TU WIEN Informatics

Enhancing Image Retrieval Re-Ranking using Mutual Information Minimization

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Marvin Seidl, BSc
Matrikelnummer 11777747

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec Schahram Dustdar
Mitwirkung: Univ.Ass. Dipl.-Ing. Alireza Furutanpey

Wien, 17. Oktober 2024

Marvin Seidl	Schahram Dustdar

TU WIEN Informatics

Enhancing Image Retrieval Re-Ranking using Mutual Information Minimization

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Marvin Seidl, BSc
Registration Number 11777747

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec Schahram Dustdar
Assistance: Univ.Ass. Dipl.-Ing. Alireza Furutanpey

Vienna, October 17, 2024

Marvin Seidl	Schahram Dustdar

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

Erklärung zur Verfassung der Arbeit

Marvin Seidl, 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.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, habe ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT-Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 17. Oktober 2024

Marvin Seidl

Danksagung

Zunächst möchte ich meinen Betreuern Dr. Schahram Dustdar und Dipl.-Ing. Alireza Furutanpey für die Möglichkeit, diese Arbeit zu erstellen, bedanken. Die „Distributed Systems Group“ hat nicht nur die notwendigen Rechenressourcen bereitgestellt, sondern war darüber hinaus eine positive und freundliche Arbeitsumgebung.

Ich bin besonders dankbar für die Unterstützung von Alireza, welcher mir sehr wertvolles und detailliertes Feedback gegeben hat und das noch dazu immer in einer beeindruckend schnellen Zeit. Auch möchte ich mich dafür bedanken, dass er mich ermutigt hat, mein eigenes Forschungsthema zu finden sowie für seine generelle Hilfe bei der Gestaltung dieser Arbeit.

Ich möchte mich auch bei meinen Freunden bedanken, mit denen ich während meines Studiums viele schöne Momente, Biere und lange Nächte geteilt habe. Ohne diese emotionale Unterstützung wäre das Studium und diese Arbeit einfach nicht möglich gewesen.

Abschließend möchte ich meinen Eltern und Großeltern für ihre unerschütterliche Unterstützung danken. Danke, dass ihr immer an mich geglaubt habt und dass ich von euch im Grunde bedingungslose Liebe erfahre.

Acknowledgements

First and foremost, I would like to thank my advisors, Dr. Schahram Dustdar and Dipl.-Ing. Alireza Furutanpey, for enabling me to pursue this work. Besides simply providing computational resources, the *Distributed Systems Group* has been a positive and helpful environment.

I am especially grateful for the support from Alireza, who not only provided invaluable and detailed feedback but did so in an incredibly fast and responsive way. Additionally, I want to thank him for encouraging me to find my own research topic and for his overarching help in shaping this work.

Second, I would like to thank my friends with whom I have shared many memories, beers, and late nights throughout my studies and this work. Without this bedrock of emotional support, this would simply not have been possible.

Lastly, I would like to thank my parents and grandparents for their unwavering support. Thank you for always believing in me and for showing, what is basically, unconditional love.

Kurzfassung

Content-Based Image Retrieval zielt darauf ab, relevante Bilder in einer Datenbank anhand des visuellen Inhaltes eines Anfragebildes zu finden.

Ein häufig verwendeter Ansatz ist es, zwei Arten von erlernten Bildrepräsentationen zu verwenden. Globale Repräsentationen erfassen die Semantik auf einer komplexen Ebene, während lokale Repräsentationen die Semantik auf einer einfachen Ebene erfassen. *Re-ranking* wird benutzt um den Suchraum einzuschränken. Zuerst werden Bilder anhand der globalen Repräsentation vorgefiltert und dann mittels *Geometric Verification* der lokaler Repräsentationen umgereiht. *Geometric Verification* funktioniert anhand der räumlichen Position der lokalen Repräsentationen, lässt aber die Ähnlichkeit anhand globaler Repräsentationen außer Acht. Bei aktuellen Methoden kommt es zu einer beträchtlichen Menge an Redundanz zwischen globalen und lokalen Repräsentationen. Die Dimensionalität der Repräsentationen im latenten Raum ist begrenzt, weshalb diese Redundanz die Ausdruckskraft der Repräsentationen beeinträchtigt. Eine Verringerung der Redundanz sollte daher die Effizienz des *Re-Rankings* verbessern.

In dieser Arbeit wird vorgeschlagen, informationstheoretische Konzepte und *Multi-View Representation Learning* zu nutzen, um die Redundanz zwischen globalen und lokalen Repräsentationen zu verringern. Zunächst untersuchen wir den Effekt von Transinformation zwischen Repräsentationen auf *Image Retrieval* Systeme. Um Redundanz zu "bestrafen", fügen wir die Schätzung von Transinformation als kontrollierbaren Faktor zum Optimierungsziel des Netzwerkes hinzu. Das Modell ist durchgängig mittels *image-level supervision* trainierbar.

Wir evaluieren unsere Methodik anhand zweier Ansätze der Schätzung von Transinformation und des *Re-Rankings*. Wir führen Experimente auf dem *Revisited Oxford and Paris* sowie dem *Stanford Online Products* Datensatz durch. Unsere Ergebnisse zeigen, dass die Reduktion von Redundanz durch Schätzung von Transinformation das *Re-Ranking* deutlich verbessern kann.

Abstract

Content-Based Image Retrieval aims to find relevant images in a database given the visual content of a query image.

A common setup is using learned feature extractors to obtain two types of image descriptors. Global features capture high-level semantics, while local features encode low-level details. Re-ranking is used to reduce the search space. First, images are matched using global feature similarity and then re-ranked using geometric verification of local features. Geometric verification works based on the spatial location of local features but ignores global feature similarity. However, current methods leave considerable redundancy between global and local features. Since latent dimensions are finite, the redundancy inhibits expressiveness. Therefore, reducing redundancy should improve re-ranking performance.

This work proposes drawing from information-theoretic concepts and multi-view representation learning to minimize redundancy between global and local features. We first investigate the degree and effect of mutual information between representations in image retrieval systems. Then, we apply (neural) mutual information estimation as a controllable term that penalizes redundancy during training. The model is end-to-end trainable using image-level supervision.

We evaluate our methodology using two approaches to mutual information estimation and re-ranking. We perform experiments on the Revisited Oxford and Paris datasets and the Stanford Online Products dataset. Our results demonstrate that reducing redundancy with (neural) information estimation can significantly improve re-ranking.

Contents

Kurzfassung **xi**

Abstract **xiii**

Contents **xv**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Problem Statement . . . 2
1.3 Solution Approach . . . 2
1.4 Research Questions . . . 4
1.5 Aim of the Thesis . . . 5
1.6 Structure . . . 5

2 Background **7**
2.1 Information Theory . . . 7
2.2 Representation Learning . . . 9
2.3 Image Retrieval . . . 10
2.4 Measuring Mutual Information . . . 15

3 Related Work **19**
3.1 Multi-View Representation Learning . . . 19
3.2 Image Retrieval . . . 22

4 Solution Approach **25**
4.1 Underlying Methods . . . 26
4.2 Feature Export for Re-Ranking with RRT . . . 32
4.3 Measurement and Minimization of Mutual Information . . . 33

5 Evaluation Methodology **37**
5.1 Evaluation . . . 37
5.2 Experiments . . . 41

6 Results **45**

6.1 Benchmark Comparison 45
6.2 Minimization of Mutual Information 46
6.3 Performance Improvements of Re-Ranking 55

7 Conclusion **59**
7.1 Research Questions 59
7.2 Limitations & Future Work 61

A Retrieval Results for Individual Runs **63**

Übersicht verwendeter Hilfsmittel **69**

List of Figures **71**

List of Tables **73**

Acronyms **75**

Bibliography **77**

CHAPTER 1

Introduction

1.1 Motivation

The task of *Content-Based Image Retrieval* is to find relevant images out of a database given the visual content of a *query* image. State-of-the-art methods use deep neural networks to extract image feature representations (or "descriptors"). A typical setup uses two feature representation types: global and local features [14].

Global features summarize the whole image into a single representation consisting of a high dimensional vector. The representation captures high-level similarities and abstract concepts but loses information about the spatial arrangement of the image contents [12]. *Local Features* are associated with specific regions of the image and describe *keypoints* for the relevant object in the image. Modern approaches like DELG [12] can extract both feature types simultaneously.

An ideal retrieval system would exploit the benefits of both feature types. Similarity of the global feature is efficiently computable, e.g., using the L2 norm. However, calculating image similarity by local features is computationally expensive. Matching all images in the database against the query image is unfeasible. *Re-ranking* is commonly used to cut down on the search space [14]. First, images are matched by their global descriptor, and only the top-N results are re-ordered by the similarity of their local descriptors. *Geometric verification* is one such approach to re-ranking. It considers matching local features as the images' keypoints and evaluates the best fit of a perspective transformation to the location of these keypoints. The goal is to determine which parts of an image correspond to which parts of another image, for example, to answer whether two images of a building from different viewpoints depict the same building.

1.2 Problem Statement

Re-ranking using geometric verification [12] only considers the similarity based on local features. Apart from selecting the top N images for re-ranking in the first place, the initial ordering by global similarity is discarded. We observe the following:

- By calculating the global similarity, we gain information about the final order of the retrieved images. Geometric verification as a re-ranking approach ignores this information.
- Both global and local features contain sufficient information to decide the final order. Therefore, local and global features should contain redundant information. The information content of these representations is limited. Without this redundant component, the local features could better use their finite latent dimension and expressive power to encode auxiliary information not already present in the global feature.

We adopt the information-theoretic concept of *mutual information*, a measure of statistical dependence between two random variables to reason about redundant information. Compared to correlation coefficients, mutual information can additionally capture non-linear dependencies at the cost of being harder to compute.

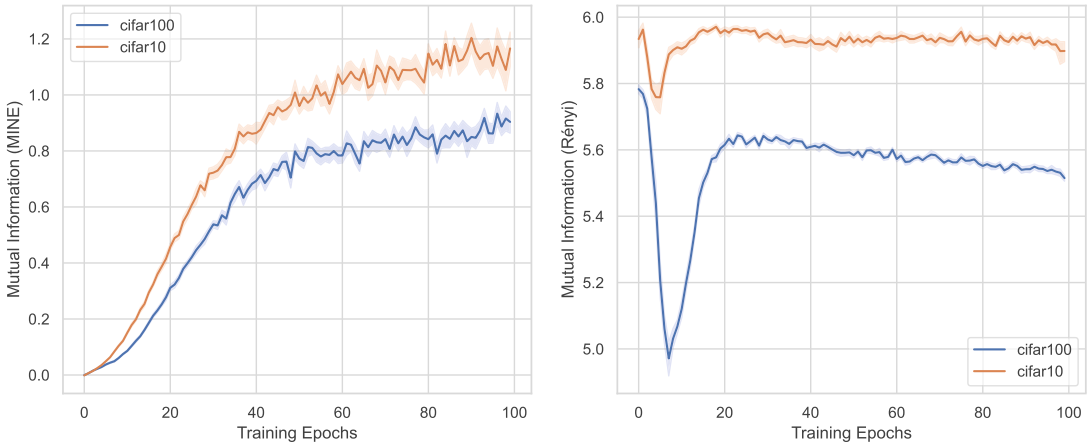
A significant challenge of using mutual information for representation learning in modern machine learning is its runtime complexity. Due to the high dimensionality of the input data and target representations, the calculation is intractable in general. Variational approximation and optimizing a bound on mutual information is commonly used instead [47]. These methods require explicit knowledge about the distribution of one of the random variables. In representation learning, this restriction is placed upon the learned representation when the true underlying distribution of the input data is unknown.

Recently, several general purpose estimators of mutual information have been introduced, such as MINE [8], NWJ [42], or NCE [65]. General purpose estimators do not rely on prior knowledge of a particular distribution. Instead, they require an auxiliary neural network that optimizes a lower bound by estimating expectations over sample distributions. Another promising direction is using kernel-based methods based on Rényi's α-order entropy [52].

To demonstrate the existence of non-zero mutual information between global and local descriptors, we trained a feature extractor similar to [12]. We used MINE [8] and Rényi's α-order matrix-based functional [53] to estimate mutual information. Global features were extracted from the third layer, and local features from the second layer of a ResNet-18. Figure 1.1 shows the mutual information between these feature types.

1.3 Solution Approach

Modern *image retrieval* is a form of *representation learning*. The representation should contain as much relevant information about an image as possible while discarding irrelevant



(a) MI estimated using MINE (b) MI estimated using Rényi

Figure 1.1: Mutual Information between Global and Local Descriptors. Results averaged over three training runs on the CIFAR10 and CIFAR100 datasets. a) shows the estimation using MINE [8]. b) shows the estimation using Rényi's α-order matrix-based functional [53]. The X-axis depicts the training progress in epochs. The Y-axis depicts the estimated mutual information values by the two different methods. Note that MINE gradually increases its lower bound during training, so the estimates of early training steps do not represent the actual mutual information value.

information. Multi-view representation learning extends this concept to incorporate information from multiple data views. The data can be heterogeneous, like image-text, video-text, or audio-text. Alternatively, it can be homogeneous, like two images depicting the same object from different viewpoints.

Reasoning what information is relevant and can be discarded is challenging. In supervised learning, we can consider relevant information that allows the representation to predict a target label. The Information Bottleneck (IB) method [63] formalizes the trade-off between preservation and compression in representations using mutual information. In multi-view representation learning, mutual information encourages or discourages shared information between views and representations. We consider our image retrieval system's global and local descriptors to be two views on the same underlying image data in the framework of multi-view representation learning.

Reranking Transformer (RRT) [62] is a re-ranking approach incorporating global and local descriptors. This alleviates the issue of geometric verification ignoring global similarity. RRT can be used with any feature extractor that produces global and local descriptors, and we evaluate it as an alternative to geometric verification. The second issue remains, and we hypothesize that reducing the redundant information between global and local features increases retrieval performance.

1.4 Research Questions

RQ1: How can we measure and minimize mutual information between global and local features for image retrieval?

Directly measuring mutual information in a very high-dimensional setting is not tractable. Several recent methods have been proposed to approximate or bound mutual information.

Can we observe mutual information between global and local features? To minimize, we first need to measure MI in a way that makes it possible to use this measurement as part of a loss function for neural network training.

Can (neural) mutual information estimation adequately minimize mutual information? In this work, we will investigate the usage of MINE [8] as a representative among similar methods and of Rényi's α-order matrix-based functional [53] for minimizing mutual information between feature types in the context of an image retrieval system. Both methods have already been applied to the information bottleneck method.

RQ2: How does minimizing mutual information between global and local features affect the local features?

Mutual information is a high-level measure of statistical dependence and provides valuable additional insight into the random variables. Are there any other measurable effects apart from a reduced mutual information estimation?

