X_v)$$

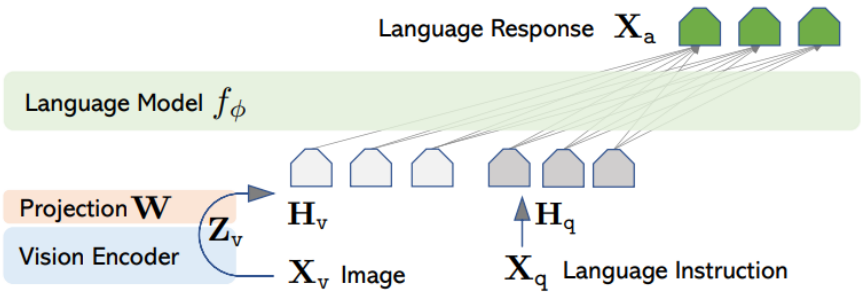


Figure 4.3: LLaVA network architecture. It includes a trainable linear projection layer W to convert visual features to language embeddings. Source: [LLWL], page 4

Training. For each image X_v, multi-turn conversation data is generated in the form of sequences $(X_q^1, X_a^1, \ldots, X_q^T, X_a^T)$, where T is the total number of turns, X_q and X_a the questions and answers. Each answer is treated as the assistant's response, and the instruction for the t-th turn, X_{instruct}^t, is defined as follows:

$$X_{\text{instruct}}^t = \begin{cases} \text{Randomly select } [X_q^1, X_v] \text{ or } [X_v, X_q^1], & \text{if } t = 1 \\ X_q^t, & \text{if } t > 1 \end{cases}$$

The language model is instruction-tuned on the prediction tokens using its original auto-regressive training objective.

For a sequence of length L, the probability of the target answers X_a is computed as:

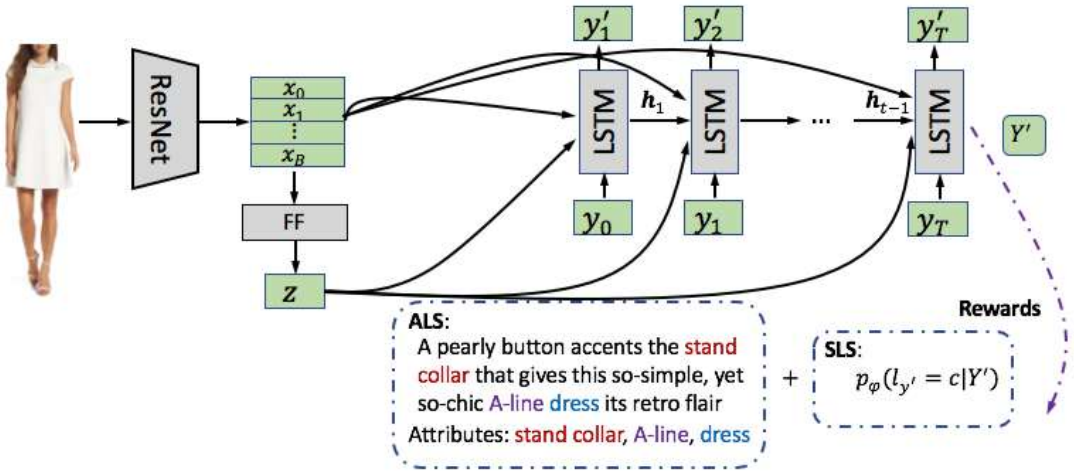


Figure 4.4: SRFC-architecture and proposed loss functions: attribute-level semantic (ALS) and sentence-level semantic (SLS). Source: [YZJ+], page 7

$$p(X_a|X_v, X_{\text{instruct}}) = \prod_{i=1}^{L} p_\theta(x_i|X_v, X_{\text{instruct},<i}, X_{a,<i}),$$

where θ is the trainable parameters and $X_{\text{instruct},<i}$ and $X_{a,<i}$ are the instruction and response tokens on all turns before the current prediction token x_i, respectively.

LLaVA is also trained in a two-stage approach where (1) the model is pre-trained for feature alignment by only training the projection layer and (2) fine-tuning it end-to-end where the projection layer and language model are trained. The vision encoder remains frozen throughout the training process.

LLaVA-1.5 [LLLL] added a two-layer MLP vision-language connector instead of a single linear projection matrix and added academic-task-oriented VQA data.

4.1.3 Semantic Rewards guided Fashion Captioning (SRFC)

This is the model presented by Yang et al. [YZJ+] which focuses on Fashion Captioning. The proposed model follows an encoder-decoder architecture, where a pre-trained ResNet-101 [HZRS] is used as the encoder to extract image features. This encoder is fine-tuned on the FAshion CAptioning Dataset (FACAD). The extracted image features are dynamically re-weighted using an attention mechanism, allowing the model to focus on specific regions of the image during caption generation. These features are then passed on to an LSTM decoder that generates captions word by word.

To ensure the captions accurately describe the fine-grained attributes of fashion items, the model incorporates a visual attribute predictor. This predictor extracts attributes from the images and embeds them into the input of the LSTM decoder, effectively

seeding the caption generation process with attribute-level information. The model also introduces two types of semantic rewards: the Attribute-Level Semantic (ALS) reward and the Sentence-Level Semantic (SLS) reward. The ALS reward encourages the model to generate captions containing correct attributes by matching n-grams in the generated sentences with ground-truth attributes. Meanwhile, the SLS reward ensures the global semantic consistency of the captions by matching the predicted category of the generated description with the ground-truth category. This is achieved through a pre-trained text classifier.

The training process consists of two stages. In the first stage, the encoder-decoder model is trained using the maximum likelihood estimate (MLE) to initialize the parameters. In the second stage, the model is fine-tuned with Reinforcement Learning (RL), incorporating the ALS and SLS semantic rewards. The RL training employs the REINFORCE algorithm to optimize the non-differentiable rewards, with a baseline reward used to stabilize the training process. The general loss function combines MLE, semantic rewards, and attribute prediction to jointly optimize the model.

The captions were carefully preprocessed resulting in a vocabulary of 15,807 words. The model was trained using the Adam optimizer, starting with a learning rate of 1e-4, which was gradually annealed. Training took approximately four days on two NVIDIA 1080 Ti GPUs.

The results demonstrate improvements in captioning performance compared to baseline image captioning models. The introduction of semantic rewards led to enhanced attribute precision and category accuracy, while the use of FACAD enabled the generation of detailed and expressive captions tailored to the fashion domain.

4.2 Fine-tuning

4.2.1 LoRA

LoRA (Low-rank adaptation) [HSW+] significantly reduces the number of trainable parameters by decomposing the weight update matrix ΔW into the product of two low-rank matrices B and A. During fine-tuning, the original pre-trained model weights W are updated as: $W + \Delta W = W + BA$, where $W \in \mathbb{R}^{d \times k}$, $B \in \mathbb{R}^{d \times r}$, and $A \in \mathbb{R}^{r \times k}$, with $r \ll \min(d, k)$. This decomposition helps maintain computational efficiency, avoiding extra latency during inference.

For our experiment, we needed to specify four parameters, namely, rank r, alpha α, dropout rate, and the layers to be tuned. The rank determines the number of columns in A and the number of rows in B. Basically, it decides how much information is captured by these smaller matrices. The rank controls the complexity and expressiveness of the adaptation. A higher rank means that the low-rank matrices can capture more information (up to the level of full fine-tuning), leading to a more flexible adaptation. However, this also means more parameters, which can increase computation and the risk of overfitting. The original paper [HSW+] and a follow-up paper conducted experiments

on InstructBLIP (BLIP architecture focused on visual reasoning tasks) with a maximum rank of 8. Due to our available computing resources (A100 GPUs with 80G), we decided to set the rank to 32.

We used the **rank-stabilized LoRA (rsLoRA)** method [Kal] to enhance the performance of our model adaptation. In the standard LoRA architecture, each adapter is scaled during every forward pass by a fixed scalar that is set during initialization and is dependent on the rank r. Typically, this scalar is defined as: $\frac{\texttt{lora\_alpha}}{r}$ in the original implementation. However, rsLoRA modifies it by using: $\frac{\texttt{lora\_alpha}}{\sqrt{r}}$. This adjustment helps stabilize the adapters, especially when using higher ranks and allows for more effective adaptation. By adopting the rsLoRA approach, we can get the benefits of higher rank values without encountering instability, resulting in improved performance and more reliable fine-tuning outcomes.

Alpha α is a scaling factor. By controlling the magnitude of α, you can scale the impact of the learned low-rank changes. A higher α means the adaptation has a stronger effect, while a lower α means it has a weaker effect. Based on the model used in Sungkyung et al. [KLP+], we decided to base our experiment setup on their work and use an alpha value of $\alpha = 2 \times$ rank and a dropout rate of 0.05 which is lower than the dropout value in the original LoRA paper.

Lastly, we decided to treat the layers as a hyperparameter to be tuned and included different settings: 1) we add weights to all linear layers (based on the idea of QLoRA [DPHZ], 2) the default LoRA settings which are to add trainable weights to the query and value layers of each attention block.

Using a rank of 32 for LoRA we see the resulting number of trainable parameters in Table 4.1 (retrieved by calling the `print_trainable_parameters()` of the `PeftModel` class for each model).

Table 4.1: Number of model parameters and trainable percentage and number after application of LoRA.