RQ3: How does minimizing mutual information between global and local features affect re-ranking performance?

We experiment with two re-ranking approaches: *Geometric verification* and RRT [62].

Does minimizing mutual information improve re-ranking performance? We hypothesize that penalizing mutual information between global and local features forces the network to encode less redundant information in local features, leading to improved re-ranking performance.

Does minimizing mutual information affect the two approaches differently? *Geometric verification* does not consider global feature similarity when calculating the final retrieval rank. RRT uses global and local features to provide an image similarity score directly. We hypothesize that reducing MI affects *geometric verification* negatively as the missing global feature should contain the information we consider as redundant. Conversely, RRT should not be negatively affected as it includes the global feature in its input sequence.

1.5 Aim of the Thesis

The main goal of this thesis is to develop a novel image retrieval system that improves upon previous work by explicitly modeling the information-theoretic relationship between global and local feature representations. We integrate two methods of estimating mutual information into a feature extraction model and penalize the estimated value in the training objective. We compare the quality of our learned features using two methods for re-ranking. We evaluate our implementation on standard benchmark datasets and perform experiments to answer the outlined research questions. Overall, our method aims to improve the quality of the feature representations, leading to improved re-ranking performance.

1.6 Structure

Chapter 2 explains some mandatory and preliminary knowledge for our work. Chapter 3 provides an overview of related and relevant literature for the fields of *Multi-View Representation Learning* and *Image Retrieval*. Chapter 4 presents our solution approach and details how we implement our model and perform the minimization of mutual information. Chapter 5 details how we evaluate our model and which experiments we perform. Chapter 6 presents the results of these experiments. Chapter 7 concludes this thesis with a discussion of the research results in the context of the presented research questions. We also outline future research directions and discuss our work's limitations.

CHAPTER 2

Background

2.1 Information Theory

Entropy describes the information content of a discrete random variable $X \sim p$ [54]. It provides a measure of uncertainty about the outcomes of the random variable: A surprising result provides more information than an unsurprising one. The *information content* is defined as the negative-log probability of an outcome and the *entropy* as the expected value of the information content. The basis of the logarithm decides the unit of information. For $\log_2$, the unit is *bits*.

$$H[\mathbf{X}] = -\mathop{\mathbb{E}}_{x\sim p}[\log\ p(\mathbf{x})] = -\sum_{x\in\mathcal{X}} p(x)\log\ p(x) \tag{2.1}$$

Joint Entropy describes the entropy of the joint probability distribution of a pair of random variables (X, Y).

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x,y) \tag{2.2}$$

Conditional Entropy is defined analogously to the conditional expectation. It describes the amount of information needed for the outcome of one random variable, given that the outcome of another is known.

$$H(X|Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)} \tag{2.3}$$

Note that the chain rule of probability translates to the definitions of conditional and joint entropy:

$$H(X,Y) = H(X) + H(Y|X) \tag{2.4}$$

Mutual Information (MI) measures the shared information between two random variables. It can be defined based on the (conditional) entropy or joint entropy of the variables:

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) \end{aligned} \tag{2.5}$$

Mutual Information is symmetric and non-negative. If the two random variables are independent, then the mutual information is zero, as knowing one does not provide any insight about the other. If both variables are in a deterministic relationship, then the conditional entropy term is zero, and the mutual information is just the entropy of both variables alone.

In the case of discrete random variables, the mutual information can be defined based on their marginal $(p(x), p(y))$ and joint probability $(p(x,y))$ distributions:

$$I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{2.6}$$

Alternatively, it can be expressed by the Kullback-Leibler (KL) divergence between the product of the marginal distributions and their joint distribution:

$$I(X;Y) = D_{KL}(P_{XY} || P_X \times P_Y) \tag{2.7}$$

The KL divergence measures the dissimilarity between a probability distribution Q and a reference distribution P. It is also commonly referred to as *relative entropy*. Encoding samples of the reference distribution P with a code optimized for Q results in an average information overhead of $D_{KL}(P||Q)$ bits (compared to directly encoding it with a code for P).

$$D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \tag{2.8}$$

In contrast to MI, the KL divergence is not symmetric. The divergence is zero if and only if Q and P are equal. For MI, this is the case when the product of the marginal $(p(x) \times p(y))$ and joint probability distributions $(p(x,y))$ are equal, which would mean statistical independence of X and Y.

2.2 Representation Learning

Representation learning aims at discovering *representations* (or *features*) suitable for machine learning tasks such as classification by uncovering underlying structures in raw data [9]. Instead of manually engineering features, representation learning algorithms learn to transform data into representations that capture relevant information for downstream tasks. Modern *Image Retrieval* makes extensive use of learned representations. Good representations can improve performance by improving generalization. They can also reduce the computational complexity of downstream tasks by transforming high-dimensional data into lower-dimensional data [25]. Discerning relevant information from irrelevant information is challenging, and multiple definitions of effective representation exist.

2.2.1 Information Bottleneck

The Information Bottleneck (IB) [63] method extracts relevant information from a random input variable X about a target variable Y. The core idea is to create a compressed representation of X, denoted here as T, that retains as much information as possible about Y while minimizing the information it carries about itself.

The data processing inequality states that processing data cannot increase the information content about the target variable. The above-mentioned random variables form a Markov chain $Y \leftrightarrow X \leftrightarrow T$, and for any such Markov chain, the data processing inequality implies that:

$$I(X;Y) \geq I(T;Y) \tag{2.9}$$

As the representation T can only lose information, the IB method provides a tradeoff between compression and prediction formalized by the two mutual information terms $I(X;T)$ and $I(Y;T)$. Lagrangian relaxation with the parameter β is used to control the tradeoff, and the overall goal is to minimize the following functional:

$$L_{IB} = I(X;T) - \beta I(T;Y) \tag{2.10}$$

The first term aims to compress the representation T, while the second aims to preserve information about the target Y. The target Y allows us to specify which information of X is considered relevant and should be contained in T.

2.2.2 Deep Learning and the Information Bottleneck

Tishby and Zaslavsky [64] proposed that Deep Neural Networks (DNNs) implicitly perform information bottleneck optimization. As such, the IB method provides a framework for understanding neural networks, where the output of each layer is a representation of the input data. This process can be interpreted as a Markov chain $Y \leftrightarrow X \leftrightarrow T_1 \leftrightarrow T_2 \leftrightarrow ... \leftrightarrow T_n$ where each intermediate representation T_i captures

progressively more abstract features of X relevant to predicting Y. Again, the data processing inequality implies that each layer of the neural network can only lose but not gain more information about the target task than the input X.

$$I(X;Y) \geq I(T_1;Y) \geq I(T_2;Y) \geq ... \geq I(T_n;Y) \tag{2.11}$$

Tishby and Zaslavsky argue that during training, DNNs undergo two distinct phases: fitting and compression. In the fitting phase, the network extracts information from the input, increasing $I(X;T)$. In the compression phase, the network refines its representations to minimize $I(X;T)$ while maintaining or increasing $I(T;Y)$ thus discarding irrelevant details that do not aid in predicting Y [64].

2.2.3 Multi-View Representation Learning

Multi-view representation learning aims at integrating information from multiple views of the same data to increase predictive performance. The data can be heterogeneous across different modality pairs like image-text, video-text, or audio-text. Alternatively, it can consist of multiple views on only one modality, like multiple augmented pictures based on one original image in the case of self-supervised learning. As different views usually contain complementary information, the goal is to learn better representations than possible by only observing a single view [34]. [34] propose a taxonomy based on two major categories: *Multi-View representation alignment*, where an embedding function first maps each individual view input into a common representation space, then some sort of alignment or distance loss between the obtained representations is optimized. [1, 3, 5, 71] described in section 3.1 are examples of this group. *Multi-view representation fusion*, where a single fused representation is obtained directly. [24, 39, 67, 70, 75, 81] described in section 3.1 are examples of this group.

2.3 Image Retrieval

The task of *Content-Based Image Retrieval* is to find relevant images in a database, given the visual content of a query image. Most image retrieval systems aim to recognize a specific instance of an object, e.g., Schönbrunn Palace, the Eiffel Tower, or the Golden Gate Bridge [14]. In contrast, image *classification* aims to recognize classes or categories of objects, e.g., buildings, cats, or planes. As only the content of an image is relevant, the distinction between instances or classes is not important for the system itself. In literature the number of classes or instances dictates the framing: For few instances the problem is referred to as *classification* and for many as *(instance-level) retrieval/recognition* [1]. Image retrieval has numerous applications, such as landmark retrieval [73], online product search [60], remote sensing [35] or face retrieval [20].

[1] If not stated otherwise, this text uses the terms *image retrieval*, *content-based image retrieval*, *instance-level retrieval* and *instance-level recognition* interchangeably.

The image database is usually referred to as the *image gallery* or *image index*. In real-world applications, the gallery can contain millions of images. Managing and efficiently retrieving images thus requires highly efficient feature extraction and similarity computation.

The core challenge of image retrieval is the development of feature representations that fulfill the following requirements:

- **Discriminative**

  Features must have discriminative power, being distinct enough to differentiate between object instances and scenes while minimizing false matches.

- **Compact**

  Features should be scalable, meaning they should be computationally efficient to extract and compare, thus enabling rapid searches over vast datasets.

- **Robust**

  Features must be invariant to geometric transformations (such as rotations, scaling, and translation) and photometric changes (such as variations in brightness, contrast, and color)

2.3.1 Feature Types

We consider the common image retrieval setup that fulfills the above-mentioned requirements [14]. It uses two types of feature representations: global and local features.

Global features summarize the whole image into a single representation, most commonly an *embedding* vector. This representation of the visual content is compact and captures high-level similarities well, requiring a level of abstraction invariant to a viewpoint and photometric transformations [12]. Information about the spatial arrangement of individual elements in an image is lost, and unrelated elements to the object instance might influence the representation.

Local Features are associated with specific image regions and describe keypoints for the depicted instance. They contain localized visual information as well as spatial information, which allows the application of geometric consistency checks (also called *geometric verification*) using techniques such as RANSAC [11]. This verification step reflects image similarity more reliably than global feature similarity, making local features especially suited for rigid objects [12].

Some recent work diverges from the classic feature types, e.g., [55, 76]. Section 3.2 elaborates on alternative approaches.

2.3.2 Feature Extraction

The history of image retrieval can be roughly categorized into two periods: feature engineering using hand-crafted techniques and feature learning using DNNs.

SIFT [36], SURF [7] and Bag-of-visual-Words [59] are examples of hand-crafted algorithms that detect local features. The detected features were directly matched against a large database of local descriptors or aggregated first to provide a global similarity score.

Learned representations are more effective than hand-crafted methods. A DNN learns to extract features from raw pixel data during training. Convolutional Neural Networks (CNNs) are the most common architecture for visual applications. They can learn features with multiple levels of abstraction: Early network layers learn low-level features like edges, corners, or textures and deeper layers capture high-level features such as objects, faces, or buildings. Modern systems can jointly extract local and global features while only being trained on a conceptually simple classification task [12]. Recently, transformer-based architectures without convolutions have outperformed CNNs in the visual domain [22].

2.3.3 Two-Step Retrieval: Re-Ranking

Calculating the similarity between all local features of all images in the database is inefficient. *Re-ranking* is a common approach to cut down the search space in retrieval systems: First, images are ordered by the similarity of their global descriptor, and then only the top-N results are re-ranked using some form of domain-specific refinement. For image retrieval, this is often geometric verification based on the positions of the local descriptors in the image.

While the global descriptor similarity calculation is fast (e.g., dot product, L2 distance, or cosine similarity), geometric verification is computationally expensive. Re-ranking allows the benefits of more complex calculations for important top retrieval results while providing a good tradeoff for performance. Figure 2.1 depicts the overall re-ranking process.

Random Sample Consensus (RANSAC) [11] is commonly used to implement geometric verification. The goal is to determine whether two images depict the same keypoints from different viewpoints. RANSAC is a general purpose and robust algorithm to fit a mathematical model to data that contains outliers. It works by repeatedly evaluating hypothesis models for random subsets. An affine transformation is usually used as the model for the application in image retrieval. It acts as an approximation for the projective transformation or *homography* and needs at least three point pairs in 2D space to fit. Algorithm 2.1 shows the steps of the method. It requires the model, number of iterations, and residual threshold as parameters.

RANSAC is repeated for all top-N gallery images. Point pairs are generated by matching local features between query and gallery images based on L2 distance. Finally, Images are ordered by their highest number of transformation inliers.

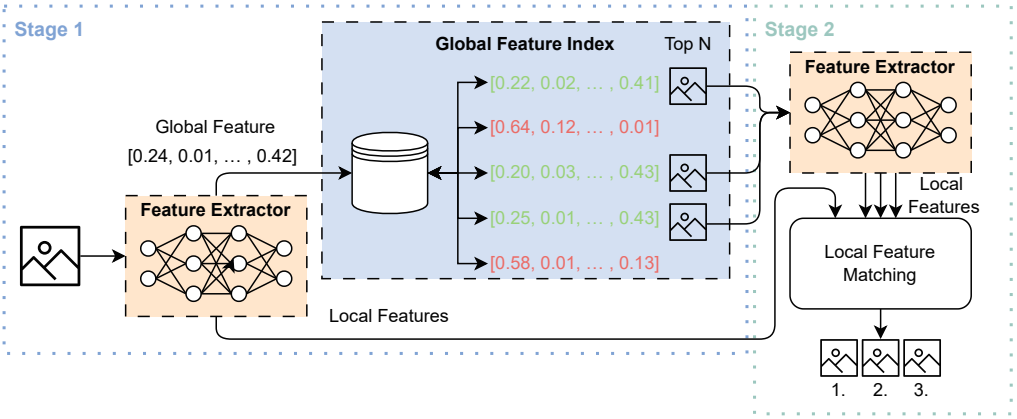


Figure 2.1: The image retrieval re-ranking process

Algorithm 2.1: RANSAC

```
forall iterations do
    randomly sample enough data points to calculate parameters of model ;
    calculate parameters of the model ;
    count number of data points (inliers) based on residual threshold to model;
    if new best model by number of inliers then
        record set of inliers ;
    end
end
calculate final model using all data points of best inlier set ;
```

2.3.4 Learning Feature Extractors using Classification

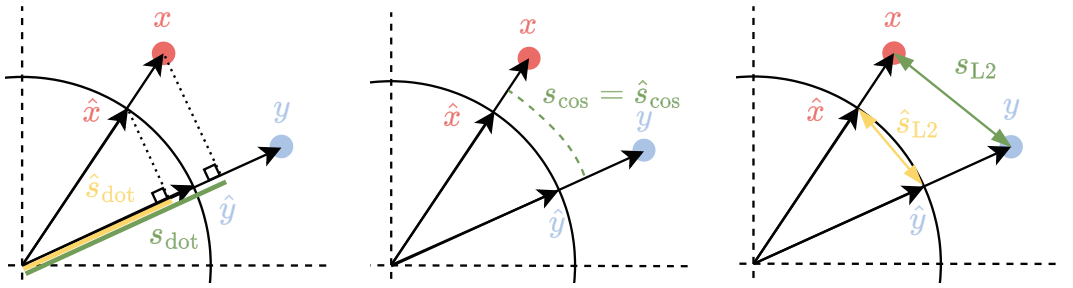
Feature extractors can be trained using only image-level supervision by adding a simple classifier on top of the feature extractor. This classifier typically comprises a linear neural network layer (with bias) that maps the feature vector x to a logits vector z of dimensionality equal to the number of classes in the training data. The logits are then passed through a softmax function to obtain a probability distribution over the classes. Finally, cross-entropy loss is computed between the predicted probability and the true labels.

$$\begin{aligned} z &= Wx + b \\ p &= \text{softmax}(z) \end{aligned} \tag{2.12}$$

To compare features during testing, three main similarity measures are used: *Dot product*(eq. 2.13), *cosine similarity*(eq. 2.14) and *euclidian distance*(eq. 2.15). Compared to the dot product, the cosine similarity is independent of the magnitude of the vectors

as it is normalized by the product of the magnitudes, i.e. for already normalized vectors the two measures are equal. While Euclidian distance applied to normalized vectors becomes more similar to cosine similarity, they are not equal. Figure 2.2 shows the different distance measures relative to a unit circle.

$$\sum_i x_i \cdot y_i \qquad (2.13) \qquad \frac{\sum_i x_i \cdot y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} \qquad (2.14) \qquad \sqrt{\sum_i (x_i - y_i)^2} \qquad (2.15)$$



(a) Dot Product (b) Cosine Similarity (c) Euclidean Distance (L2)

Figure 2.2: Different similarity measures. x and y denote points in feature space while $\hat{x}$ and $\hat{y}$ denote their normalized vectors.