Model	All Params	Trainable Params	Trainable %
BLIP-2-2.7B	3,755,165,696	10,485,760	0.2792
BLIP-2-6.7B	7,769,515,520	16,777,216	0.2159
BLIP-2-XL	3,961,320,960	18,874,368	0.4765
BLIP-2-XXL	12,267,345,408	37,748,736	0.3077
LLaVA-7B	7,083,350,016	19,922,944	0.2813
LLaVA-13B	13,380,854,784	29,360,128	0.2194

4.2.2 Hyperparameter Tuning

Fine-tuning LLMs can be a resource-intensive procedure that (depending on many factors such as where the servers are located and which GPUs are used) can result in different CO_2 emissions. We run our experiments on an A100 SXM4 80GB on the GPU cluster provided by TU Wien. The maximum time the fine-tuning can run on this cluster is 168 hours. With an average Carbon Efficiency (kg/kWh) of 0.432 based on the OECD's 2014 yearly average, a full run emits 29.03 kg of CO_2 (based on [LLSD] and their website[2]).

We conducted a grid search to tune the hyperparameters, focusing on the learning rate, batch size, and LoRA layers. The learning rate was tested with values of {1e-5, 5e-5, 1e-4, 5e-4}, while the effective batch size was varied across $\{16, 32\}$. If the entire batch could not fit into GPU memory, we used gradient accumulation to adjust accordingly. For the LoRA layers, two configurations were explored: `all-linear`, where LoRA was applied to all linear layers, and `QV`, where only the query and value layers were adopted. This setup resulted in a total of 16 possible combinations (see Table 4.2).

Table 4.2: Hyperparameter tuning results with early stopping using the smallest BLIP-2 model with 2.7 billion parameters and the H&M dataset.

Best Loss	Runtime	Batch Size	Learning Rate	LoRA Layers	Early Stopping
0.028	39h 15m 10s	16	5e-5	all linear	No
0.031	36h 12m 50s	32	5e-5	all linear	Yes
0.034	39h 22m 57s	16	1e-5	all linear	No
0.036	39h 6m 43s	32	1e-5	all linear	No
0.038	27h 45m 35s	32	1e-4	all linear	Yes
0.045	25h 1m 58s	16	1e-4	QV layers	No
0.045	30h 51m 23s	32	1e-4	QV layers	No
0.046	24h 45m 38s	16	5e-5	QV layers	No
0.048	30h 38m 18s	32	5e-5	QV layers	No
0.057	24h 38m 44s	16	1e-5	QV layers	No
0.062	12h 41m 28s	32	5e-4	QV layers	Yes
0.062	24h 42m 5s	32	1e-5	QV layers	No
0.065	15h 58m 19s	16	1e-4	all linear	Yes
0.356	15h 25m 21s	16	5e-4	QV layers	Yes
0.366	23h 20m 53s	32	5e-4	all linear	Yes
0.816	16h 9m 19s	16	5e-4	all linear	No

[2] `https://mlco2.github.io/impact/`, last accessed 14.03.2025, 10:26

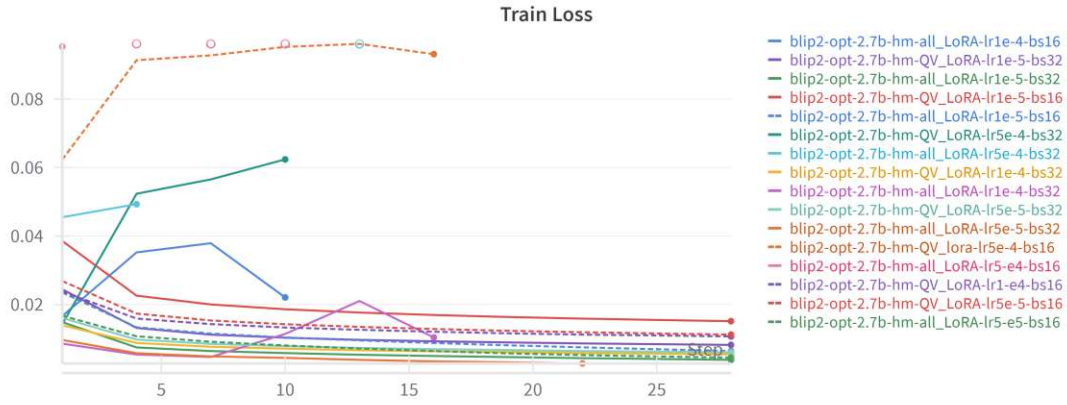
Fine-tuning 6 models across 16 different parameter settings and two datasets would have led to an estimated CO_2 emission of 5568 kg (assuming all 192 variations would run the full 168 hours). This is approximately equivalent to the annual electricity consumption of an average U.S. household (about 10,700 kWh, assuming 0.52 kg of CO_2 per kWh). Due to this environmental impact, we decided to run the parameter sweep using only the smallest model (BLIP-2 with 2.7 billion parameters) and the smaller H&M dataset. We then use the best hyperparameter setting with the lowest validation loss for all fine-tuning settings.

The fixed settings for the framework include the following hyperparameters. The learning rate is set to 5e-4, controlling the step size during optimization. A `CosineAnnealingLR` scheduler is used to adjust the learning rate over time, gradually decreasing it in a cosine pattern, with a minimum learning rate of 1e-6 and a `T_max` of 500 iterations, specifying the period during which the rate decreases. We deemed this scheduler the most suitable because it focuses on larger updates at the start and smaller, more precise updates as training progresses. It is particularly beneficial for tasks where gradual fine-tuning towards the end improves convergence and helps avoid overshooting. A weight decay of 1e-6 is also applied to regularize the model, preventing overfitting by penalizing large weights. As we are working with large datasets, we deem a relatively small value for weight decay suitable because the model can generalize better without needing as much regularization. However, this is a parameter that could also be optimized.

We encountered numerical instabilities during training resulting in NaN values for the loss computation. This issue was resolved using `bfloat16` as a floating-point format because it provides a greater range. We used the `bfloat16` for all models.

Analyzing the line charts of the train and validation loss progress of the 16 settings presented in Figure 4.5, we see certain settings that lead to instable training with either too high learning rates or batch sizes. We assume this is the case when using the learning rate 5e-4 for both tried batch sizes because they start with a relatively high loss but lead to NaN values later on (see Table 4.3). Those runs are marked by dots in the line plot. If the runs do not end with NaN values they do not converge e.g. the dotted orange line in the plots and end due to early stopping.

An interesting observation is that the learning rate 1e-4 leads to unstable training using all linear layers, whereas when fine-tuned using only query and value layers, the training converges and leads to smaller losses. This can be due to a high sensitivity to parameter changes when fine-tuning all layers, making the model prone to instability with higher learning rates. In contrast, focusing on only the query and value (QV) layers reduces trainable parameters, promoting stable adjustments even with relatively high learning rates. The best results in terms of validation loss (for BLIP-2 the loss represents the language modeling loss) are achieved using all linear layers and a smaller learning rater (1-e5 or 5-e5). However, the best model using only QV runs almost in half the time (39h vs. 25h) with the best validation loss being 60.7% worse but still reasonably low (0.028 vs. 0.045).



(a) Train loss for the 16 settings.

(b) Validation loss for the 16 settings.

Figure 4.5: Line plots of the train and validation loss (plotted using the weights and biases framework [Bie] supervising the runs). The Y axis shows the loss and the X axis shows the step. The model names include the hyperparameter used for the setting.

Summarizing the observations for the hyperparameter sweep:

- The learning rate of 5-e4 was too high and led to unstable training (independently from the batch size).
- The learning rate of 1-e4 is too high for using all linear layers to fine-tune.
- Tuning all linear layers led to better results than just tuning the QV layers because it resembles full fine-tuning more (see Figure 4.6).
- The batch size did not have as much impact as expected.

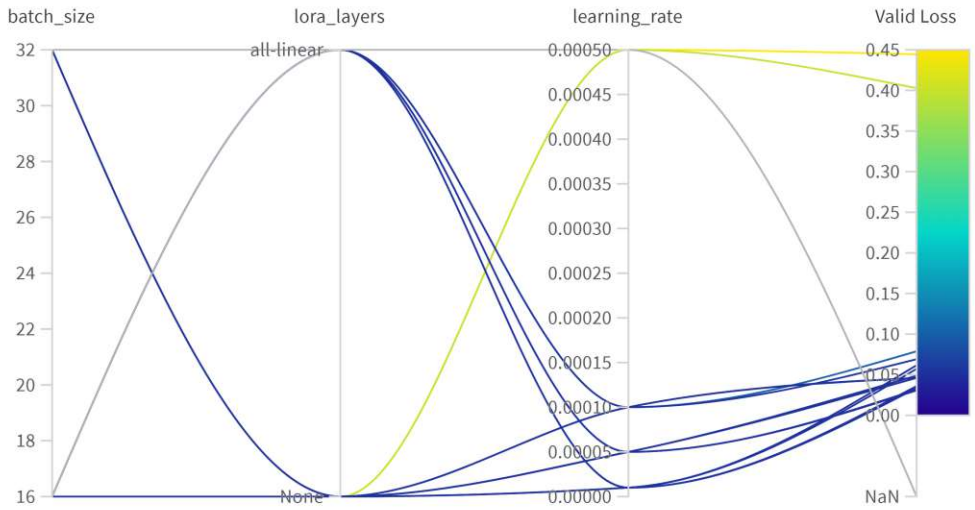


Figure 4.6: Parallel coordinates plotting the hyperparameters that were tuned, namely the batch size, LoRA layers, and learning rate. For the parameter LoRA layers value "None" is equivalent to choosing query and value layers.

Table 4.3: Fine-tuning results with the worst validation losses, train and valid loss showing the last loss recorded. All runs can be seen in Table 4.2.

Best Loss	Runtime	Batch Size	Learning Rate	LoRA Layers	Train Loss	Valid Loss
0.816	16h 10m 0s	16	5e-4	all-linear	NaN	NaN
0.366	23h 22m 2s	32	5e-4	all-linear	NaN	NaN
0.356	15h 25m 58s	16	5e-4	QV-layers	0.093	0.403
0.065	15h 59m 0s	16	1e-4	all-linear	0.022	0.079
0.062	24h 42m 38s	32	1e-5	QV-layers	0.008	0.062

In conclusion, we see that as long as the learning rate is small enough, the training converges, and higher learning rates can be used if the trainable parameters are reduced, e.g., using only QV layers. Considering that the validation loss is relatively similar using a validation set of 10423 items, we decided to prioritize the run time (also considering the previously mentioned CO_2 emissions) over the performance. Therefore, we select the best QV layer setting with learning rate 1e-4 and batch size 16 as the setting used to fine-tune the other models (see Table 4.2).

4.3 Evaluation

Additionally, for the metrics described in Section 2.1.3, we include the mean average precision and category accuracy as described by Yang et al. [YZJ+].

4.3.1 MAP

The authors describe the mean average precision as the following "we compare the attributes in the generated captions with those in the test set as ground truth to find the average precision rate for each attribute using mean average precision (MAP)" (Source: [YZJ+], page 11), which technically does not describe the mean average precisions usual definition. Based on the code from the Github repo, the authors compute the average precision, taking the precision for each caption and then dividing it by the number of captions. We keep the name for comparison.

As described previously in Chapter 3, the attributes for FACAD are provided and the attributes for H&M are generated based on the approach of Yang et al. [YZJ+]. For the generated captions, we follow the same procedure to identify positives: lower-case, split words connected by hyphens, extract nouns, adjectives, and proper nouns, then select attributes that appear in the pool of attributes of the train set. This then gives the attributes to be compared to the ground truth and computes the precision and recall, respectively.

At this point, we also mention that the authors Yang et al. do not provide the original attributes for the test set. In order to recreate the attributes, we use a file that was provided with all metadata (including attributes), check for overlapping captions, and then use the attributes for the corresponding item.

4.3.2 Accuracy

The authors originally pre-train a 3-layer text CNN [Kim], but the model is not provided nor can it be found through the sources linked by the authors. As a result, we used a pre-trained BERT model for text classification and fine-tuned it for the H&M and FACAD datasets, where captions are the input and the product category is the class. Both models use the same architecture (`bert-base-uncased`), but due to the difference in the dataset size, the training times vary.

For the FACAD dataset, the authors report a classification accuracy of 90% on the test set. Our model is trained for 5 epochs with a learning rate of 1-e5 and a batch size of 128, achieving a final accuracy of 90.3% on the test set, which consists of 78 classes. Therefore providing an equivalently good model for category classification.

For the H&M dataset, the model is trained for 15 epochs with a learning rate of 1-e5 and a batch size of 128, achieving a final accuracy of 94.7% on the test set, which contains 89 classes.

To evaluate our fine-tuned models, we use the classifier models in combination with the generated captions as inputs and report the accuracy for category classification.

It should be mentioned that the majority of captions for both datasets include the product category, e.g., "a crepe blouse" and the models then predict blouse as the category. This metric can then be interpreted as how well models are able to identify the correct product category.

4.4 Recommendations

In Section 3.2, we describe the processing and splitting of the data that we use for the recommendation experiment of this thesis. We also presented the metrics that we will use for evaluation in Section 2.2.1. Finally in this section, we will shortly review the pipelines used for feature extraction as well as the algorithms used for recommendation. To conduct the recommendation experiment, we use the Ducho-meets-Elliot framework presented by Attimonelli et al. [ADF+]. It provides feature extraction as well as a recommendation pipeline.

4.4.1 Feature Extraction

The framework provides an integration of Ducho [ADM+] for feature extraction. For our experiments, we will explore 6 different setups:

- Visual features extracted from ResNet50
- Textual features extracted from SentenceBert
- Multimodal trained visual and textual features extracted from CLIP
- Multimodal features concatenating ResNet50 and SentenceBert features
- Features based on the output of the Q-Former from the fine-tuned BLIP-2 model
- Visual features extracted from the fine-tuned BLIP-2 model

ResNet50

ResNet, short for Residual Network, is a specialized type of convolutional neural network (CNN) designed to address the vanishing gradient problem by using residual connections. ResNet-50 is a deep architecture consisting of 50 layers, including 48 convolutional layers, one max pooling layer, and one average pooling layer. These networks are built by stacking residual blocks, which help maintain stable gradients during training. We extract visual features using a pre-trained ResNet 50[3], with the average pooling layer as the extraction layer, followed by z-score normalization, which standardizes the feature values by subtracting the mean and dividing by the standard deviation.

SentenceBert

The model we use for text embedding of the descriptions is a fine-tuned version of `microsoft/mpnet-base`[4], trained using a contrastive learning objective on a dataset

[3] `https://pytorch.org/vision/main/models/generated/torchvision.models.resnet50.html`, last accessed 13.02.2025, 13:45

[4] `https://HuggingFace.co/sentence-transformers/all-mpnet-base-v2`, last accessed 13.02.2025, 13:45

containing 1 billion sentence pairs. The objective requires the model, given one sentence from a pair, to identify its correct counterpart among a set of randomly sampled sentences. It was developed during the Hugging Face Community Week on JAX/Flax for NLP & CV as part of the project *Train the Best Sentence Embedding Model Ever with 1B Training Pairs*, leveraging TPU v3-8 hardware and benefiting from guidance provided by Google's Flax, JAX, and Cloud teams. The model is designed as a `sentence-transformers` model, mapping sentences and short paragraphs to a 768-dimensional dense vector space. These embeddings capture semantic meaning and can be used for tasks such as information retrieval, clustering, and semantic search. Input sequences exceeding 384 word pieces are truncated. Fine-tuning involved computing cosine similarities between sentence pairs in a batch and applying a cross-entropy loss. The training process consisted of 100k steps with a batch size of 1024 (128 per TPU core), utilizing the AdamW optimizer with a learning rate of 2e-5 and a warm-up phase of 500 steps. The sequence length was limited to 128 tokens. The training data consisted of a mixture of multiple datasets, sampled based on weighted probabilities.

CLIP

We used CLIP (Contrastive Language-Image Pre-training)[5], a multimodal neural network developed by OpenAI that learns to understand images and text together. It is trained using a contrastive learning approach, where a large dataset of image-text pairs is used to align visual and textual representations in a shared embedding space. The model consists of two separate encoders: one for images (typically a ResNet or Vision Transformer, in our case ViT-B/16 Transformer) and one for text (a Transformer-based language model). During training, CLIP learns by maximizing the similarity between correct image-text pairs while minimizing the similarity between incorrect ones. This results in a unified embedding space where semantically related images and texts are mapped close to each other.

BLIP-2

A detailed explanation of BLIP-2 was provided in Section 4.1.1. For feature extraction, we decided to "fully" fine-tune the smallest BLIP-2 models (BLIP-2-2.7B) using all linear layers and a learning rate 5e-5 (based on our hyperparameter tuning experiment, Section 4.2.2). The model used was the one that achieved the best result on our H&M test image captioning set and was trained 8 epochs with a batch size of 8. We then use the last hidden layers from the Q-Former as Queries features (32x768 values) and the last hidden layers from the fine-tuned image encoder as visual features (257x1408).

Using the flattened visual features (381,856 values) for over 100K items required over 700GB of GPU memory, making it impractical to process with our biggest GPU having

[5] `https://HuggingFace.co/openai/clip-vit-base-patch16`, last accessed 13.02.2025, 13:50

80GB. To address this issue, two different dimensionality reduction techniques were explored: (1) Principal Component Analysis (PCA) and (2) Global Average Pooling.

PCA provides an effective way to retain the most informative structures of the data while significantly reducing its dimensionality. Given the large size of the dataset, Incremental Principal Component Analysis (Incremental PCA) was used, as it allows processing data in smaller batches without exceeding memory constraints. To determine the optimal number of components, an initial analysis was performed on a subset of the data, selecting the smallest number of components that retained 95% of the total variance.

Once the appropriate dimensionality was established, the full dataset was processed iteratively using Incremental PCA. The model was trained in batches, gradually learning the lower-dimensional representation of the feature vectors. Finally, the trained PCA model was used to transform the original high-dimensional feature vectors into their compressed representations, making them significantly more manageable while preserving their essential structure.

Due to the size of the dataset and the high dimensionality of the feature vectors, we employed Incremental Principal Component Analysis (Incremental PCA) for dimensionality reduction. Unlike standard PCA, which requires loading the entire dataset into memory, Incremental PCA processes data in mini-batches, making it well-suited for large-scale datasets.

To determine the optimal number of components, we analyzed a representative subset of 5,000 vectors, selecting the number of components that retained 95% of the variance. This resulted in 201 principal components, significantly reducing the original vector size from 361,856 dimensions to 201. For average pooling, we reduced the dimensionality to 1,408 by averaging across the 257 values.

When comparing the results (see Table 4.4), PCA features performed better, though the differences are small.

Table 4.4: Recommendation results for H&M dataset, using PCA and average pooled visual features, on the test set explained in Section 2.2.1.

Setting	**NDCG**		**MAP**	
	@5	**@12**	**@5**	**@12**
VBPR (PCA)	0.02187	0.02656	0.01110	0.00853
VBPR (avgpool)	0.02167	0.02631	0.01092	0.00840

4.4.2 Recommendation Algorithms

We are limited by the algorithms supported by the elliot pipeline [ABF+]. We choose to use Visual Bayesian Personalized Ranking (more details presented in Section 2.2) as

it supports using different features for recommendation and implicit feedback. For all algorithms, we use the default settings from elliot if not stated otherwise.

Unpersonalized Algorithms

As a baseline, we use unpersonalized recommendation algorithms that suggest items without considering individual user preferences. Most Popular (MostPop) recommendations show the items that are most purchased by all users, such as bestsellers or trending products. This approach is simple and works well when user tastes are similar. Random recommendations, on the other hand, select items by chance, offering more variety but often showing irrelevant suggestions. These algorithms help us compare and evaluate the performance of our recommendations using the extracted features.