Normalization of features is generally accepted to boost performance during testing [69], but the *linear* classifier described above uses an (unnormalized) dot product. This influence of the vector magnitudes creates a discrepancy between training and test objectives.

The classifier can be modified to optimize cosine similarity instead, sometimes referred to as *cosine classifer* [69]. However, trivially applying normalization to the feature vector x and weight vector w often fails to converge due to the limited range of the cosine function (-1 to 1). This results in a loss that cannot effectively distinguish between well-separated and non-separated samples. Consequently, a scale factor is introduced to increase the range of the cosine similarity output before passing it to the softmax function. This helps the model to better differentiate between classes during training. The scaling factor can be automatically learned through back-propagation.

$$z = s \cdot \cos(\theta) = s \cdot \frac{w}{||w||} \cdot \frac{x}{||x||} \qquad (2.16)$$

Margin penalties aim to improve intra-class compactness and inter-class distance by improving the decision boundary. For example, *ArcFace* [18] can easily be applied to the above cosine classifier. It adds a fixed margin penalty to the vector angle theta

for the true class. The added penalty forces the model to further decrease the angle between feature vectors and the class center further, leading to a less ambiguous decision boundary.

2.4 Measuring Mutual Information

Calculating mutual information for two random variables X and Z is intractable in general, and estimation based on samples is challenging [47]. Additionally, one also wants to be able to optimize based on mutual information, e.g., maximize MI between learned representations and task-specific information or to upper bound MI to limit the capacity of a representation [47].

Special cases allow for the calculation of MI: If the random variables are discrete, like in the original information bottleneck[63], or with explicit knowledge about their underlying distributions.

Upper bounds on MI require knowledge of the conditional distribution $p(y|x)$, which is the case if y is a stochastic representation. Using a variational approximation $q(y)$ of the intractable marginal $p(y) = \int dx\ p(x)p(y|x)$ a tractable bound can be defined using the KL divergence:

$$I(X;Y) \leq E_{p(x)}[\mathrm{KL}(p(y|x)||q(y))]$$

For example, the *Deep Variational Information Bottleneck* [2] uses this upper bound to reduce irrelevant information from input data about a target task in a learned representation.

Lower bounds on MI are possible without knowledge of the conditional distributions. Multiple estimators exist, such as NWJ [42] or NCE [65]. As a representative of similar methods, we will discuss and analyze MINE [8] as it has already been applied to the information bottleneck. Section 2.4.2 explains MINE in detail.

[53] propose a matrix-based functional based on Rényis α-order entropy. It has also been applied to the information bottleneck and is described in section 2.4.3.

2.4.1 Relationship of Cross-Entropy and Mutual Information

Estimating MI with the goal of maximization is not always necessary. Consider a standard supervised learning task with an input X, label information Y, and a prediction $\hat{Y}$. The goal is to align Y and $\hat{Y}$. This can be achieved using a loss based on the KL divergence (equation 2.8). In practice, the Cross-Entropy (CE) is used as a loss instead.

$$H_{\mathrm{CE}}(P,Q) = -\sum_{x\sim\mathcal{X}} p(x)\log q(x) \tag{2.17}$$

The KL divergence can be defined based on CE. Note that the inherent entropy of the label data $H(\hat{Y})$ is not part of the optimization. Minimizing Cross-Entropy (CE) and KL divergence are thus equal optimization objectives.

$$D_{KL}(\hat{Y}||Y) = H_{\text{CE}}(\hat{Y}, Y) - H(\hat{Y}) \tag{2.18}$$

Additionally, consider an intermediary representation T like in the information bottleneck (equation 2.10) that is trained using a classifier network $q(\hat{Y}|T)$ and CE loss. The IB objective requires the maximization of the MI between representation T and target Y. [10] prove that minimizing the conditional cross-entropy loss $H_{\text{CE}}(Y; \hat{Y}|T)$ is equivalent to maximizing $I(T; Y)$. This implicit maximization of mutual information has been commonly applied for the supervised information bottleneck [2, 81].

2.4.2 Mutual Information Neural Estimation (MINE)[8]

What makes MINE possible is the usage of the Donsker-Varadhan (DV) lower bound [21] on the KL-divergence. Specifically, for two distributions p_x and q_x over X with finite KL divergence, the following holds over all bounded functions $f : \mathcal{X} \to \mathbb{R}$:

$$D_{\text{KL}}(p_x||p_y) \geq \mathop{\mathbb{E}}_{x\sim p_x}[f(x)] - ln \mathop{\mathbb{E}}_{x\sim q_x}[e^{f(x)}] \tag{2.19}$$

The DV bound can then be used to construct a bound on mutual information by using its KL divergence-based definition:

$$\begin{aligned} I(X, Z) &= D_{KL}(P_{XZ}||P_X \times P_Z) \\ I(X, Z) &\geq \mathop{\mathbb{E}}_{(x,y)\sim p_{xy}}[f(x, y)] - \ln \mathop{\mathbb{E}}_{(x',y')\sim p_x \times p_y}[e^{f(x',y')}] \end{aligned} \tag{2.20}$$

MINE [8] now introduce a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that is parameterized by a neural network. The final objective of training the neural network f is to maximize:

$$\sup_f \frac{1}{N}\sum_{i=1}^{N} f(x_i, y_i) - \ln \frac{1}{N}\sum_{i=1}^{N} e^{f(x'_i, y'_i)} \tag{2.21}$$

During training, the expectations in the above equation are evaluated using samples (x_i, y_i) drawn from p_{xy} and (x'_i, y'_i) drawn from $p_x \times p_y$. In practice, only the joint distribution, i.e., several pairs $(x, z) \sim p_{XZ}$ in a mini-batch, are needed as the samples of the marginals can be obtained from the joint distribution by dropping either side of the pair and shuffling along the batch axis.

[8] also presents an alternative definition termed MINE-f. It is based on the f-divergence representation proposed in [40, 42]. For the same conditions as in equation 2.19:

$$D_{\mathrm{KL}}(p_x||p_y) \geq \mathop{\mathbb{E}}_{x\sim p_x}[f(x)] - \mathop{\mathbb{E}}_{x\sim q_x}[e^{f(x)-1}] \tag{2.22}$$

The right-hand side of the f-divergence bound is smaller than the Donsker-Varadhan bound (eq. 2.19), which leads to a looser bound numerically, although both bounds are tight asymptotically [8].

However, compared to MINE, MINE-f has the advantage that applying SGD in a mini-batch setting produces unbiased estimates of the gradient of the full batch. For MINE, the gradient estimate for f_Θ (Θ being the parameters of the neural network) is as follows:

$$\widehat{G}_B = \mathbb{E}_B\left[\nabla_\theta f_\theta\right] - \frac{\mathbb{E}_B\left[\nabla_\theta f_\theta\, e^{f_\theta}\right]}{\mathbb{E}_B\left[e^{f_\theta}\right]} \tag{2.23}$$

The expectations of the second term are over samples of the minibatch B which leads to a biased estimate of the gradient of the full batch. Replacing the denominator in the second term with an exponential moving average reduces the bias and improves performance [8].

2.4.3 Matrix based Rényi's α-order Entropy

Rényi's α-order entropy [52] is a one-parameter generalization of Shannon entropy. Given a discrete random variable $X \sim p$, it is defined as:

$$\mathbf{H}_\alpha(X) = \frac{1}{1-\alpha}\log\int_{x\in\mathcal{X}} f^\alpha(x)dx \tag{2.24}$$

In the limit of $\alpha \to 1$ (from above), the α-order entropy is equal to the Shannon entropy. The application of Rényi's α-order entropy to machine learning is hindered by the difficulty of estimating probability density functions for high-dimensional and continuous cases.

[53] introduce a matrix-based functional that does not require probability density function estimation while exhibiting similar properties to Rènyi's α-order entropy. The data gets projected to a reproducing kernel Hilbert space and the Gram matrix is calculated. The functional is then defined in terms of the normalized Eigenspectrum of the matrix.

Given N samples of a random vector x, i.e. samples in a minibatch, the entropy of x can be defined as:

$$\mathbf{H}_\alpha(A_x) = \frac{1}{1-\alpha}\log_2(\mathrm{tr}(A_x^\alpha)) = \frac{1}{1-\alpha}\log_2\left(\sum_{i=1}^{N}\lambda_i(A_x)^\alpha\right) \tag{2.25}$$

'tr' denotes the trace of a matrix, i.e. the sum of its main diagonal. A_x is the normalized version ($A_x = K_x/\mathrm{tr}(K_x)$) of the gram matrix $K_x \in \mathbb{R}^{N\times N}$ with Gaussian kernel κ ($K_x(m,n) = \kappa(x^m, x^n)$). $\lambda_i(A_x)$ denotes the i-th eigenvalue of A_x.

For N pairs of samples of random vectors x and y, given another corresponding normalized Gram matrix $A_y \in \mathbb{R}^{\mathbb{N}\times\mathbb{N}}$ for y, the joint entropy can be defined as:

$$\mathbf{H}_\alpha(A_x, A_y) = H_\alpha\left(\frac{A_x \circ A_y}{\mathrm{tr}(A_x \circ A_y)}\right) \tag{2.26}$$

where $\circ$ denotes the element-wise product and tr denotes the trace of the matrix, i.e. the sum of its main diagonal. Analog to Shannon (see section 2.1), the Rényi's α-order mutual information can then be defined as:

$$\mathbf{I}_\alpha(X;Y) = H_\alpha(A_x) + H_\alpha(A_y) - H_\alpha(A_x, A_y) \tag{2.27}$$

The kernel size σ is an important hyperparameter for the matrix-based functional. It is commonly set heuristicly, for example, using *Silverman's rule of thumb* [57] or based on the distances between all pairwise data points. $10 - 20\%$ of the total range of the distances is commonly used [56]. [74] point out that unsupervised rules of thumbs often fail for high dimensional data. They instead propose an optimality criterion based on *kernel alignment loss* [17].

2.4.4 Limitations

As pointed out in [47], the estimates of MINE "are neither an upper or lower bound on MI" [47], due to estimates being constructed from finite samples. Specifically, its high variance can be traced back to the expectation of the exponential in the DV-bound in equation 2.19.

$$\mathop{\mathbb{E}}_{x\sim q_x}[e^{f(x)}]$$

The expectation can be overpowered by extremely rare events that are not observed by sampling from q_x. [65] point out that their bound becomes loose when the mutual information is higher than $ln(N)$. The results presented for MINE in [8] seemingly do not suffer from this. However, [38] shows that the confidence claim in [8] is wrong due to an invalid derivation of an equation. They formally prove further that "any distribution-free high-confidence lower bound on mutual information cannot be larger than $O(\ln N)$ where N is the number of samples" [38]. In support of the proof, [47] empirically tested multiple MI estimators and found that none of their compared estimators exhibits both low variance and good estimates.

The matrix-based Rényi α-order functional introduced in [53] is limited to the mutual information calculation between two variables. This limits its usefulness for analyzing CNN networks as they contain multiple filters per layer. Therefore, an input is represented by many feature maps, which requires a multivariate measurement.

[79, 80] define an extension for multivariate Rényi *alpha*-order joint entropy and mutual information. This approach does not scale well with the number of filters in a CNN. [74] propose an approach based on tensor kernels better suited for large-scale CNNs.

CHAPTER 3

Related Work

3.1 Multi-View Representation Learning

Canonical Correlation Analysis (CCA) [29] is an early and widely used method for multi-view learning. Features are learned by maximizing the correlation between a weighted linear combination of the original variables. This can be seen as projecting two sets of random variables into a low-dimensional subspace, simplifying the statistical dependence analysis process. The projected data points can then be used as representations for various predictive modeling tasks [70].

Various kernel-based extensions [5, 1] have been proposed to capture nonlinear and more complex relationships between the two views, e.g., by projecting the data to a Hilbert space before maximizing the correlation between projected points. These methods are hard to scale for large training data, and the resulting representations are kernel dependent.

[3] improves kernel CCA by using deep neural networks to map the two data views to new representations for which correlation is maximized. [71] replace the standard fully connected neural network with an autoencoder. Their reconstruction objective allows for a configurable trade-off between the information captured by the individual representations and the information captured by the relationship between views.

[39] propose an approach using autoencoders without resorting to CCA for the creation of correlated representations. Instead, the shared latent representation is extracted by a single network while the reconstruction is performed by a separate network for each view.

CCA based methods have a critical flaw when used for (multi-view) representation learning. Correlation maximization is a distinct step before predictive modeling, which prohibits using label information for feature learning. [70].

The information bottleneck principle has gained attention by explicitly modeling the trade-off between predictive performance and the complexity of learned representations based on information theory. [75] use it in the context of supervised multi-view representation learning. They construct a joint latent representation using a linear combination of individual views. The projection matrices are learned using the IB principle. This approach is limited to linear relationships as with the traditional CCA based methods.