Neighborhood-based Algorithms

We use Amazon's item-to-item collaborative filtering (ItemKNN) [LSY] as a recommendation approach that suggests products by analyzing the relationships between items based on user interactions. This method finds items similar to those a user has previously interacted with, using factors like co-purchasing behavior and browsing patterns. Compared to user-based collaborative filtering, it relies on item similarities rather than user similarities, making it a better choice for large-scale platforms such as Amazon.

Table 4.5: Recommendation results for H&M dataset, using different neighborhoods for ItemKNN, on the test set explained in Section 2.2.1.

Setting	**NDCG**		**MAP**	
	@5	**@12**	**@5**	**@12**
ItemKNN (k=20)	0.05760	0.06421	0.03557	0.02534
ItemKNN (k=40)	0.05743	0.06409	0.03558	0.02533

As UserKNN, we refer to the algorithm used by GroupLens [RIS+], an open architecture for collaborative filtering of netnews. The GroupLens system uses collaborative filtering to recommend news articles to users based on the preferences of similar users. By analyzing patterns in users' interactions with content, GroupLens was one of the first systems to demonstrate the power of collaborative filtering in a real-world setting. If using implicit feedback (e.g., clicks, purchases), the algorithm recommends items most frequently interacted with by the k-nearest neighbors.

For both algorithms, we use cosine similarity and the `aiolli` implementation setting for implicit feedback.

Table 4.6: Recommendation results for H&M dataset, using different neighborhoods for UserKNN, on the test set explained in Section 2.2.1.

Setting	NDCG		MAP	
	@5	@12	@5	@12
UserKNN (k=5)	0.00767	0.00880	0.00359	0.00246
UserKNN (k=10)	0.01198	0.01355	0.00558	0.02533

Visual Bayesian Personalized Ranking from Implicit Feedback

Visual Bayesian Personalized Ranking (VBPR) [HMb] extends traditional Matrix Factorization (MF) models by integrating visual features to improve recommendation accuracy. Unlike conventional approaches that are solely based on user-item interactions, VBPR introduces an additional layer of visual embeddings extracted from product images, allowing the model to better capture the impact of an item's appearance on user preferences. The model is trained using Bayesian Personalized Ranking (BPR), a pairwise ranking optimization framework that employs stochastic gradient ascent to learn personalized rankings from implicit feedback data such as purchases or clicks.

The core mathematical formulation of VBPR is given by:

$$\hat{x}_{u,i} = \alpha + \beta_u + \beta_i + \gamma_u^T \gamma_i + \theta_u^T (E f_i) + \beta'^T f_i$$

where:

- α is the global bias term.
- β_u, β_i are the user and item biases.
- γ_u, γ_i are the latent factors representing the user u and item i.
- θ_u, θ_i are the visual preference vectors for the user and item.
- f_i represents the deep CNN-extracted visual features of item i.
- E is the embedding matrix that transforms f_i into a lower-dimensional space.
- β' is a visual bias vector that captures the general influence of item appearance.

While the study primarily relies on CNN-based feature extraction, the VBPR framework is highly flexible and can accommodate alternative feature vectors beyond deep visual models. Instead of using visual embeddings alone, it is possible to incorporate textual embeddings derived from product descriptions and reviews, handcrafted features such as color and

material, or multimodal representations that combine various sources of information. This adaptability enables VBPR to be extended to a variety of recommendation scenarios beyond strictly visual domains, making it perfectly suited for our experiment.

CHAPTER 5

Results & Discussion

5.1 Fashion Captioning

5.1.1 Setting

We followed the procedure with the setting described in Section 4.2.2. We then proceed to save the model with the lowest validation loss and use early stopping with patience of 3, meaning that if there is no improvement on the validation set after 3 epochs, we stop training. This setup with the different model and dataset sizes and a limitation of 168 hours (a week) on the cluster leads to the checkpoints displayed in Table 5.1. The FACAD dataset includes a train set size of 888,293 and a validation set size of 19,915. The H&M dataset includes 83,385 samples for training and 10,423 samples for validation. The epochs presented in Table 5.1 show the checkpoints later used for inference.

5.1.2 Quantitative Analysis

The quantitative analysis of the captions includes the metrics presented in Section 2.1.3 and Section 4.3. We present the results of the pre-trained models and the fine-tuned models side-by-side for the H&M dataset in Table 5.4 and for the FACAD dataset in Table 5.5.

H&M. One notices that performance improved for all models except for the BLIP-2 models that are based on the FLAN-T5 language models. These models could only complete 3 or 6 epochs, and it appears that the fine-tuning process introduced more noise than targeted specialization. However, all other models that trained at least 8 epochs show improvements across multiple evaluation metrics. Among them, BLIP-2-6.7B achieves the best performance making it the strongest candidate for fashion captioning on the H&M dataset. One interesting fact that we observed and that we also investigated during our qualitative analysis (see Section 5.1.3) is the trade-off between the precision

Table 5.1: Training and validation times for the different model checkpoints on H&M and FACAD datasets. The epochs stating the epoch the model was saved that was later used for inference. Time per epochs shows the approximate time for a training/validation epoch using a A100 with 80GB.

Model	H&M		FACAD	
	Epochs	Time Per Epoch	Epochs	Time Per Epoch
BLIP-2-2.7B	8	2.25h train / 10min val	5	16h train / 10min val
BLIP-2-6.7B	8	3.25h train / 15min val	3	27h train / 16min val
BLIP-2-XL	6	3.5h train / 15min val	2	29h train / 17min val
BLIP-2-XXL	3	7.75h train / 26min val	1	77h train / 41min val
LLaVA-1.5-7B	8	4.16h train / 17min val	1	35.5h train / 20min val
LLaVA-1.5-13B	8	7.5h train / 25min val	1	70h train / 33min val

Table 5.2: Fashion captioning results for dataset H&M in pre-trained (PT) and fine-tuned (FT) scenarios. The best results are highlighted in bold. *CIDEr shows results beyond 100 due to multiple scaling (see Section 2.1.3).

H&M														
Model	BLEU-4		METEOR		ROUGE-L		CIDEr		SPICE		MAP/MAR		Acc	
	PT	FT	PT	FT	PT	FT	PT	FT	PT	FT	PT	FT	PT	FT
BLIP-2-2.7B	0.3	40.8	5.3	33.5	14.8	63.4	7.0	275.2*	8.0	44.7	37.2 / 11.4	70.8 / 65.4	53.0	83.5
BLIP-2-6.7B	0.3	**41.4**	5.5	**33.9**	15.3	**63.8**	7.3	**281.3***	8.3	**45.4**	38.3 / 11.6	**70.8** / 66.0	53.8	**83.8**
BLIP-2-XL	0.3	0.2	5.1	4.4	15.0	10.6	6.3	6.6	8.0	7.3	38.9 / 10.8	37.0 / 10.3	56.5	51.7
BLIP-2-XXL	0.3	0.4	5.3	5.6	15.5	16.6	7.0	7.2	8.4	8.7	38.0 / 11.2	38.2 / 11.9	58.0	56.0
LLaVA-1.5-7B	0.5	23.4	7.4	32.8	14.5	46.0	0.8	24.2	5.3	34.6	15.2 / 13.0	53.1 / **71.3**	32.0	82.9
LLaVA-1.5-13B	0.4	22.9	7.4	32.3	14.2	45.5	1.2	22.0	5.8	34.9	17.8 / 13.8	54.0 / 70.8	31.0	83.5

and recall of the attributes for the different models. Due to the longer captions produced by the LLaVA models, they achieve higher recall scores but then lack precision. The opposite for the BLIP-2 models. They achieve better overall performance by having a balance between precision and recall.

FACAD. Comparing the accuracy reported by Yang et al. for the category classification, we noticed that our models (fine-tuned BERT classifier) worked better than the reported CNN classifier. One should note that the accuracy metric generally describes how well the models recognize the correct category from the image because as long as the generated caption includes a category, e.g., t-shirt or top, the accuracy models will predict the category based on the caption input (with an accuracy of >90%, based on the performance on the testset). Given these considerations, we cannot fully explain why the zero-shot setup (using BLIP-2 results) achieves nearly 40% better accuracy than the results reported by Yang et al. We hypothesize that this improvement may be due to

the BERT model outperforming the originally used CNN-based model or because SRFC struggles to accurately classify clothing items.

Also here, similar to the H&M results previously mentioned, the larger BLIP-2 models (BLIP-2-XL and BLIP-2-XXL) could not complete 1 or 2 epochs due to the size of the models, the size of the dataset, and the limitation of time on the cluster. Besides those, we see an improvement for BLIP-2-2.7B, showing similar performance to the model by Yang et al. and improved MAP results. For the LLaVA models, we see the same behavior as previously described for H&M with producing longer captions leading to higher recalls and lower precision.

Table 5.3: Fashion captioning results for the FACAD dataset in pre-trained (PT) and fine-tuned (FT) scenarios. The last two models show the worst (CNN-C) and best (SRFC) models reported by Yang et al., the best model being the one presented by the authors. The best values for each metric are highlighted in bold.

FACAD														
Model	**BLEU-4**		**METEOR**		**ROUGE-L**		**CIDEr**		**SPICE**		**MAP/MAR**		**Acc**	
	PT	**FT**	**PT**	**FT**	**PT**	**FT**	**PT**	**FT**	**PT**	**FT**	**PT**	**FT**	**PT**	**FT**
BLIP-2-2.7B	0.3	3.7	4.6	10.1	12.6	19.7	4.3	36.4	6.4	10.3	17.4 / 7.9	**24.6** / 22.5	52.0	**69.9**
BLIP-2-6.7B	0.3	3.5	4.6	9.8	12.9	19.1	4.1	34.4	6.4	9.8	17.7 / 7.8	23.3 / 21.2	51.4	69.0
BLIP-2-XL	0.2	0.1	4.2	3.3	12.9	8.2	3.2	2.3	6.4	5.8	17.6 / 7.2	18.5 / 6.3	52.2	45.7
BLIP-2-XXL	0.2	0.1	4.2	3.3	12.8	9.2	3.1	2.3	6.4	5.4	18.4 / 7.4	17.7 / 5.5	53.1	41.6
LLaVA-1.5-7B	0.2	1.5	6.8	11.2	12.0	15.4	0.1	1.2	4.1	7.4	9.8 / 8.7	14.7 / **27.9**	45.1	65.5
LLaVA-1.5-13B	0.2	0.7	6.5	8.3	12.4	12.6	0.8	0.1	4.5	3.7	9.9 / 8.5	8.4 / 16.3	44.2	27.4
CNN-C [ADS]	2.1		7.2		16.3		20.8		6.5		4.9 / -		10.8	
SRFC [YZJ+]	**6.8**		**13.2**		**24.2**		**42.1**		**13.4**		9.5 / -		18.2	

5.1.3 Qualitative Analysis

For the qualitative analysis, we created an interface that displays each product image given as input to the models, the captions produced, and their scores as well as the categories classified (see Figure 5.1). We then take a closer look into the captions, the zero-shot, and fine-tuned ones, to identify edge cases, patterns, impact on certain performance metrics, and more. This is done for the first 20 samples in the H&M dataset and the first 3 distinct items in the FACAD dataset (because the dataset includes the same caption for multiple images entailing the same item).

We exclude the CIDEr score in the qualitative analysis because comparing two captions is always zero. This is because for each n-gram in the generated caption (test caption), the TF-IDF weight is computed using the formula:

$$\text{TF-IDF} = \text{TF} \times (\log(\text{Number of References}) - \log(\text{DF}))$$

where TF is the term frequency (frequency of the n-gram in the test caption), DF is the document frequency (number of reference sentences containing the n-gram), and

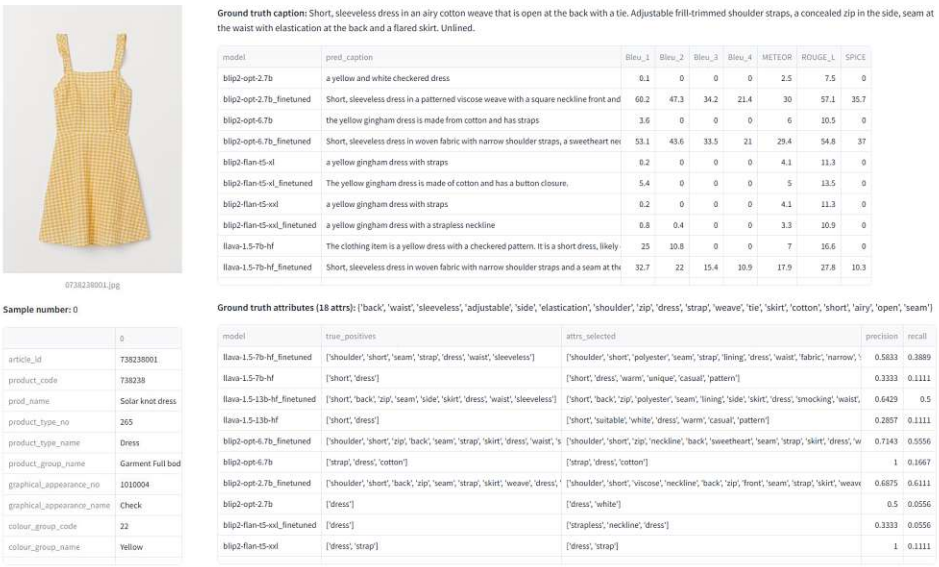


Figure 5.1: Example (using H&M item) of the information displayed for qualitative analysis. It includes captions, scores (except for CIDEr), attributes, and the respective recall and precision. For H&M, we additionally display the information that was provided dataset. We created this interface using the Streamlit library for Python.

Number of References is the total number of reference sentences. Since there is only one reference sentence in this case, $\log(\text{Number of References}) - \log(\text{DF}) = \log(1) - \log(1) = 0$, which makes the IDF term equal to 0. As a result, the TF-IDF weight for every n-gram in the test caption becomes $\text{TF-IDF} = \text{TF} \times 0 = 0$, because the dot product of a zero vector with any other vector is 0.

H&M. For the particular sample seen in Table 5.4, we see that the fine-tuned LLaVA models manage to include most information of the original caption. After reaching the end of the original caption length, the model starts to "hallucinate" polyester material and funding. This might be due to the high presence of polyester in the train captions (2269 samples), but there is no "European Regional Development Fund" in the train data. Looking at the different metrics we noticed the highest BLEU score does not always indicate that the other metrics are also the highest compared between all models. We saw in Section 2.1.3 that BLEU includes a brevity penalty, METEOR includes a fragmentation penalty, ROUGE-L focuses on the longest, most common sequence, and SPICE focuses on the semantic overlap. The example shows well how fine-tuning adjusted the model's expression, wording, and sentence structure (punctuation and capitalizing). It also shows that the models were untrained to include the color and pattern due to H&M captions not including either. The BLIP-2 FLAN-T5 models (due to the short fine-tuning process) only improved slightly.

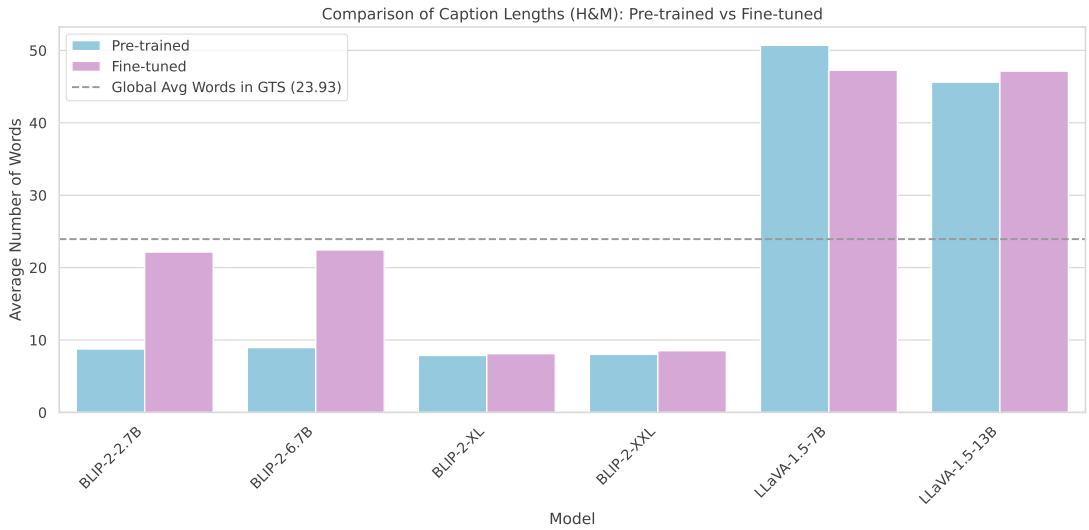


Figure 5.2: Comparison of length of captions before and after fine-tuning, the horizontal line showing the average number of words in the captions of the H&M test set.

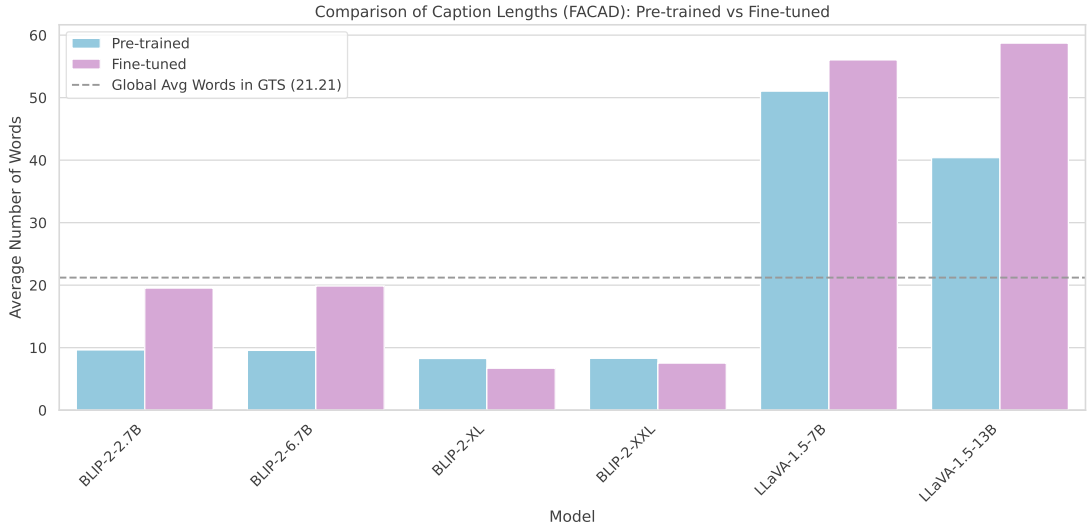


Figure 5.3: Comparison of length of captions before and after fine-tuning, the horizontal line showing the average number of words in the captions of the FACAD test set.

For the other samples, we noticed that due to the fact that the LLaVA models are set with `max_new_tokens` to 256 (required parameter in the HuggingFace interface) for text generation, they overgenerate and fill up the missing space between the ground truth length and generation length. The models stubbornly generate the given length without considering the end of a sentence (see Table 5.4, fine-tuned LLaVA models). This, however, does not happen with the BLIP-2 models. We can see in Figure 5.2 that when comparing the average length of the generated captions before and after fine-tuning the bar plot shows that the longer fine-tuned BLIP-2 models learn the length of the captions.