[70] employ deep neural networks to model non-linear relationships. Given two views X_1 and X_2 and label information Y, an individual representation for each view is learned. Additionally, a second neural network fuses the individual representations into a joined representation and the information bottleneck objective is defined as:

$$\begin{aligned} \max_{Z,Z_1,Z_2} \; & I(Y,Z) - \alpha I(X_1,Z_1) - \beta I(X_2,Z_2) \\ & \text{s.t. } Z = f_\theta(Z_1,Z_2) \end{aligned} \tag{3.1}$$

Z_1 and Z_2 denote the individual representations, f_θ the neural network used to fuse the representations, and α and β are regularization parameters used to control the level of compression applied to the individual representations.

[24] apply the information bottleneck principle in an unsupervised multi-view setting. Conceptually, an information bottleneck facilitates discarding information not shared between the two views. Discarding non-shared information relies on the assumption that each view is sufficient for potential downstream tasks and that this task-relevant information is the same across each view [82].

More formally, given two images x_1, x_2, that represent the two views and a label y, assuming the two views contain the same predictive information about the label, a representation z also has the same predictive information about the label if it retains all information shared between the two views. The two views are thus mutually redundant for the prediction of y. Take a representation z_1 of v_1 for which the above property holds, by factorizing the mutual information between v_1 and z_1, the authors identify two components:

$$I(\mathbf{v}_1;\mathbf{z}_1) = \underbrace{I(\mathbf{v}_1;\mathbf{z}_1 \mid \mathbf{v}_2)}_{\text{superfluous information}} + \underbrace{I(\mathbf{v}_2;\mathbf{z}_1)}_{\text{predictive information for } \mathbf{v}_2} \tag{3.2}$$

The first term describes the information in v_1 that is not also contained in v_2 (which means it is not predictable by observing v_2). Since mutual redundancy is assumed, this information can be discarded. The overall loss function for both views is[1]:

$$L = -I(z_1;z_2) + \beta D_{SKL}(p_\theta(z_1|v_1)||p_\psi(z_2|v_2)) \tag{3.3}$$

[1] $D_{SKL}(A||B)$ is the average of $D_{KL}(A||B)$ and $D_{KL}(B||A)$

To maximize $I(z_1; z_2)$ a lower bound like NCE [65] is used. Direct calculation of the D_{SKL} term is possible as a normal distribution model for both stochastic encoders.

[67] similarly presents an unsupervised multi-view representation learning method based on the information bottleneck principle. Contrary to [24], they define three types of representations: A view-specific representation S, a shared representation H, and the resulting "multi-view IB representation" Z. A reconstruction-based method obtains the representations S and H. For the shared representation, the method assumes that there is an underlying latent representation from which the individual views originate. All views are encoded into a shared space, yielding H, while the individual views are then reconstructed from H using individual reconstruction networks. According to [67], this "could reveal the common latent structure shared by different views" [67]. The view-specific representation S aims to preserve *complementary* information. This is different from the multi-view assumption used in [24] that the information not present in all views (i.e. the complementary information) is excessive for a downstream task. Deciding which information is complementary and which is shared is "difficult" [68], instead a view-specific autoencoder (also optimized with reconstruction loss) is employed to learn a representation for each view independently, guaranteeing that their private information is contained. The overall S is constructed by concatenating the individual representations, and the final objective is:

$$\max_{\mathbf{Z}} I(\mathbf{Z}, \mathbf{H}) + I(\mathbf{Z}, \mathbf{S}) - \sum_{v=1}^{V} \beta_v I\left(\mathbf{Z}, \mathbf{X}^{(v)}\right) \tag{3.4}$$

[70, 24, 67] all use variational approximation to obtain an optimizable bound on the information bottleneck objective. However, bound tightness is hard to guarantee.

[81] instead use Rényi's α-order matrix-based entropy to directly optimize the IB objective. The model architecture and objective are the same as in the original multi-view information bottleneck using deep learning [70], but generalized to multiple views:

$$\begin{aligned} \max_{Z, Z_i, \ldots, Z_k} & I(Y, Z) - \sum_{i=1}^{k} \beta_i I(X_i; Z_i) \\ \text{s.t. } & Z = f_\theta(Z_1, \ldots, Z_k) \end{aligned} \tag{3.5}$$

3.2 Image Retrieval

Global features allow for efficient image matching but lose spatial information about the image content. Without spatiality, performance-improving methods for retrieval, like geometric verification, cannot be applied to these features. Local features are associated with specific image regions and ideally, describe keypoints for an image. Multiple local features exist per image, and a many-to-many matching is performed during retrieval. Two paradigms for extracting local features exist [14]:

Detect-then-describe first proposes image regions (i.e. *image patches*), and then extracts local features for these regions. For example, [50] use sliding windows, [83] a spatial pyramid, and [51] use a neural network to propose image regions. The local descriptors for a given image region are typically extracted by CNNs.

Describe-then-detect first applies a CNN and later detects which areas of the feature maps are to be used as local descriptors. [58] use local maxima in the feature maps as candidates for keypoints. [41, 12] use an auxiliary attention network to calculate attention scores over the spatial extent of the CNN feature maps. Locations with high scores are then used as keypoints. Similarly, (Visual)-Transformer-based architectures implicitly calculate attention scores that can be used for detection [23, 72].

Using many local features per image is expensive for storage and matching computation. Hence, re-ranking is used extensively for image retrieval [14]. It allows for local features to be extracted on the fly and only for a small subset of images as determined by the global feature similarity. E.g. [43, 45, 6, 41, 12] all use RANSAC[11] to perform geometric verification on the local feature matches. A detailed description of this process can be found in section 2.3.3. RANSAC is an iterative algorithm that requires random restarts, which makes the re-ranking process computationally expensive, even on the smaller subset of images.

RRT [62] propose an alternative based on transformer neural networks that directly computes image similarity and is easily parallelizable. While RRT is faster than the traditional geometric verification process, all re-ranking-based methods have the downside of higher retrieval latency because they are inherently a two-step process. Several recent works present an alternative to re-ranking using local features:

[72] propose using mid-level features called “Super-features”, which are constructed by an iterative attention module. They note that local features are redundant as they are directly constructed from CNN feature maps, which cover overlapping image patches. Their attention module produces an ordered set of features, trained using a contrastive loss that matches the features across positive image pairs. An additional decorrelation loss encourages the feature set to be less redundant. Performance is competitive with DELG [12], but the method does not address the problems that re-ranking solves: Re-ranking using geometric verification is computationally more expensive, but storing all “Super-features” requires considerably more storage.

[55] propose a re-ranking process without using local features. Instead, re-ranking works by iteratively refining global representations. This is similar to *Query Expansion*, which also has been extensively applied to image retrieval [16, 27]. First, M candidate images and K of their nearest neighbors are retrieved based on global similarity. For each retrieved image, a refined descriptor is constructed using a weighted pooling of the descriptors from the K nearest neighbors and the descriptor of the database image itself. The global similarity score is used as the weight. Then, the refined descriptors of the candidate images are matched against the original query descriptor. The top K closest are used to construct a refined descriptor for the query image using max pooling. The final similarity score between database images and the query is an average of two scores: The similarity of the original query descriptor with the refined database images and the similarity between the refined query descriptor with the refined database images. Additionally, [55] also presents an improved pooling strategy when extracting global descriptors, which, even without the above-described re-ranking process, can outperform both DELG [12] and RRT [62] using only global descriptor retrieval.

[76] employ a single-stage retrieval process instead of re-ranking and focus on improving the image representation. They do this by aggregating both global and local features into a single descriptor. Local features are projected onto the global feature vector and for each feature vector, an orthogonal vector (to the global feature vector) is calculated. The orthogonal vector is the difference between the projected and original local feature vectors. The global feature and all orthogonal points are concatenated, pooled, and passed through a linear layer to produce a single final representation of the image. Using orthogonal projection has two goals: Capturing critical complementary information from the local features while removing redundant components to existing global information. Removing redundant information between global and local features is similar to our proposed method. However, we explicitly model the mutual information of global and local features, aiming to improve local feature representations for re-ranking. In comparison, [76] aims at providing a single improved representation.

CHAPTER 4

Solution Approach

We propose a re-ranking *Image Retrieval* system that explicitly models the mutual information relationship between global and local descriptors. Global (Z_g) and local (Z_{la}) descriptors should be maximally informative about the target task Y ($\max I(Z_g; Y), \max I(Z_{la}; Y)$). We train our retrieval model using image-level supervision where Y represents the labels of a classification task. Additionally, we penalize the mutual information between the descriptors to reduce redundant information ($\min I(Z_g; Z_{la})$). A hyperparameter β controls the degree of penalization. Figure 4.1 shows these relationships, and our overall training objective is to maximize:

$$I(Z_g; Y) + I(Z_{la}; Y) - \beta I(Z_g; Z_{la}) \tag{4.1}$$

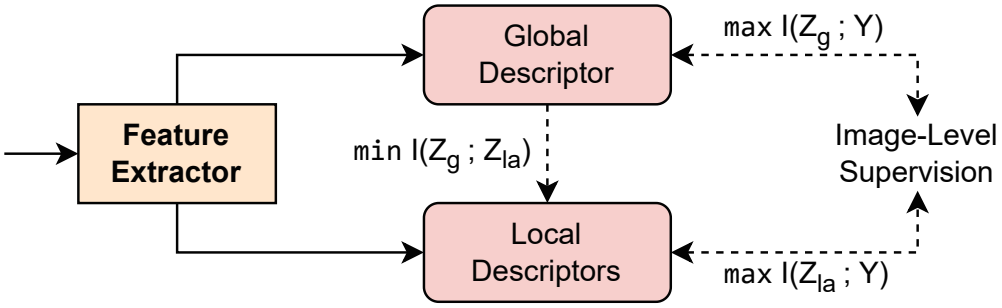


Figure 4.1: A schematic overview of the information-theoretic objectives for the proposed feature extractor. A feature extraction model produces global and local descriptors (red). During inference, these descriptors are used for image retrieval with re-ranking. During training several mutual information-based objectives act on global and local descriptors.

Our approach is agnostic to the underlying feature extractor as long as the system follows some basic requirements: It must (i) extract global and local descriptors for re-ranking

use, (ii) be based on a neural network, (iii) and be trainable end-to-end using image-level supervision.

We base our implementation on that of DELG [12], a popular image retrieval system that fulfills our requirements. It extracts image descriptors using a CNN with two heads. Both model heads are trained using a classification loss. The implementation of DELG[12] is available on GitHub[1] and written using *TensorFlow* [37]. For our experiments, we have re-implemented the model in *PyTorch* [4]. Additionally, our experiments use the implementation of RRT [62]. We use this model as an alternative to re-ranking based on geometric verification.

We summarize these two methods and discuss the differences in our implementation in section 4.1. Section 4.2 discusses the feature export for our experiments with RRT. Section 4.3 details how we measure and minimize mutual information.

4.1 Underlying Methods

4.1.1 Unifying Deep Local and Global Features for Image Search (DELG)

[12] introduces *DELG*, a model that simultaneously extracts both global and local features from a CNN backbone for image retrieval. Additionally, it supports dimensionality reduction of local features using an autoencoder and requires no post-processing steps. It can be trained end-to-end using two classification losses and a reconstruction loss with only image-level supervision. The complete model architecture is shown in figure 4.2.

ResNet-50 and ResNet-101 [28] are used as the CNN backbone in their experiments. Global features are extracted from the final layer of the CNN (*conv5*) and aggregated using Generalized Mean Pooling (GeM) [49].

$$g_{\text{pooled}} = \left(\frac{1}{HW} \sum_{h,w} g_{h,w}^{p} \right)^{\frac{1}{p}} \tag{4.2}$$

H and W refer to the spatial dimensions of the feature map. GeM is a generalization of average ($p = 1$) and max pooling ($p \to \infty$). The parameter p can be learned during training to effectively weight the contributions of features [12]. After pooling, *whitening* is applied to down-weigh co-occurrences of features in the pooled feature map. Following [26], this is implemented using a fully connected layer after pooling.

A *cosine classifier* with *ArcFace* margin is used to train the feature extractor (see section 2.3.4). Local features are extracted from the penultimate layer of the backbone (*conv4*). An attention module selects relevant regions of the image.

[1] https://github.com/tensorflow/models/tree/master/research/delf

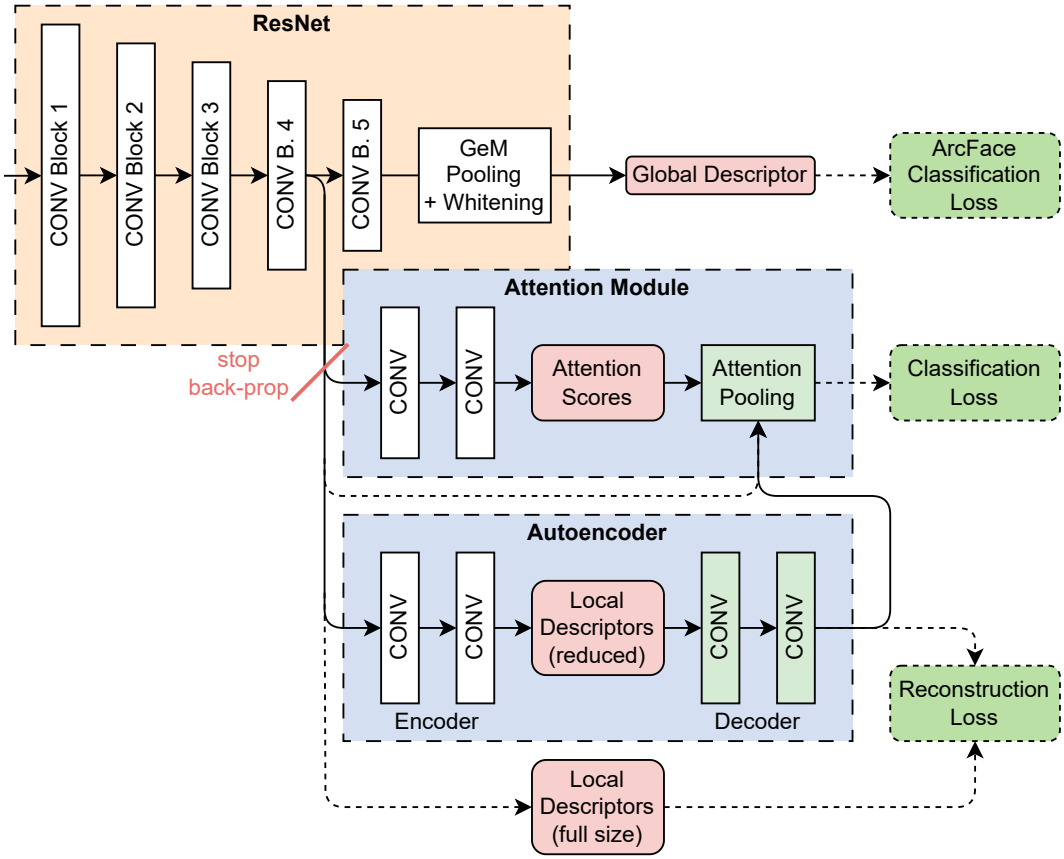


Figure 4.2: Model architecture of DELG. The loss components, the decoder part from the autoencoder, and the attention pooling component are only needed during training (green). The global descriptor, attention scores, and local descriptors are the outputs of the model during inference (red). The autoencoder is optional, so the local descriptors are simply the output of the ResNet *conv4* layer. Depending on whether the autoencoder is used, the reconstructed local descriptors or the original local features are passed to the Attention Module for pooling.