The longer captions of the LLaVA models lead to a higher recall for the extracted attributes but lower precision and the opposite for the BLIP-2 models. Except when there is a small number of attributes (due to shorter descriptions usually), then BLIP-2 also achieves good results for recall.

Between BLIP-2 with 2.7B and 6.7B parameters, we notice that BLIP-6.7B (coherent with the results in Table 5.2) shows better performance by including more detail about the items. We assume this is due to the larger number of parameters.

We see that the limitations of the models are captioning the material, e.g., satin vs. velvet vs. corduroy, or textile, e.g., viscose and polyester, and the size of the products, e.g., 32x32cm. Also, products for which the product images are not identifying enough, e.g., showing a plain fabric cloth which is a handkerchief but could also be a scarf or a snippet of a bigger item.

FACAD. For the sample shown in Figure 3.5a, we noticed that the part of the caption including “inspired by the work of rising spanish photographer coco capit n” has no visual connection to the image. Samples like this can introduce noise to a model trying to learn the connection between words and visual features. Comparing the fine-tuned LLaVA models between both datasets, we see that they took over a “selling” type of expression, e.g., “a timeless appeal that never goes out of style”.

With regard to caption comparison between the two models, we see a similar behavior as to the H&M dataset in regards to caption length (see Figure 5.3). We notice that LLaVA achieves good results for creating “sales” like phrases. However, the models overdo it for simple item descriptions (see example in Table 5.5).

For LLaVA, some captions cycled back to producing the prompt template that was used to generate the caption, providing answers (example below). The models’ responses begin with `ASSISTANT:` and then regenerate a response.

```
USER:
Describe the appearance of the clothing item.

ASSISTANT:
A classic tailored jacket is updated in a fresh floral print
 ↪  and cut from a lightweight woven fabric that's a fresh,
 ↪  breezy option for warm weather day and night ahead of the
 ↪  season.

ASSISTANT:
A fresh floral print blossoms on a lightweight blazer that's a
 ↪  polished standout.
```

When analyzing the category prediction for this sample, we noticed that the models could not identify the category blazer and miscategorize it as a jacket. And for the material image and back image it miscategorizes it as tee/top.

We noticed that some of the attributes extracted and included by the authors [YZJ+] do not seem reasonable as they do not represent visual attributes of the items e.g. “chrissy”, “teigen” from “designed in partnership with chrissy teigen” or “work” from the sample presented in Figure 3.5a. One would assume attributes to be related the visual appearance of a clothing item.

Generally, we noticed a problem with partially visible clothing items. If the image does not show the item fully, e.g. see Figure 3.5a material shot or detail shot (middle image), the models have a problem identifying the correct category. Same for not visible details, e.g., open at the back with a tie or double back vent.

It is difficult for the captioning models to identify nuances of material, e.g., satin vs. velvet, color dark blue vs. black, or clothing type jacket vs. blazer. We think that this can be due to the resizing of the image for the image encoder, where details get lost, and the depiction of the items in the dataset.

5.1.4 Summary

To evaluate the impact of fine-tuning, we analyze how well it improved our pre-trained models, addressing RQ1 (Section 1.3). The results, presented in Table 5.6, show the average percentage change in performance—whether an increase or decrease—per model and dataset based on the tables presented in Section 5.1.2.

We observe that fine-tuning was more effective for the H&M dataset, which we attribute to the presence of a one-to-one relationship between images and captions. To further investigate this hypothesis, one could subsample the FACAD dataset to include only one-to-one samples, though we leave this for future work.

Additionally, we note that insufficient fine-tuning led to a performance decrease for BLIP-2-XL and BLIP-2-XXL. Finally, our results indicate that fine-tuning was generally more beneficial for BLIP-2 models compared to LLaVA models.

Comparing the results between SRFC, the model that was specifically trained for Fashion Captioning by Yang et al., and the results for our fine-tuned models, we notice that even though the image captioning metrics are "worse", the fine-tuned models manage to generate more accurate captions in terms of attributes and categories. We would, therefore, argue that our method provides the advantage of simplicity due to the avoidance of additional rewards while still keeping good results, focusing on attributes and semantics, and having the same amount of training time.

We, additionally, were able to present the limitations of fine-tuned models through a qualitative analysis presented in Section 5.1.3 where we showed that the models struggle to connect caption parts that have no visual representation because they are either not visible in the image e.g. backside or are abstract information e.g. recycled material or designed by a celebrity. For future work that relates to this task, we, therefore, recommend captions that describe visible parts of the clothing item and avoid multiple images with the same caption.

5.2 Recommendations

As explained in the Section 3.2.2, we use a temporal split and no cross-validation. The results reported for VBPR are the results for the best model based on the validation set. The VBPR model was trained for 200 epochs with a learning rate of 1e-5. We combined multiple feature vectors, e.g., visual and textual, by concatenation. The number of latent factors was set to 64.

With the results presented in Table 5.7 we are able to answer RQ2 (Section 1.3). Based on the H&M dataset the answer is textual embeddings. However, comparing the results, we see that all different feature spaces perform similarly. and not as good as the best overall result using the ItemKNN algorithm. This might also indicate that VBPR might not be the right algorithm for this dataset in general.

Table 5.4: Descriptions generated by different models for the image seen in Figure 3.5b. The ground truth caption is “Short, sleeveless dress in an airy cotton weave that is open at the back with a tie. Adjustable frill-trimmed shoulder straps, a concealed zip in the side, seam at the waist with elastication at the back and a flared skirt. Unlined.”.

Model	Description
BLIP-2-2.7B	a yellow and white checkered dress
BLIP-2-2.7B_finetuned	Short, sleeveless dress in a patterned viscose weave with a square neckline front and back, narrow shoulder straps and a concealed zip at the back. Seam at the waist and a gently flared skirt. Unlined.
BLIP-2-6.7B	the yellow gingham dress is made from cotton and has straps
BLIP-2-6.7B_finetuned	Short, sleeveless dress in woven fabric with narrow shoulder straps, a sweetheart neckline and a concealed zip at the back. Seam at the waist and a gently flared skirt. Unlined.
BLIP-2-XL	a yellow gingham dress with straps
BLIP-2-XL_finetuned	The yellow gingham dress is made of cotton and has a button closure.
BLIP-2-XXL	a yellow gingham dress with straps
BLIP-2-XXL_finetuned	a yellow gingham dress with a strapless neckline
LLaVA-1.5-7B	The clothing item is a yellow dress with a checkered pattern. It is a short dress, likely designed for warm weather or casual occasions. The dress features a yellow color and a checkered pattern, which adds a unique and eye-catching touch to the outfit.
LLaVA-1.5-7B_finetuned	Short, sleeveless dress in woven fabric with narrow shoulder straps and a seam at the waist. Unlined. The polyester content of the dress is partly recycled. The lining is made from recycled polyester. The production of this dress is partly funded by the European Regional Development Fund. The polyester content of the dress is rec
LLaVA-1.5-13B	The clothing item is a yellow dress with a checkered pattern. It is a short dress, suitable for warm weather or casual occasions. The dress is hanging against a white background, which highlights its vibrant color and pattern.
LLaVA-1.5-13B_finetuned	Short, sleeveless dress in woven fabric with smocking at the back, a concealed zip in one side, seam at the waist, and flared skirt. Unlined. The polyester content of the dress is partly recycled. The polyester content of the lining is recycled. The polyester content of the lining is rec

Table 5.5: Descriptions generated by different models for the first product image (left top corner) seen in Figure 3.5a). The ground truth caption is “this shawl collar jersey blazer bloom with black and white floral patterning inspired by the work of rising spanish photographer coco capit n”.

Model	Description
BLIP-2-2.7B	the black and white floral print jacket is a great option for women
BLIP-2-2.7B_finetuned	a drapey shawl collar and a waist defining tie add polished finishing touch to this versatile open front cardigan
BLIP-2-6.7B	the women’s jacket is made from a floral print.
BLIP-2-6.7B_finetuned	a lightweight jacket with a clean modern look is cut from a stretch knit with a subtle pattern and a relaxed fit
BLIP-2-XL	a woman wearing a black and white floral print jacket
BLIP-2-XL_finetuned	the floral printed jacket jacket
BLIP-2-XXL	a woman wearing a black and white floral jacket
BLIP-2-XXL_finetuned	the blazer is made of cotton and silk
LLaVA-1.5-7B	The clothing item in the image is a black and white jacket, which is worn by a woman. The jacket features a floral pattern, adding a touch of elegance and style to the outfit. The woman is standing in front of a white background, which helps to emphasize the contrast between the black and white jacket and the rest of the scene.
LLaVA-1.5-7B_finetuned	a classic blazer is updated in a boxy cropped cut of lightly textured crepe with a pretty floral print to add a touch of romance to your ootd game plan a well a a softening effect on your silhouette a flattering tie belt cinch the look a well a your new favorite work to weekend go to this season is
LLaVA-1.5-13B	The clothing item is a black and white jacket, which appears to be a cardigan or a blazer. The woman is wearing it over a black shirt, and she is standing in front of a white background. The jacket has a floral pattern, adding a touch of elegance and style to the outfit.
LLaVA-1.5-13B_finetuned	a classic cable knit cardigan is made extra cozy with a generous shawl collar and soft faux shearling lining the hood and hem to keep out the cold chill of winter day and night alike a perfect top to top into your cold weather look this cardigan ha a timeless appeal that never go out of style so it s one you

Table 5.6: Percentage of relative improvement (or decrease) of average improvement (or decrease) overall metrics per model and dataset.