Attention Module

The attention module was introduced in earlier work [41], where it had to be separately trained after training the feature extractor itself. In DELG, it can be jointly trained with the global feature extractor. However, stopping the back-propagation of its corresponding loss gradients into the network backbone is essential as it disturbs the feature representations the CNN learns [12].

The attention module consists of two convolutional layers with 1×1 filters and stride $= 1$. The first layer halves the channels and is followed by batchnorm [30] and a ReLu activation function. The second layer has only a single output channel and is followed by a Softmax

activation to produce a relevance score for each local feature. This score is later used to select which local features are relevant to the depicted instance in the image. The local features are pooled based on their score after normalization to train the attention module. This is done by simply multiplying the feature vectors with the score and then taking the mean value across the width and height of the feature map.

Autoencoder

The autoencoder is an optional component for dimensionality reduction of local features. As with the attention module, it can be jointly trained, and the back-propagation of its corresponding reconstruction loss should be stopped before the CNN backbone. The autoencoder eliminates the need for post-processing steps to reduce dimensionality like PCA.

Similarly to the attention module, the autoencoder consists of two convolutional layers with 1×1 filters and stride $= 1$. The first layer has output channels equal to the desired (reduced) local descriptor dimension. The second layer has output channels equal to the original feature dimension taken from the CNN backbone and is followed by a ReLu activation. The autoencoder is trained using MSE loss, and a hyperparameter controls its contribution to the overall loss.

Retrieval

The global features are normalized for retrieval, and their similarity is calculated based on their dot product (see section 2.3.4). The top 100 images are then selected for re-ranking. The re-ranking process itself uses *RANSAC* [11] to perform geometric verification:

First, local features are paired by L2 distance, and their position in the feature map is recorded . Second, an affine transformation is fitted on the spatial locations of all feature pairs. Third, based on a threshold value, the fitted *RANSAC* model produces several pairs of inliers. The number of inliers is then used as the final score of the re-ranking process.

Geometric verification has several downsides: It requires rigid objects, e.g., a human changing poses between images, would change the relative location of detected keypoints, and is sensitive to significant viewpoint changes. It is also a computationally expensive iterative process that requires many local features.

4.1.2 Reranking Transformer

[62] propose *Reranking Transformers (RRTs)* as an alternative to the expensive geometric verification step. It is based on the BERT [19] transformer architecture and directly calculates a similarity score between image pairs. The input sequence to the transformer includes global and local descriptors of both images, along with special tokens to distinguish between images and positional and segment embeddings.

RRTs are lightweight (2.2 million parameters), easily parallelizable, and can be optimized jointly with a feature extractor to improve retrieval accuracy further. Figure 4.3 illustrates an overview of the architecture. The authors provide their implementation on GitHub [2].

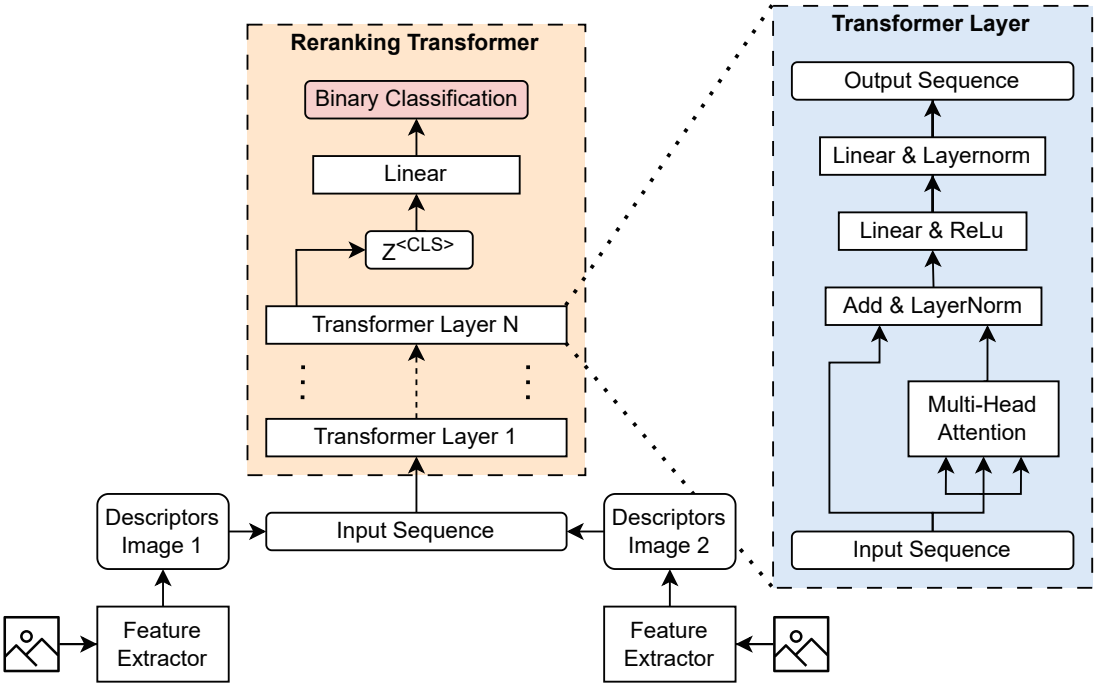


Figure 4.3: Model architecture of RRT.

Input Sequence

Transformers convert an input sequence comprising of tokens to an output sequence ('sequence transduction model'). Following the overall architecture of BERT [19], the input sequence in RRT for an image pair $(\mathbf{I}, \bar{\mathbf{I}})$ is defined as follows:

$$\mathbf{X}(\mathbf{I}, \bar{\mathbf{I}}) := [\langle \text{CLS} \rangle; f_g(\mathbf{x}_g); f_l(\mathbf{x}_{l,1}); \cdots; f_l(\mathbf{x}_{l,L}); \langle \text{SEP} \rangle; \bar{f}_g(\bar{\mathbf{x}}_g); \bar{f}_l(\bar{\mathbf{x}}_{l,1}); \cdots; \bar{f}_l(\bar{\mathbf{x}}_{l,L})] \tag{4.3}$$

[2] https://github.com/uvavision/RerankingTransformer

The special token ⟨CLS⟩ is used to retrieve the output signal and ⟨SEP⟩ separates the tokens of the two images.

To distinguish between global and local descriptors and which image the descriptors belong to, the one-dimensional segment embeddings α, $\overline{\alpha}$, β, $\overline{\beta}$ are used.

For global descriptors, the final input representation is as follows:

$$\begin{aligned} f_g\left(\mathbf{x}_g\right) &:= \mathbf{x}_g + \alpha \\ \bar{f}_g\left(\overline{\mathbf{x}}_g\right) &:= \overline{\mathbf{x}}_g + \bar{\alpha} \end{aligned} \tag{4.4}$$

Each local descriptor has an associated image scale and position. Two additional embeddings are used to encode this information into the sequence:

The positional embedding φ from [13] is applied to the local descriptor positions $p_{l,i}$ and $\overline{p}_{l,i}$. It is based on sinusoidal functions like in the original transformer [66] but has been generalized to work on the two-dimensional case of images. $\frac{d}{2}$ sine and cosine functions with different frequencies are concatenated for each spatial axis to construct a d dimensional embedding.

The descriptor image scale uses a simple linear embedding ψ. As input, an integer indexing pre-defined image scale is passed. In practice, the segment and linear embedding for the scale behave the same: An integer indexes a learned embedding vector. The final input representation for local descriptors is as follows:

$$\begin{aligned} f_l\left(\mathbf{x}_{l,i}\right) &:= \mathbf{x}_{l,i} + \varphi\left(\mathbf{p}_{l,i}\right) + \psi\left(s_{l,i}\right) + \beta \\ \bar{f}_l\left(\overline{\mathbf{x}}_{l,i}\right) &:= \overline{\mathbf{x}}_{l,i} + \varphi\left(\overline{\mathbf{p}}_{l,i}\right) + \psi\left(\bar{s}_{l,i}\right) + \bar{\beta} \end{aligned} \tag{4.5}$$

Model

RRT uses a multi-layer bi-directional transformer architecture based on the original transformer architecture of [66]. Each transformer layer is comprised of Multi-Head Attention (MHA), two linear layers with an activation function, and normalization functions:

$$\begin{aligned} t_0(Z) &:= Z + \mathrm{MHA}(Z, Z, Z) \\ t_1(Z) &:= \mathrm{LayerNorm}(t_0(Z)) \\ t_2(Z) &:= \mathrm{Linear}(t_1(Z)) \\ t_3(Z) &:= \mathrm{ReLU}(t_2(Z)) \\ t_4(Z) &:= \mathrm{Linear}(t_3(Z)) \\ t_5(Z) &:= \mathrm{LayerNorm}(t_4(Z)) \end{aligned} \tag{4.6}$$

Note that for the multi-head attention $Q = K = V$, i.e., only a single set of vectors is used. Applying a transformer layer to the input sequence produces a new sequence of vectors of the same length:

$$\begin{aligned} Z_0 &= \mathbf{X}(\mathbf{I}, \bar{\mathbf{I}}) \\ Z_{i+1} &= t_5(Z_i) \end{aligned} \tag{4.7}$$

A final linear layer projects the ⟨CLS⟩ token of the last transformer layer to a single dimension. After being passed through an activation function (sigmoid), it produces a logit scalar. The model is trained using a Binary Cross-Entropy (BCE) loss function to predict whether two images represent the same object or scene:

$$\begin{aligned} \text{logit} &= \text{Sigmoid}(\text{Linear}(Z_C^{\langle \text{CLS} \rangle})) \\ \text{E}(\mathbf{I}, \bar{\mathbf{I}}) &= \text{BCE}(\text{logit}, \text{EQ}(\mathbf{I}, \bar{\mathbf{I}})) \\ \text{EQ}(\mathbf{I}, \bar{\mathbf{I}}) &:= \begin{cases} 1, & \text{if } \mathbf{I} \text{ and } \bar{\mathbf{I}} \text{ depict the same entity} \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{4.8}$$

4.1.3 Implementation Differences

We simplify the implementation of DELG[12] to ease with the evaluation for our experiments. We do not use an autoencoder to reduce the dimensionality of local features. This simplifies the training procedure, as no reconstruction loss (and corresponding hyperparameter) for the autoencoder is needed. Following the experimental setup in RRT[62] for the SOP dataset, we instead use an additional convolutional layer to project the conv4 output of ResNet to the desired number of dimensions. The reduced output is then passed to the attention module. We do not use whitening for global features and use average instead of GeM pooling.

DELG uses an image pyramid at inference time to produce multi-scale representations. For global features, the scales are average-pooled; for local features, the top attention scores are selected across the scales. Selecting the top scores is done greedily to avoid selecting multiple features at the same image location across different scales. Depending on the experiment, they use 3 or 7 image scales. For simplicity, we do not use an image pyramid in our experiments.

Figure 4.4 shows an overview of our model architecture.

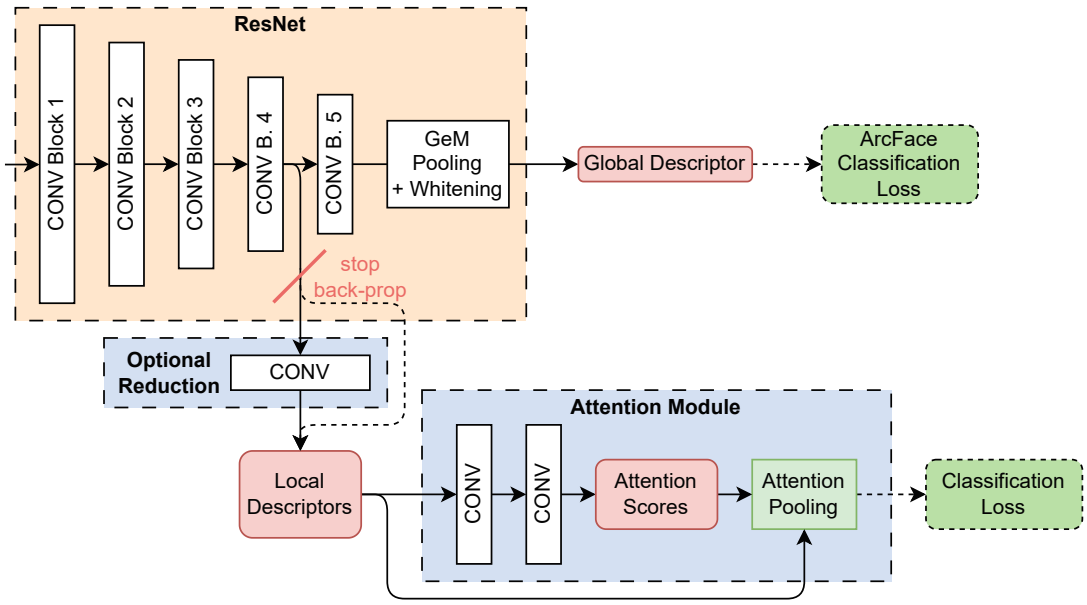


Figure 4.4: Model architecture without MI estimation. It is a simplified version of the architecture presented in [12] that is shown in figure 4.2. Instead of an autoencoder, we use a single convolution with a 1×1 kernel and stride $= 1$ to (optionally) reduce the dimension of local features. The rest of the implementation is identical. The loss components, the decoder part from the autoencoder, and the attention pooling component are only needed during training (green). The global descriptor, attention scores, and local descriptors are the outputs of the model during inference (red).

4.2 Feature Export for Re-Ranking with RRT

[62] perform re-ranking experiments using RRT on pretrained descriptors extracted from DELG. To be able to evaluate an alternative method to geometric verification, we built a compatible feature export to reuse the publicly available RRT model[3] with minor modifications. Exporting descriptors prohibits the usage of data augmentation during the training of RRT. This is a limitation of our experimental setup as we use the provided implementation of RRT. As demonstrated by the experiments of [62] on SOP, a joint model would perform better.

The export produces two folders, one for training and one for testing. Both contain a descriptors folder and a file containing the nearest neighbors. The descriptors folder contains two files for each image, one containing just the global descriptor and one containing all local descriptors with corresponding positions, scales, and attention scores. Descriptor related data is stored as *NumPy*[4] arrays in *pickle* format[5]. Local descriptors

[3] https://github.com/uvavision/RerankingTransformer
[4] https://numpy.org/
[5] https://docs.python.org/3/library/pickle.html

are exported in descending order by their attention scores. Index files for training, gallery, and query contain the filenames of all descriptors, image labels, and size information.

The nearest neighbor files contain a matrix of image IDs (0 to $N-1$ of the dataset split) that indicate the closest K images by global descriptor similarity.

Each image in the training set contains the closest $K = 100$ images. The test set contains the closest gallery images for each query image. For evaluation, the testing folder also contains a ground truth file that maps query image IDs to gallery image IDs considered positive instances.

4.3 Measurement and Minimization of Mutual Information

One core requirement for our work is the ability to measure mutual information between various parts of the model. Subsequently, it has to be possible to minimize mutual information between global and local features. We use the global descriptor after pooling and the local features after averaging them based on their attention score (see section 4.1.1). Using the attention-pooled local descriptor circumvents having to estimate between the global descriptor and all individual local descriptors. The overall loss function for our DELG-like model is:

$$L = L_g + L_l + \beta I(Z_g; Z_{la}) \tag{4.9}$$

L_g and L_l refer to the classification losses of the global and local feature paths, respectively. Cross-Entropy (CE) is used as the loss function which implicitly maximizes mutual information: $\min L_g \equiv \max I(Z_g; Y)$ and $\min L_l \equiv \max I(Z_{la}; Y)$. See 2.4.1 for details.

$I(Z_g; Z_{la})$ refers to the mutual information between the global feature (Z_g) and the attention-pooled version of the local features (Z_{la}). The hyperparameter β controls the degree of the penalization. Two methods measure and minimize $I(Z_g; Z_{la})$: Rényi's α-order matrix-based functional and MINE. The theoretical background of both methods is described in detail in section 2.4.

4.3.1 MINE

Figure 4.5 gives an overview of the different components of model training and the gradient paths. MINE requires an auxiliary network whose parameters must also be trained. Its training can be unstable, and its estimates exhibit a high variance (see section 2.4.4). Inspired by the approach in [31], we use a very low learning rate for more stable training but train the auxiliary network for multiple epochs whenever a mutual information estimate is required.

The overall procedure is shown in algorithm 4.1: First, the feature extraction model is trained, followed by MINE using a second optimizer. The second optimizer uses a lower learning rate, so the mutual information estimates can converge more slowly. During this phase, only the weights of the mine network are updated. The back-propagation of gradients into the feature extraction model is blocked. This prevents unstable mutual

information estimates from disturbing the learning process. During the training of the feature extraction model, the mutual information penalty as calculated by MINE is added to its loss function. The gradient is then back-propagated through the auxiliary network into the backbone.

Additionally, the gradients of MI estimation are never back-propagated via the global feature path as we do not want to affect the results of global retrieval. Our method aims to improve re-ranking performance and thus only affects the local features.

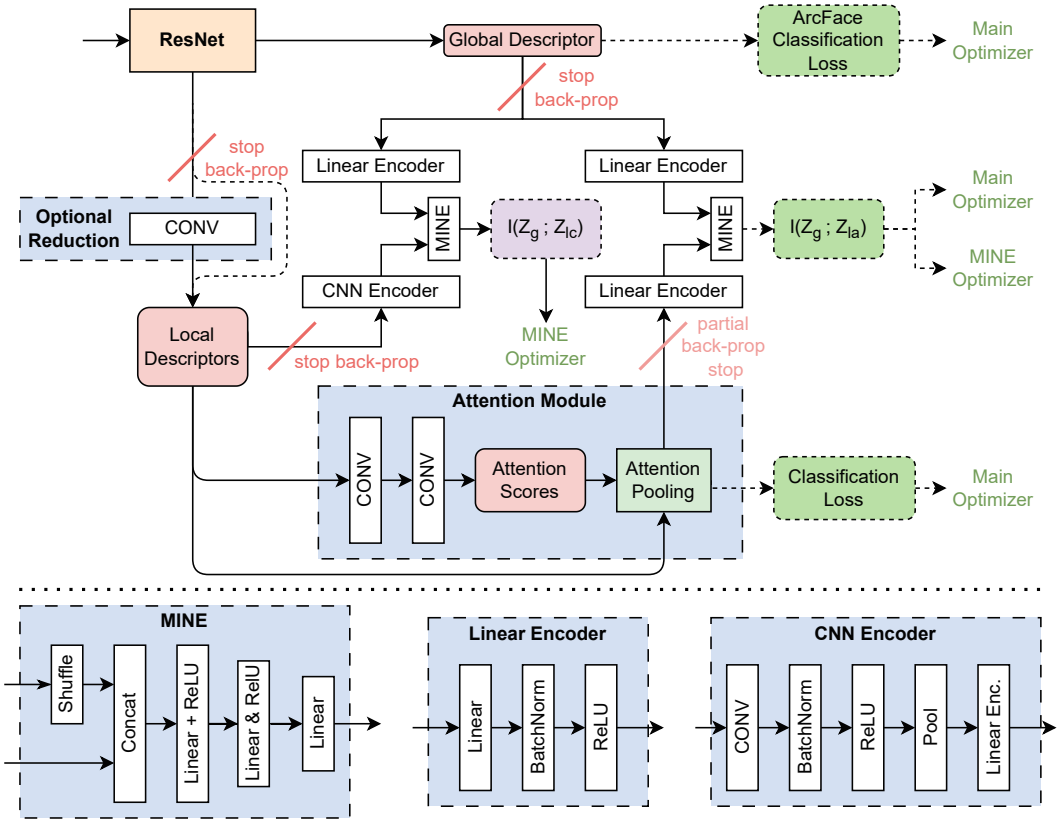


Figure 4.5: Our model architecture using MINE for estimation

All operations required to calculate the MI bound of MINE (equation 2.20 in section 2.4.2) are implemented in the autograd engine of *PyTorch*. This enables us to add the mutual information estimate $I(Z_g; Z_{la})$ to the overall loss for minimization. The gradients of the minimization objective will then correctly propagate back into the feature extractor. For training the MINE model itself, we maximize the MI estimate instead. The optimizers have the opposite goal regarding the mutual information estimates, which leads to a zig-zag pattern during training.

MINE places no restrictions on the dimensionality of its input random variables. However, we always use random variables of equal dimensions. If one has more, we use a linear

Algorithm 4.1: Model Training with MINE estimation

```
forall feature extractor training epochs do
    disable grad for feature extractor;
    forall MINE training epochs do
        forall batches do
            run feature extractor ;
            calculate MINE loss l_MINE ;
            backprop l_MINE ;
            step MINE specific optimizer (max) ;
        end
    end
    enable grad for feature extractor ;
    forall batches do
        run feature extractor ;
        calculate global arcface loss l_g;
        calculate classification loss for local features l_la ;
        estimate I(Z_g; Z_la) using mine network;
        L = L_g + L_la + β * I(Z_g; Z_la) ;
        backprop l step feature extractor specific optimizer (min) ;
    end
end
```

layer for projection before passing it to the MINE network.

Our setup with two optimizers allows us to add additional MINE networks to examine the mutual information relationship between different model parts. For example, we add a second network to measure $I(Z_g; Z_{lc})$ between the global feature and the local descriptors before attention pooling as ablation to study the effects of the attention mechanism. In this case, we first use an additional convolution layer to halve the spatial resolution while doubling the number of dimensions of the local descriptors before pooling and passing them to the linear encoder.

4.3.2 Matrix based Rényi's α-order Functional

[78] derive the differentiability of the matrix-based Rényi's α-order equations presented in section 2.4.3. However, in practice, the automatic differentiation engines of deep-learning frameworks make them directly applicable.

Following [81], we fixate the α-order hyperparameter to 1.01, and the kernel width σ is calculated based on the mean distance (L2 in feature space) of the k ($k = 10$) nearest samples. For each mini-batch, the average of the distance means is used. While the authors provide an open source implementation[6] of their work, please note that their σ

[6] https://github.com/archy666/MEIB

calculation is erroneous.

We do not use our Rényi-based implementation for auxiliary MI measurements, such as between the global descriptor and the local descriptors before attention pooling. CNNs produce multiple feature maps that require a multi-variate method for measuring MI. Extensions for multi-variante measurements exist (see section 2.4.4), but since MINE is already capable of performing this, we did not implement them.

At the beginning of training, the magnitude of the local feature attention scores is small (for details on the attention module, refer to section 4.1.1). Minimizing the mutual information between global and local features can easily overpower the classification signal and collapse the local features. To remedy the issue, we warm up the β parameter by setting it to 0 for a definable number of training steps and then linearly increasing it over a definable number of training steps until it has reached its target value. This is seemingly not a problem for the MINE estimator as the initial estimates' magnitude starts small and increases during training.

CHAPTER 5

Evaluation Methodology

We evaluate our model using standard benchmark datasets and standard metric definitions. Section 5.1 defines the used metrics and gives details on the datasets. We train our model with varying degrees for the mutual information penalty factor β between global and local descriptors. We evaluate two methods for estimating and minimizing mutual information: MINE [8] and Rényi's α-order matrix-based functional [53]. During training, we record additional MI measurements for ablations. Using the trained models, we evaluate two methods for re-ranking: *geometric verification* [12] and RRT [62]. Section 5.2 details the model hyperparameters and experiments performed. We log the results of experiments using the *Weights & Biases*[1] platform.

5.1 Evaluation

5.1.1 Metrics

Classification metrics like *Precision*, *Accuracy* and *Recall* do not take the position of the retrieved image into account. Arguably, having more relevant images up front for retrieval, recommender systems, and applications like search engines is crucial. Several ranking metrics are commonly used:

Precision at rank k (P@k)

Analog to classification *precision* but with a cutoff at rank k.

$$\text{P@k} = \frac{\text{\# of recommended items @k that are relevant}}{\text{\# of recommended items @k}} \tag{5.1}$$

[1] https://wandb.ai/site

Recall at rank k (R@k)

Analog to classification *recall* but with a cutoff at rank k. This metric is monotonically increasing with k.

$$\text{R@k} = \frac{\text{\# of recommended items @k that are relevant}}{\text{\# of relevant items in dataset}} \tag{5.2}$$

Note that the metric learning community does not use the definition of equation 5.2 which is used in nearly all other contexts. Instead, they chose to redefine it, but confusingly to keep the same name (see equation 5.3).

$$\text{R@k} = \begin{cases} 1, & \text{if } \textbf{any} \text{ of the recommended items @k is relevant} \\ 0, & \text{if none are relevant} \end{cases} \tag{5.3}$$

Average Precision at rank k (AP@k)

Summarizes both P@k and R@k for all ranks into a single metric. It is defined as the average of P@k values of those ranks with a new relevant item, i.e., when the recall increases, up to a cutoff at rank k. If no cutoff is provided, then k equals the position of the last relevant item, often denoted as AP@all or simply AP.

$$AP@k = \frac{1}{\text{\# of relevant items @k}} \sum_{i}^{k} P@i * R_i \tag{5.4}$$

$$R_i = \begin{cases} 1, & \text{if document } i \text{ is relevant} \\ 0, & \text{if document } i \text{ is not relevant} \end{cases}$$

During evaluation, the ranking metrics are calculated for each query image, and the mean value for all queries is reported. For P@k and R@k the mean is typically referred to by the same name. For AP@k the name mAP@k is used instead.

5.1.2 Datasets

We perform experiments on datasets also used by the baseline methods [12] and [62] that we compare our work to. The definitions of the used metrics can be found in section 5.1.1. Additionally, we use CIFAR-10 & CIFAR-100 for preliminary experiments. Table 5.1 gives an overview of the scale of the used datasets.

CIFAR-10 & CIFAR-100

CIFAR-10 and CIFAR-100 [33] consist of 60k images initially provided as a training set of 50k images and a test set of 10k images. Each class contains the same number of

Dataset	# Images				# Classes
	Total	Training	Query	Gallery	
CIFAR-10	60k	5k	1k	59k	10
CIFAR-100	60k	5k	1k	59k	100
SOP	120k	59k	61k	61k	22k
GLD 'v2-clean'	1580k	1580k	-	-	81k
ROxf	-	-	70	4,9k	-
RPar	-	-	70	6,3k	-

Table 5.1: Overview of the used datasets

images, and the datasets contain 10 and 100 classes each. For retrieval, we follow the standard setup of hash-based retrieval methods that test on this dataset (e.g., described in [61]): 1k(100 per class for CIFAR-10/10 per class for CIFAR-100) images are used as the query set and the rest are used as the gallery set. The training set is sampled from the gallery set and contains 5k images (500/50 per class). Additionally, for reporting metrics during training, we sample a validation set of 1k images from the gallery set that does not overlap with the training set.

Google Landmarks Dataset v2 (GLDv2)

Google Landmarks Dataset v2 [73] is a large-scale benchmark for (landmark) image retrieval, containing over five million images across 200k classes. The dataset is split into a training, query (called 'test'), and gallery (called 'index') subset. The training subset contains 4,132,914 images with 203,094 classes. The query subset contains 117,577 images with ground truth annotations. The gallery subset contains 761,757 images. The classes in the training and gallery subset are not disjoint, which makes the dataset a closed-set classification problem.

[77] provide a cleaned version ('v2-clean') of the training split containing 1,580,470 images with 81,313 classes. Figure 5.1a shows the distribution of the number of training images per class. We use this split to train the feature extraction model, but for reporting metrics during training, we perform an additional 95/5 split on the provided split. [62] also uses GLDv2 in their experiments for RRT. The model is relatively lightweight, so they train it with an even smaller subset. They sample 12k landmarks, each with at least ten images. Each landmark is capped to at most 500 images, which results in about 20% size of the 'v2-clean' split. As the authors provide the image IDs of the split, we use the same split ('v2-rrt') to train RRT for a fair comparison.

Revisited Oxford / Paris

Revisited Oxford ($\mathcal{R}$Oxf) and Revisited Paris ($\mathcal{R}$Par) [48] is a standard benchmark for image retrieval and is an improved evaluation protocol for the *Oxford* [44] and *Paris* [46] datasets. The datasets are only used for evaluation, and the model is trained on GLDv2.

$\mathcal{R}$Oxf contains 4,993 gallery images and $\mathcal{R}$Par contains 6,322. Both use 70 images depicting landmarks of Oxford and Paris, respectively, as a query set.