Model	H&M % Improvement	FACAD % Improvement
BLIP-2-2.7B	2685.50	313.21
BLIP-2-6.7B	2696.34	297.97
BLIP-2-XL	-13.39	-21.81
BLIP-2-XXL	7.09	-23.78
LLaVA-1.5-7B	1289.53	288.39
LLaVA-1.5-13B	1255.67	17.27

Table 5.7: Recommendation results for H&M dataset on the test set explained in Section 3.2.2. The best results are highlighted in bold. We highlight the best results overall (using ItemKNN) and for the different feature spaces (using textual features).

Setting	NDCG		MAP	
	@5	@12	@5	@12
Random	0.00005	0.00007	0.00005	0.00004
MostPop	0.00538	0.00640	0.00439	0.00373
UserKNN (k=10)	0.01198	0.01355	0.00558	0.00379
ItemKNN (k=20)	**0.05760**	**0.06421**	**0.03557**	**0.02534**
Visual (ResNet50)	0.02414	0.02882	0.01213	0.00917
Visual (BLIP-2)	0.02187	0.02656	0.01110	0.00853
Textual (SentenceBERT)	**0.02509**	**0.02986**	**0.01270**	**0.00955**
Multimodal (ResNet50+BERT)	0.02427	0.02892	0.01220	0.00922
Multimodal (CLIP)	0.02455	0.02917	0.01241	0.00934
Queries (BLIP-2)	0.02451	0.02912	0.01247	0.00939

CHAPTER 6

Conclusion

This chapter provides a summary of the work done in this master's thesis and its contributions. We also discuss different aspects of the experiment highlighting limitations and other interesting insights. Finally, we present opportunities for future work based on the experiment results.

6.1 Summary

This thesis consists of mainly two parts: 1) experiments fine-tuning pre-trained image captioning models on domain-specific fashion datasets and 2) evaluating different feature spaces used for fashion recommendation and determining which is the best one using the H&M dataset. The work was done to answer the research questions presented in Section 1.3.

To answer RQ1, we first had to determine suitable models for our experiment setup. After reviewing different options, we decided to use BLIP-2 (Section 4.1.1) and LLaVA (Section 4.1.2). The main reasons were that they are easy to use with Hugging Face, they are up-to-date, and they work well for tasks that combine images and text. Also, because they are available on Hugging Face, we can easily change and fine-tune them. We were able to use 4 different variants of BLIP-2 and 2 different variants of LLaVA for our experiments.

We then continued to preprocess the H&M data based on the data processing of Yang et al. [YZJ+] to create a fashion captioning dataset. The detailed process is explained in Section 3.2.

In its first iteration, we explore the performance of the models off-the-shelf without fine-tuning to give a baseline of the performance. The results can be seen in Table 5.5 for FACAD and Table 5.4 for H&M and can be described as underwhelming. We then proceeded to fine-tune the models using LoRA, but in order to determine the right

setup, we conducted a small hyperparameter tuning based on the smallest BLIP-2 model (Section 4.2.2) and the smaller H&M dataset for a faster iteration. Based on those results, we then fine-tuned all 6 models on both datasets, resulting in 12 fine-tuned models. Additionally, we fine-tuned 2 BERT-based text classification models to classify captions to their right product type category.

After fine-tuning, we conducted a qualitative analysis (Section 5.1.3) of the results to get deeper insights into the understanding of the model. We observed that BLIP-2 adapted almost perfectly to the captions of the H&M dataset. For example, it unlearned to include the color of the item, since this detail is not included in the ground truth—unlike in standard image captioning tasks, where including color would be desirable. We noticed that the LLaVA models tend to overgenerate captioning or lack the capability of adjusting the length of their output to the ground truth. This leads to higher recall but lower precision. For the FACAD dataset, we show (Table 5.3) that our fine-tuned models achieve similar performance as to the model presented by Yang et al. without having to express specific rewards for semantic and using reinforcement learning. These insights and more provided answers to the RQ1 (Section 5.1).

We then proceed to use the entire H&M dataset, including transaction information, for recommendations. We extract features from our fine-tuned models because, based on our hypothesis, these embeddings should be targeted to prioritize the connection between image and text specifically for fashion items, e.g., pockets, buttons, sleeves, etc. For comparison, we also use other models for feature extraction (more details are presented in Section 4.4.1).

We use VBPR [HMb] which was initially designed to use CNN-extracted visual features but can be extended to use other feature vectors. With the help of the elliot framework [ABF+], we were able to easily run different setups and document the results in Table 5.7. We chose a simple unpersonalized baseline, namely random recommendations and most popular items, to give more context to the performance results. Additionally, we ran two state-of-the-art methods based on collaborative filtering.

To our surprise and against the intuition that fashion is a visually based domain the textual embeddings provided the best results, answering RQ2 (Section 1.3). However, all VBPR-based results show only a small difference, which leads us to believe that the different feature spaces do not have a substantial influence on the performance or the algorithm is not the best one for this use case. The fact that ItemKNN provides, by far, the best results supports this hypothesis.

In conclusion, we showed that fine-tuning is a powerful approach that can be a simpler solution than engineering a whole frame around a task and that depending on the dataset, textual features can provide good recommendations even in a visually dominant domain such as fashion.

6.2 Contribution

The main contributions of this thesis are the answers provided to the research questions and the artifacts created during the process. Specifically, this work presents a detailed investigation into the capabilities of fine-tuned models for generating fashion item descriptions, which was done on real-life datasets extracted from online fashion stores. As well as a comprehensive evaluation of different feature spaces for the recommendation task using a recent dataset.

We outline the research questions that were answered in this thesis.

RQ1: To what extent can fine-tuning improve the performance of off-the-shelf image captioning models on domain-specific fashion datasets?

We show that fine-tuning can achieve competitive performance in models that were specifically trained for the semantics of fashion captions. This has the advantage of a simpler pipeline without the need for additional semantic rewards or multiple training stages. We can show that fine-tuning has its limitations that depend on the models as well as the dataset used for fine-tuning. In regards to the dataset, we see that models learn based on the ground truth to connect these to the images even if the descriptions include information that cannot be seen, e.g., back pockets and material used. Or to unlearn information, it would usually mention, e.g., color. However, if multiple images use the same description, the models seem to not be able to connect the caption as a whole to the item, especially if the images include different formats, e.g., images with humans, material images, and multiple versions of the same item. For the models we see that BLIP-2 works better than LLaVA because BLIP-2 is able to adjust the length of the generated text, and LLaVA models produce according to the set parameter, filling empty space or abruptly finishing sentences.

During this part we created the following artifacts:

- 12 fine-tuned models (6 models on 2 datasets)
- 2 fine-tuned models for sentence classification to extract categories from captions
- a fashion image captioning dataset extracted from the H&M dataset that includes attributes and categories

RQ2: Which feature embeddings (textual, visual, or multimodal) provide the best recommendations?

Using the H&M dataset this question is answered by saying the textual embeddings provided the best recommendations. We do acknowledge the limitations of our context, that we are using a temporal split instead of cross-validation relying on a single dataset. This result is still interesting as it shows that in a visually dominant domain such as fashion, textual descriptions can be as relevant for recommendations as multimodal or

visual information. However, traditional methods such as ItemKNN outperform the VBPR algorithm.

Other contributions beside the provided answers to the research questions are:

- 14 fine-tuned models
- application for qualitative analysis of image captions
- in-depth analysis of captions generated
- novelty by repurposing the H&M dataset for image captioning and using recent multimodal models
- reproducibility research by investigating the setup of Yang et al. [YZJ+] and discovering issues

6.3 Limitations and Future Work

Despite the contributions of this thesis, several limitations must be acknowledged. One key limitation is that even though we use fashion datasets, both show relevant differences as pointed out in Chapter 3. For better comparison, experiments could be repeated with a version of FACAD where images are sampled to have a 1-to-1 relationship to captions. Also, we admit that our fashion datasets are Western-focused and do not include Eastern clothing items, e.g., hijabs or saris.

The next limitation is the models. Nowadays, there is a large collection of models with multimodal capabilities that are provided in an open-source manner and are not included in this thesis, e.g., Qwen-VL [BBY+] and OpenFlamingo [ADL+], due to time limitations. Also, we did not include closed-source models as mentioned in Section 4.1. However, these gaps can be filled in the future.

Another limitation in regards to the models is the parameters that were set for LoRA as well as the training parameters. We cannot be sure that we achieved the best result possible for each setup, especially for the models that were also not able to run fully due to their size. This also affects the results of the recommendation, which are not evaluated with cross-validation and were not optimized with hyperparameter tuning.

For future work, several directions could be pursued to enhance and extend the findings of this thesis. McKinzie et al.[MGF+] demonstrate that the image encoder, along with image resolution and the number of image tokens, has a significant impact, whereas the design of the vision-language connector is relatively insignificant. One possible direction is experimenting with different encoders to assess whether they improve the model's ability to distinguish fine details. Another promising application lies in text-based image retrieval systems. Given that companies often manage vast collections of product images, leveraging vision-language models to automatically generate captions and metadata could

significantly improve search efficiency, helping customers find the exact products they are looking for.

In conclusion, we see the findings of this thesis as an invitation for further exploration in this field, encouraging more experiments to refine and expand upon these insights.

List of Figures

List of Tables

Bibliography

[ABF+] Vito Walter Anelli, Alejandro Bellogin, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco Maria Donini, and Tommaso Di Noia. Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, pages 2405–2414. Association for Computing Machinery.

[ADF+] Matteo Attimonelli, Danilo Danese, Angela Di Fazio, Daniele Malitesta, Claudio Pomo, and Tommaso Di Noia. Ducho meets elliot: Large-scale benchmarks for multimodal recommendation.

[ADL+] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning.

[ADM+] Matteo Attimonelli, Danilo Danese, Daniele Malitesta, Claudio Pomo, Giuseppe Gassi, and Tommaso Di Noia. Ducho 2.0: Towards a more up-to-date unified framework for the extraction of multimodal features in recommendation.