The evaluation protocol allows for multiple difficulty levels (*easy*, *medium*, *hard*) in retrieval by changing which gallery images are considered as ground truth. We use the *medium* difficulty for all our experiments. Following this protocol, each query image is cropped by a provided bounding box for retrieval, and mAP is used as a metric.

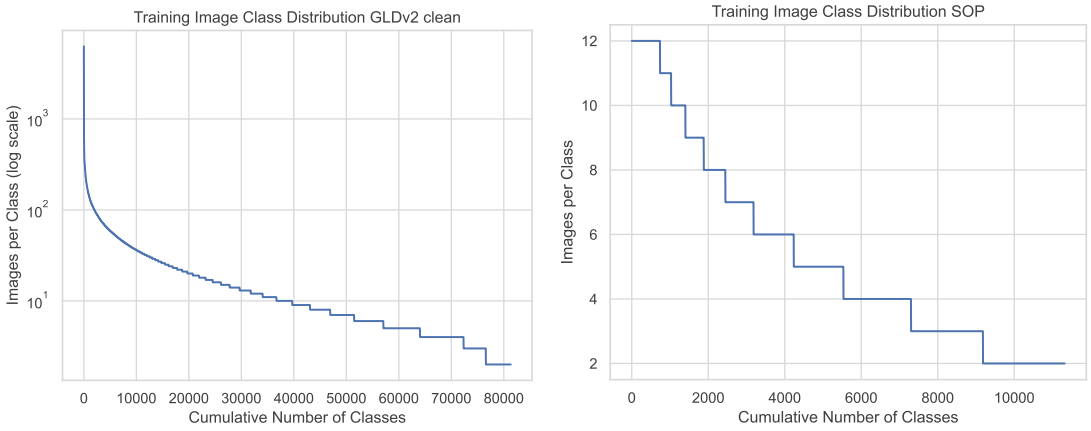
Note that the definition of equation 5.4 in section 5.1.1 is called the *finite sum method*, which is commonly used in retrieval and recommender systems literature. However, the $\mathcal{R}$Oxf and $\mathcal{R}$Par benchmarks use a different definition based on the area under the precision-recall curve. Here, two adjacent precision points on the curve are averaged and then multiplied by the recall step. The authors provide a Github repository[2] with code for their evaluation protocol.

Stanford Online Products (SOP)

Stanford Online Products [60] consists of product images crawled from *ebay.com* and is commonly used as a benchmark for metric learning. Evaluation is performed using R@K as a metric. Note that the metric learning community uses a definition of R@K, which differs from R@K in information retrieval and other contexts.

The dataset contains 120,053 images with 22,634 classes split into a training and testing subset. On average, a class has 5.3 images. The training subset contains 59,51 images with 11,318 classes, and the testing subset contains 60,502 images in 11,316 classes. Figure 5.1b shows the distribution of the number of training images per class. The classes between the two subsets are distinct, making the dataset an open-set classification problem. We perform an additional 95/5 split of the training subset for reporting metrics during training. For image retrieval, the testing split is used as both query and gallery set, i.e., the similarity from each image in the testing split is compared to every other image in the split. However, since geometric verification is a slow iterative process, instead of using every image in the dataset as a query image, a random subset of 1000 images was used instead.

[2] https://github.com/filipradenovic/revisitop



(a) Google Landmarks Dataset v2 'clean' split (b) Stanford Online Products

Figure 5.1: Distribution of the number of training images per class in the datasets. Note the log scale on the Y axis for GLDv2 as the distribution is extremely long-tailed.

5.2 Experiments

5.2.1 General Model Parameters

Following DELG, we train our feature extractor with the SGD optimizer with a momentum of 0.9. We use an ArcFace margin of $m = 0.1$ and equally weigh the global and local classification loss. The learnable cosine classifier scalar is initialized to the square root of global feature dimensionality (see section 2.3.4).

For all experiments on SOP we fixate the learning rate to 0.02 and for GLDv2 to 0.05 For experiments on *ResNet-18*, we use a batch size of 256, and for *ResNet-50*, we use 240 due to memory constraints. Training on SOP is performed for 50 epochs and on GLDv2 for ten epochs.

Due to computational constraints, we use an image size of 224×224 instead of 512×512. Due to the backbone's layout, the last layer of the CNN produces a feature map of 7×7. The penultimate layer produces a feature map of 14×14, so a maximum of 196 local features are used. For all experiments, we use local descriptors with 128 dimensions as this is the dimensionality used in RRT[62] for their experiments. We keep the dimensionality of global descriptors dependent on the backbone (i.e., 512 for *ResNet-18* and 2048 for *ResNet-50*).

MI Estimation

For all experiments, we use Adam [32] with a learning rate of 10^{-6} for the MINE network optimizer. For every epoch that we train the DELG based feature extractor, we train the auxiliary MINE network for 20 epochs. Input dimensions for MINE are fixed to 64 for all experiments and measurements.

Re-ranking using RRT

We keep all parameters the same as the official code repository suggests. As an exception, we only train for five instead of 15 epochs on the exported features from the GLDv2 dataset when evaluating on $\mathcal{R}$Oxf/$\mathcal{R}$Par as there is only a slight improvement after this point.

5.2.2 Minimization of Mutual Information

We trained feature extractor models with varying degrees of the β penalty factor. Higher β values increasingly penalize $I(Z_g; Z_{la})$ while $\beta = 0$ corresponds to DELG[12] as a feature extractor model (keeping in mind the implementation details and differences described in section 4.1.3). To perform the estimation of mutual information, we train the feature extractor models twice for two methods: MINE [8] and Rényi's α-order matrix-based functional [53]. The results of this experiment are described in section 6.2 where we report on the estimated MI values between global (Z_g) and local features (Z_la).

We add several auxiliary measurement points (using additional mine networks) to our model that are not part of the primary loss function. However, adding these auxiliary components increases the total number of parameters in the model. Using these measurements, we can investigate if MINE is actually able to minimize MI or only deteriorate its estimation (results section 6.2.1). This measurement duplicates the 'main' MINE network that estimates $I(Z_g; Z_{la})$. By comparing the mutual information before and after attention pooling, we can gain insights into the attention mechanism of the feature extraction model and how the minimization objective affects local features (results section 6.2.3). This measurement adds another CNN so the local features before attention pooling can be processed. Adding these components slows training, so we did not use them for our experiments on the large-scale datasets SOP and GLDv2. Instead, we performed them on the much smaller Cifar100.

Hyperparameters

We performed training with minimization of mutual information using Rényi on the SOP and GLDv2 datasets. On GLDv2, 5 different β values $(0.0, 5 \times 10^{-3}, 1 \times 10^{-2}, 2 \times 10^{-2}, 5 \times 10^{-2})$ and on SOP 7, different β values $(0.0, 5 \times 10^{-3}, 1 \times 10^{-2}, 2 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-1}, 2 \times 10^{-1})$ were used.

We performed training with minimization of mutual information using MINE on the SOP dataset. As training using MINE takes much longer, we did not evaluate on the bigger GLDv2 dataset. We trained the feature extractor using 7 different β values (0.0, $5 \times 10^{-3}, 1 \times 10^{-2}, 2 \times 10^{-}2, 5 \times 10^{-2}, 1 \times 10^{-1}, 2 \times 10^{-1}$) for ResNet-18 and 5 different *beta* values (0.0, $1 \times 10^{-2}, 2 \times 10^{-}2, 5 \times 10^{-2}, 1 \times 10^{-1}$) for ResNet-50.

5.2.3 Performance Improvements of Re-Ranking

We take the trained feature extractors and perform re-ranking to determine whether the minimization of mutual information between global and local features positively affects retrieval performance.

We consider $\beta = 0$ to be our baseline and compare two different methods for re-ranking: *geometric verification* as described in DELG and RRT [62]. The final retrieval rank for *geometric verification* does not consider the global feature while RRT encodes both global and local features in its input sequence. We chose to compare these two methods to evaluate whether omitting global features for the final score calculation leads to reduced performance when minimizing MI. The results of this experiment are described in section 6.3.

For models trained on GLDv2, retrieval evaluation was performed on the $\mathcal{R}$Par and $\mathcal{R}$Oxf benchmark datasets (as they are the most common in literature). For $\mathcal{R}$Par, the seed for RRT was always kept at 0 (as the feature extractor was already trained using varying seeds). For $\mathcal{R}$Oxf, two seeds for each seed of the feature extractor were evaluated as the initial results were particularly noisy. SOP provides its dataset split for evaluation.

Process

Figure 5.2 shows the overall process for our re-ranking experiment, which has to be repeated for each hyperparameter choice. First, the feature extractor is trained using the train and validation split with random data augmentations applied to the train split as described above. The trained model is then used to extract all features of all images in the query, gallery, and the second train set. For GLDv2, a smaller second train set is used to train the RRT. The same dataset split as for the feature extractor is used for all other datasets, but no random data augmentation is applied. Using the global descriptors, a nearest neighbor index is calculated. Together with the dataset ground truth (label information), the index can be used to calculate metrics for global-only retrieval. The top K, closest neighbors, based on global feature similarity, is used in the geometric verification process to match images based on local features and their location in the image.

The nearest neighbors and the ground truth and features are saved to disk for usage with RRT. Exported features can take up a lot of disk space, e.g., 19GB for the SOP dataset. For comparison, the original images take up only around 3GB. RRT has to be trained separately from the feature extractor on the second train set using the ground truth data and the nearest neighbors for sampling. After training, retrieval metrics are calculated based on the exported query and gallery features.

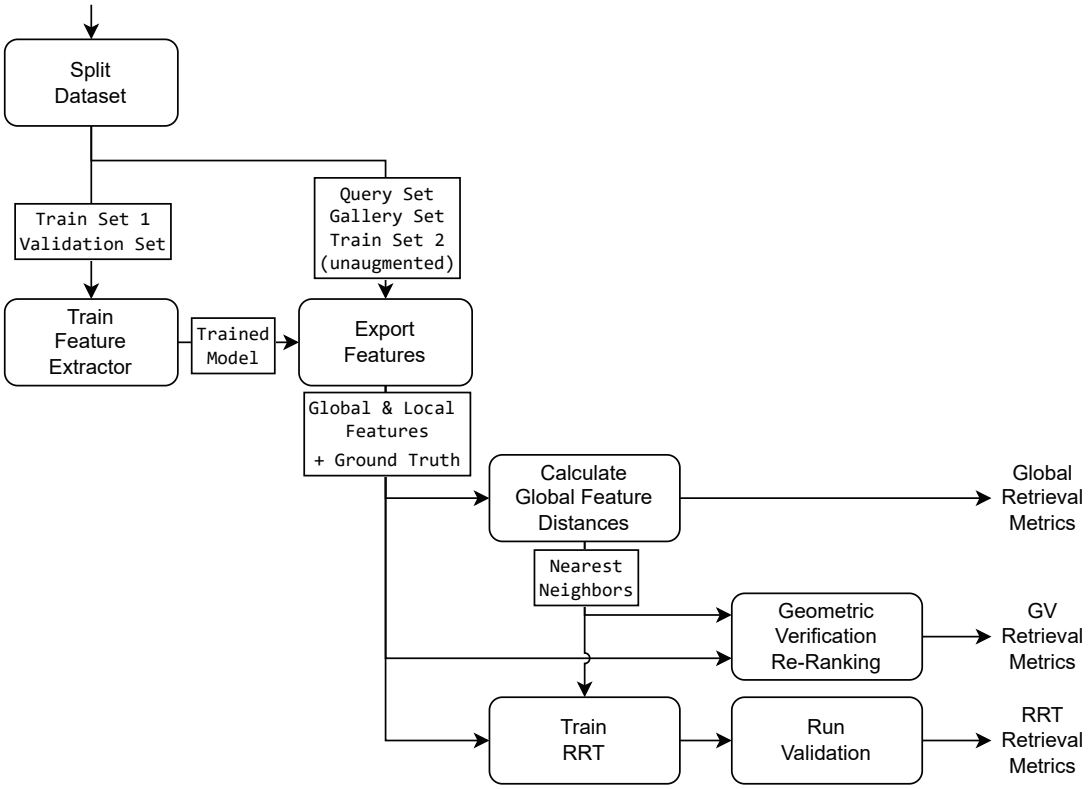


Figure 5.2: Process of Re-Ranking Experiment

CHAPTER 6

Results

6.1 Benchmark Comparison

The absolute mAP values presented for our work are not competitive with the state of the art. Our goal is to improve the benefits of re-ranking. Therefore, we report the relative improvement of re-ranking compared to global-only retrieval. The global-only performance is not an essential benchmark for our work. We removed some components of DELG[12] to simplify our implementation (see section 4.1.3). These components positively affect retrieval performance, which explains this discrepancy. Additionally, a simpler CNN backbone and smaller image size were used due to computational restraints. Table 6.1 shows the absolute retrieval performance of our model when compared to the results reported in [12]. The *ResNet-18* variant with image size 224×224 was used in nearly all experiments. For comparison, we performed a run with the same image size as DELG (512×512) and using a *ResNet-50* backbone but without multiple image sizes for retrieval (image pyramid).

		Image			RPar		ROxf	
Impl.	Backbone	Size	Pyramid	Epochs	Global	GV	Global	GV
Ours	ResNet-18	224	×	10	75.9	75.9	40.9	43.4
Ours	ResNet-50	512	×	10	78.8	79.0	52.5	54.3
DELG[12]	ResNet-50	512	✓	∼25	85.7	85.7	73.6	78.3
DELG[12]	ResNet101	512	✓	∼25	86.6	87.2	76.3	81.2

Table 6.1: Benchmark Comparison between the official DELG results and our implementation. Results (mAP) are shown on the Medium evaluation protocols of $\mathcal{R}$Oxf and $\mathcal{R}$Par for global-only retrieval and re-ranking using geometric verification. DELG results are taken as reported in table 7 of their work [12] (using 'v2-clean' dataset split). Our results are from a single training run without any tuning of hyperparameters.

Table 6.2 compares the retrieval performance for the SOP dataset between our experiments and the results presented in [62] for RRT. [62] train their feature extractor using a contrastive loss and re-rank based on local features from the last convolutional layer. The global feature is a spatially averaged version of these descriptors and is used purely for training. We stick to the approach of DELG for our experiments, which uses the penultimate layer for local descriptors and train the backbone using ArcFace[18] loss.