[ADS] Jyoti Aneja, Aditya Deshpande, and Alexander Schwing. Convolutional image captioning.

[AFJG] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. SPICE: Semantic propositional image caption evaluation.

[AHB+] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering.

[BA] Shuang Bai and Shan An. A survey on automatic image caption generation. 311:291–304.

[BBY+] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond.

[BCE+] Raffaella Bernardi, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, and Barbara Plank. Automatic description generation from images: A survey of models, datasets, and evaluation measures.

[BGRK] Ali Furkan Biten, Lluis Gomez, Marçal Rusiñol, and Dimosthenis Karatzas. Good news, everyone! context driven entity-aware captioning for news images.

[BHV] Christian Bracher, Sebastian Heinz, and Roland Vollgraf. Fashion DNA: Merging content and sales data for recommendation and article mapping.

[Bie] Lukas Biewald. Experiment tracking with weights and biases.

[BMR+] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners.

[BSC+] Manuele Barraco, Matteo Stefanini, Marcella Cornia, Silvia Cascianelli, Lorenzo Baraldi, and Rita Cucchiara. CaMEL: Mean teacher learning for image captioning.

[BV] Chris Buckley and Ellen M. Voorhees. Evaluating evaluation measure stability. 51(2):235–242. Number of pages: 8 Place: New York, NY, USA Publisher: Association for Computing Machinery tex.issue_date: July 2017.

[CFL+] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server.

[CHL+] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models.

[CLL+] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality.

[CSBC] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. Meshed-memory transformer for image captioning.

[CSC+] Wen-Huang Cheng, Sijie Song, Chieh-Yun Chen, S. Hidayati, and Jiaying Liu. Fashion meets computer vision. 54:1 – 41.

[DBK+] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale.

[DCLT] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding.

[DHR+] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description.

[DL] Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. volume 6, pages 376–380.

[DNR+] Yashar Deldjoo, Fatemeh Nazary, Arnau Ramisa, Julian McAuley, Giovanni Pellegrini, Alejandro Bellogin, and Tommaso Di Noia. A review of modern fashion recommender systems. 56(4):87:1–87:37.

[DPHZ] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs.

[DXS+] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-e: An embodied multimodal language model.

[DYL] Bo Dai, Deming Ye, and Dahua Lin. Rethinking the form of latent states in image captioning.

[GPM] Taraneh Ghandi, Hamidreza Pourreza, and Hamidreza Mahyar. Deep learning approaches on image captioning: A review. 56(3):1–39.

[GQLD] Congying Guan, Shengfeng Qin, Wessie Ling, and Guofu Ding. Apparel recommendation system evolution: an empirical review. 28(6):854–879. Publisher: Emerald Group Publishing Limited.

[HDW+] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Barun Patra, Qiang Liu, Kriti Aggarwal, Zewen Chi, Johan Bjorck, Vishrav Chaudhary, Subhojit Som, Xia Song, and Furu Wei. Language is not all you need: Aligning perception with language models.

[HG] Wei-Lin Hsiao and Kristen Grauman. Learning the latent "look": Unsupervised discovery of a style-coherent embedding from fashion images.

[HKBS] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words.

[HMa] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering.

[HMb] Ruining He and Julian McAuley. VBPR: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 144–150. AAAI Press.

[Hoc] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. 06(2):107–116.

[HRM+] Alan Hevner, Alan R, Salvatore March, Salvatore T, Park, Jinsoo Park, Ram, and Sudha. Design science in information systems research. 28:75.

[HS] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 9(8):1735–1780. Publisher: MIT Press.

[HSSL] Md Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning.

[HSW+] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models.

[HYH] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: data, models and evaluation metrics. 47(1):853–899.

[HZRS] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition.

[JK] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. 20(4):422–446. Number of pages: 25 Place: New York, NY, USA Publisher: Association for Computing Machinery tex.issue_date: October 2002.

[Kal] Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with LoRA.

[KFF] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions.

[KFWM] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. Visually-aware fashion recommendation and design with generative image models.

[Kim] Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

[KLP+] Sungkyung Kim, Adam Lee, Junyoung Park, Sounho Chung, Jusang Oh, and Jay-Yoon Lee. Parameter-efficient fine-tuning of InstructBLIP for visual reasoning tasks.

[KM] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics.

[KZG+] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations.

[LBPL] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

[LHR+] Carlos García Ling, Elizabeth HMGroup, Frida Rim, Inversion, Jaime Ferrando, Maggie, Neuraloverflow, and Xlsrln. H&m personalized fashion recommendations.

[Lina] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. page 10.

[Linb] Carlos García Ling. H&m personalized fashion recommendations.

[LLLL] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning.

[LLSD] Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning.

[LLSH] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models.

[LLWL] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning.

[LLXH] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation.

[LM] Katrien Laenen and Marie-Francine Moens. Attention-based fusion for outfit recommendation.

[LMB+] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, Lecture Notes in Computer Science, pages 740–755. Springer International Publishing.

[LSY] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. 7(1):76–80. Conference Name: IEEE Internet Computing.

[LWW] Qiang Liu, Shu Wu, and Liang Wang. DeepStyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 841–844. Association for Computing Machinery.

[LXPS] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning.

[LYL+] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks.

[LZLY] Guang Li, Linchao Zhu, Ping Liu, and Yi Yang. Entangled transformer for image captioning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8927–8936. ISSN: 2380-7504.

[MBM+] Nicholas Moratelli, Manuele Barraco, Davide Morelli, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Fashion-oriented image captioning with external knowledge retrieval and fully attentive gates. 23(3):1286. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

[MBS] Kevin Matzen, Kavita Bala, and Noah Snavely. StreetStyle: Exploring world-wide clothing styles from millions of photos.

[MGF+] Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, Anton Belyi, Haotian Zhang, Karanjeet Singh, Doug Kang, Ankur Jain,

Hongyu Hè, Max Schwarzer, Tom Gunter, Xiang Kong, Aonan Zhang, Jianyu Wang, Chong Wang, Nan Du, Tao Lei, Sam Wiseman, Mark Lee, Zirui Wang, Ruoming Pang, Peter Grasch, Alexander Toshev, and Yinfei Yang. MM1: Methods, analysis & insights from multimodal LLM pre-training.

[MMH+] Utkarsh Mall, Kevin Matzen, Bharath Hariharan, Noah Snavely, and Kavita Bala. GeoStyle: Discovering fashion trends and events.

[NZ] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729.

[Ope] OpenAI. GPT-4 technical report.

[Por] M. Porter. Snowball: A language for stemming algorithms.

[PRWZ] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318. Association for Computational Linguistics.

[QDZL] Yu Qin, Jiajun Du, Yonghua Zhang, and Hongtao Lu. Look back and predict forward in image captioning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8359–8367. ISSN: 2575-7075.

[QZZ+] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

[RIS+] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, CSCW '94, pages 175–186. Association for Computing Machinery.

[RKH+] 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. Learning transferable visual models from natural language supervision.

[RMM+] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1195. ISSN: 1063-6919.

[RN] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training.

[RYMNM] Arnau Ramisa, Fei Yan, Francesc Moreno-Noguer, and Krystian Mikolajczyk. BreakingNews: Article annotation by image and text processing.

[SBMN] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing with compositional vector grammars. In Hinrich Schuetze, Pascale Fung, and Massimo Poesio, editors, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465. Association for Computational Linguistics.

[SCB+] Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. From show to tell: A survey on deep learning-based image captioning.

[SKC+] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D. Manning. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the Fourth Workshop on Vision and Language*, pages 70–80. Association for Computational Linguistics.

[SMSC] Shaghayegh Shirkhani, Hamam Mokayed, Rajkumar Saini, and Hum Yan Chai. Study of AI-driven fashion recommender systems. 4.

[SP] Demis Hassabis Sundar Pichai. Introducing gemini: our largest and most capable AI model.

[SVL] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks.

[TB] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers.

[TC] Oyvind Tafjord and Peter Clark. General-purpose question-answering with macaw.

[TCD+] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention.

[VSP+] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need.

[VTBE] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator.

[VZP] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based image description evaluation.

[WBM+] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-UCSD birds 200.

[Wik] Wikipedia contributors. METEOR — wikipedia, the free encyclopedia.

[XBK+] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention.

[Yan] Xuewen Yang. xuewyang/fashion_captioning. original-date: 2020-07-16T00:48:41Z.

[YKB] Kota Yamaguchi, M. Hadi Kiapour, and Tamara L. Berg. Paper doll parsing: Retrieving similar styles to parse clothing items. In *2013 IEEE International Conference on Computer Vision*, pages 3519–3526. ISSN: 2380-7504.

[YLHH] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. 2:67–78. Place: Cambridge, MA Publisher: MIT Press.

[YLL] Wei Yang, Ping Luo, and Liang Lin. Clothing co-parsing by joint image segmentation and labeling. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3182–3189.

[YPLM] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning.

[YR] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR 2011*, pages 1385–1392. IEEE.

[YSR+] Fan Yang, Xueping Su, Jie Ren, Xiaomin Ma, and Yongyong Han. A survey of image captioning algorithms based on deep learning. In *2022 International Conference on Image Processing and Media Computing (ICIPMC)*, pages 108–114.

[YZC] Xu Yang, Hanwang Zhang, and Jianfei Cai. Learning to collocate neural modules for image captioning.

[YZJ+] Xuewen Yang, Heming Zhang, Di Jin, Yingru Liu, Chi-Hao Wu, Jianchao Tan, Dongliang Xie, Jue Wang, and Xin Wang. Fashion captioning: Towards generating accurate descriptions with semantic rewards.

[ZPZ+] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and VQA.

[ZRG+] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. OPT: Open pre-trained transformer language models.