		Backbone Training		R@1		R@10	
Impl.	Backbone	Loss	Epochs	Global	RRT	Global	RRT
Ours	ResNet-18	ArcFace	10	72.3	73.5	85.6	88.0
Ours	ResNet-50	ArcFace	10	79.0	77.7	90.1	90.4
RRT[62]	ResNet-50	Contrastive	100	80.7	81.8	91.9	92.4

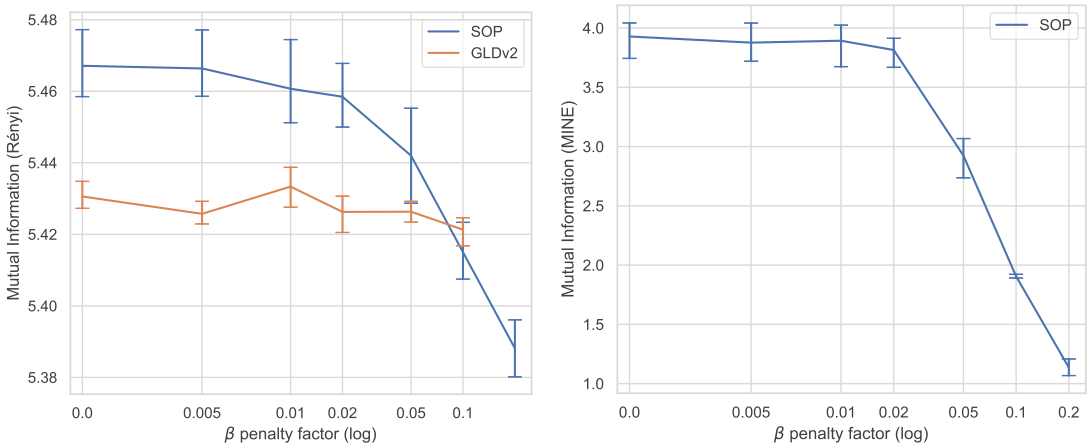
Table 6.2: Benchmark Comparison between the results presented in [62] for RRT and our experiments for the SOP dataset. The R@K definition of metric learning is used. Results for RRT are directly taken as reported in table 6 of their work. We compare against their experiments using the frozen backbone as it more closely aligns with our experimental setup.

6.2 Minimization of Mutual Information

The architecture of DELG[12] uses two classifiers to train the model: One for the CNN backbone that produces the global descriptor (Z_g) and one to train the local descriptors. The local descriptors are trained by pooling them using a weighted average based on their attention scores. This pooled version (Z_{la}) of the local descriptors is passed to the second classifier, which is for training the attention module and the dimensionality reducing CNN layer (see section 4.1.1 and 4.1.3). We minimize mutual information between global and local descriptors using a measurement of $I(Z_g; Z_{la})$.

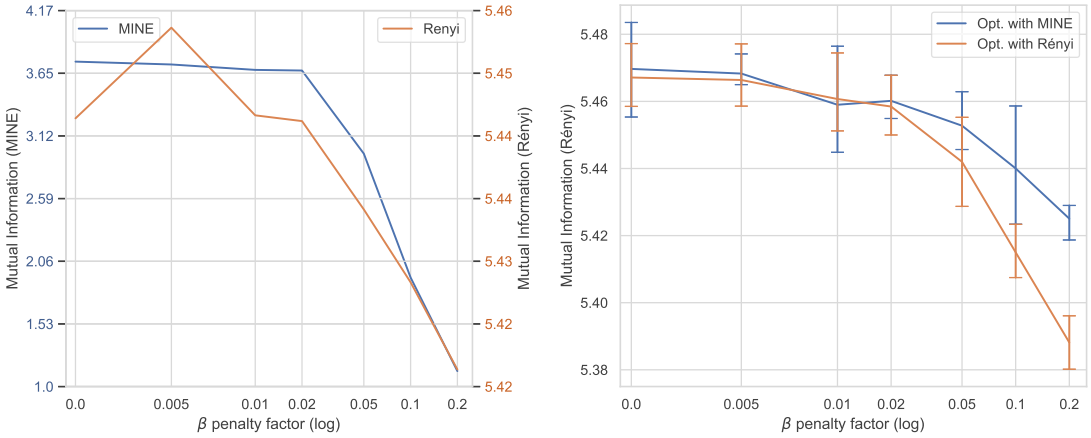
Figure 6.1 shows the effect that the β penalty factor has on the measured MI values between the global descriptor Z_g and the attention pooled local descriptors Z_{la} for both methods of measuring mutual information. The numerical values between the different methods do not agree, but their relative trend is similar. This is expected as Rényi's α-order matrix functional measures information theoretic quantities in the reproducing kernel Hilbert space. While the measurement exhibits similar properties as Shannon entropy, their numerical values differ [74].

Figure 6.2a further illustrates the numerical discrepancy between MINE and Rényi by measuring both on the same training runs. The measured values shown exhibit a linear correlation of $r = 0.956$ (Pearson). Figure 6.2b shows the influence of the minimization method on mutual information. Both methods behave similarly for very small β values, but for increasing β, Rényi can better minimize MI. This discrepancy is further discussed in section 6.2.1.



(a) Rényi (b) MINE

Figure 6.1: Effect of the β penalty factor on MI between global and local descriptors. Subfigure a) shows the effect when minimizing and measuring mutual information using Rényis α-order matrix for the datasets SOP and GLDv2. Subfigure b) shows the effect when minimizing and measuring using MINE. As this approach is computationally more expensive, evaluation has only been performed on the smaller SOP dataset.



(a) Numerical Agreement of MINE & Rényi (b) Influence of Method Choice

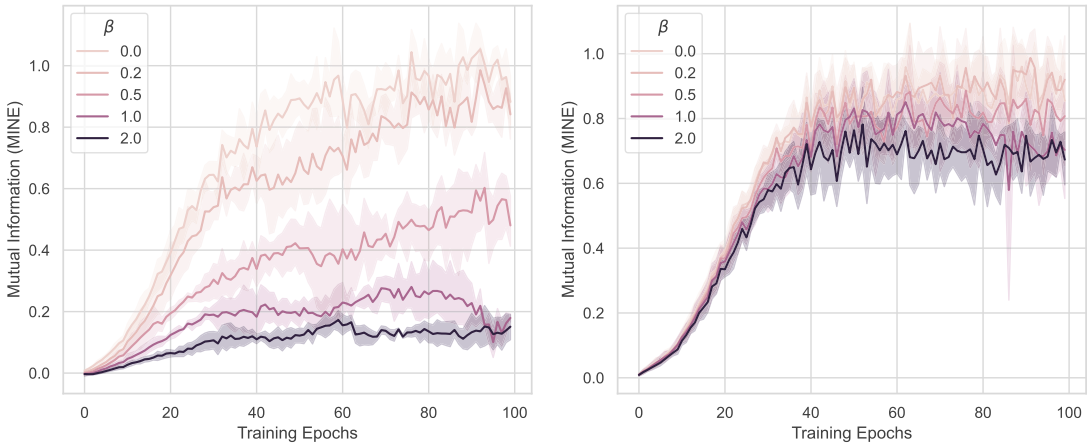
Figure 6.2: Further comparison of methods showing the effect of β on MI between global and local descriptors on the SOP dataset. Subfigure a) shows the numerical discrepancy of the measured MI values. Both measurements were done on a single training run per β value. Minimization was performed using the MINE measurement. Subfigure b) shows the influence of the method choice on mutual information minimization. MI was measured using Rényi, but both methods were used for minimization.

6.2.1 Limitations of minimizing a lower bound

As described in section 2.4.4, MINE uses a lower bound on the KL-divergence to estimate mutual information. By simply producing a worse lower bound with increasing β, MINE could produce a lower MI value without affecting the actual information content of the underlying random variables. We introduce a second MINE network to investigate whether it can minimize mutual information and not just deteriorate its estimation. This network is trained purely with the MI estimation maximization objective as with other auxiliary MINE-based measurements (see section 4.3.1). This means the β penalty has no direct effect on the second network but only on the underlying representations.

Figure 6.3 shows that using MINE for minimization is possible. While the main network's estimation for the minimization exhibits a greater reduction of the estimated MI values, the second auxiliary network clearly shows reduced MI values with increasing β. This suggests that MINE can successfully be used to minimize mutual information.

Figure 6.4 shows the same effect but for an increased range of β values. With increasing β penalty, the main MINE network collapses as the minimization objective overpowers the training objective of MINE itself. A small gradient can still pass through to the feature extractor, but using high β values yields diminishing returns on minimization.



(a) Main Network

(b) Auxiliary Network

Figure 6.3: Comparison between the MINE network used in the minimization objective vs an auxiliary MINE network that is unaffected by the minimization. Graphs show the effect of the β penalty factor on MI values of the validation set during training. Runs performed on Cifar100 with a simplified version of ResNet-18 as the backbone.

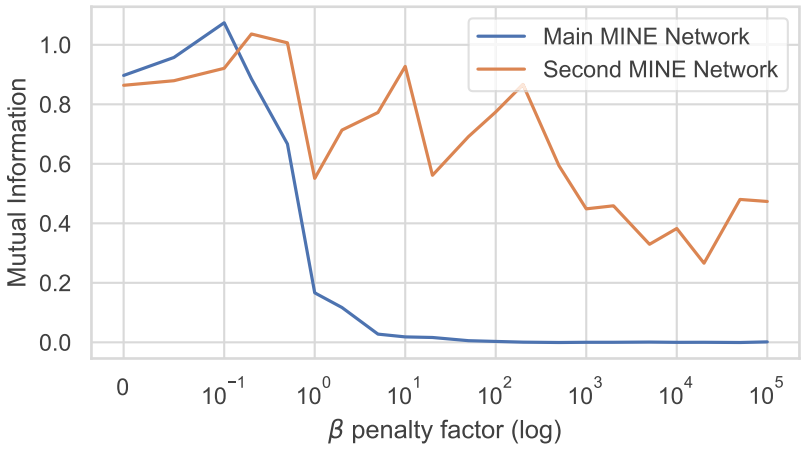
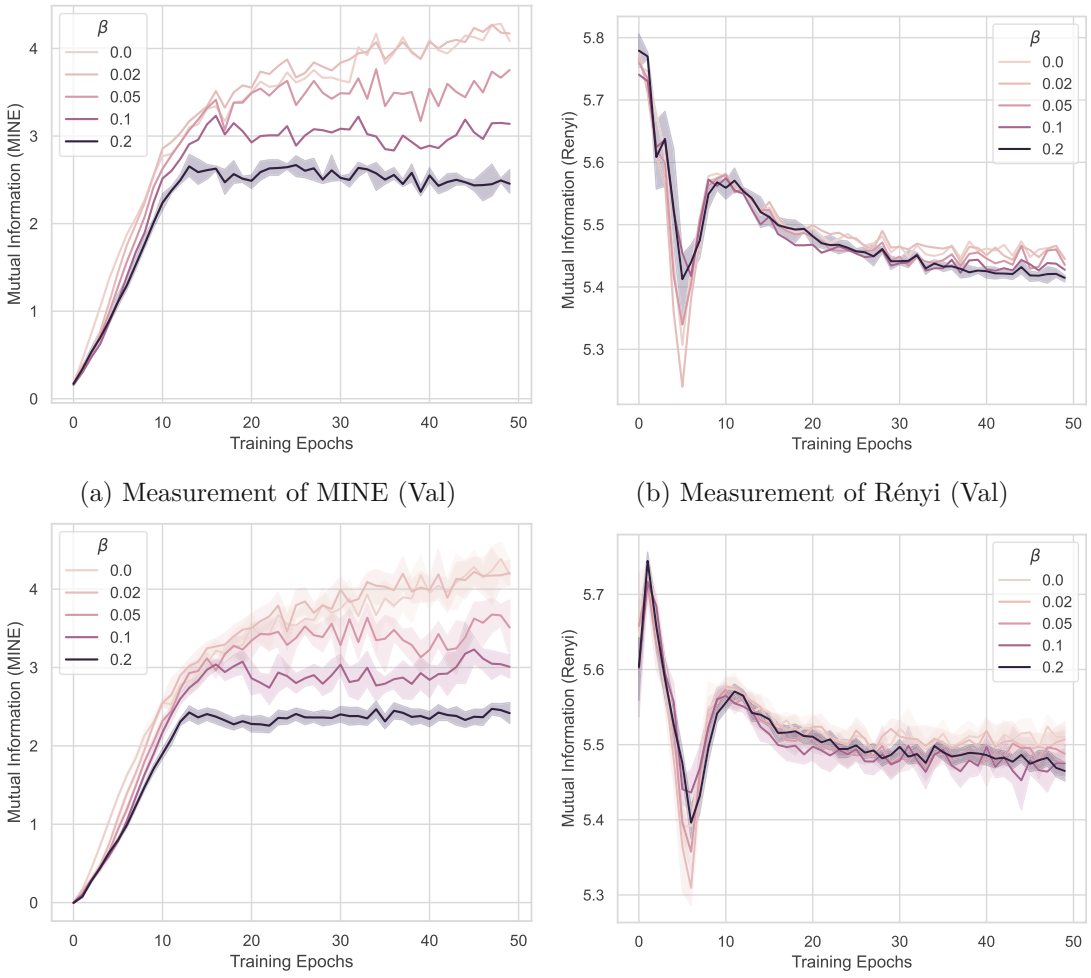


Figure 6.4: Comparison of the effect of the β penalty factor on the estimated MI values between the MINE network that is used in the minimization objective vs an auxiliary MINE network that is not affected by the minimization. Same experiment setup as figure 6.3, but with an increased range of β values.

6.2.2 Training Trajectory

Figure 6.5 compares the measurements of MI between MINE and Rényi on the same runs. While both methods produce final measurements that show that MI is dependent on β, their trajectories behave very differently. The measurements of MINE start at (close to) 0 and approach the real value during training. As the objective of MINE is to maximize a lower bound on MI, this is expected. For Rényi, the measurements start high and then rapidly decrease in the early stages of training.

Figure 6.6 shows the individual components of the Rényi measurement (recall $I(X;Y) = H(X) + H(Y) - H(X;Y)$) during training. Again, $I(Z_{la};Z_g)$ is sucessfully reduced with increasing β. Note that the shown values for $I(Z_{la};Z_g)$ do not match those in figure 6.5d as one was minimized using the MINE measurement and the other using the Rényi measurement. In this case, Rényi produces a smoother training trajectory. Since backpropagation to the global representation is stopped, $H(Z_g)$ stays the same for all runs. This leaves $H(Z_{la})$ and $H(Z_{la};Z_g)$ as the only factors where β can have an influence: $H(Z_{la})$ decreases, and $H(Z_{la};Z_G)$ increases with higher β.



(a) Measurement of MINE (Val)

(b) Measurement of Rényi (Val)

(c) Measurement of MINE (Train)

(d) Measurement of Rényi (Train)

Figure 6.5: Effect of the β penalty factor on MI between global and local descriptors during training on SOP. Minimization was performed using MINE, and its measurement can be seen in a). Additionally, the measurement by Rényi is shown in b). The X-axis shows the training process in steps. A running average with a window size of 5 (logging steps) was applied for visual clarity. The Y-axis shows the MI values of the validation set. Values are averaged across different training seeds.

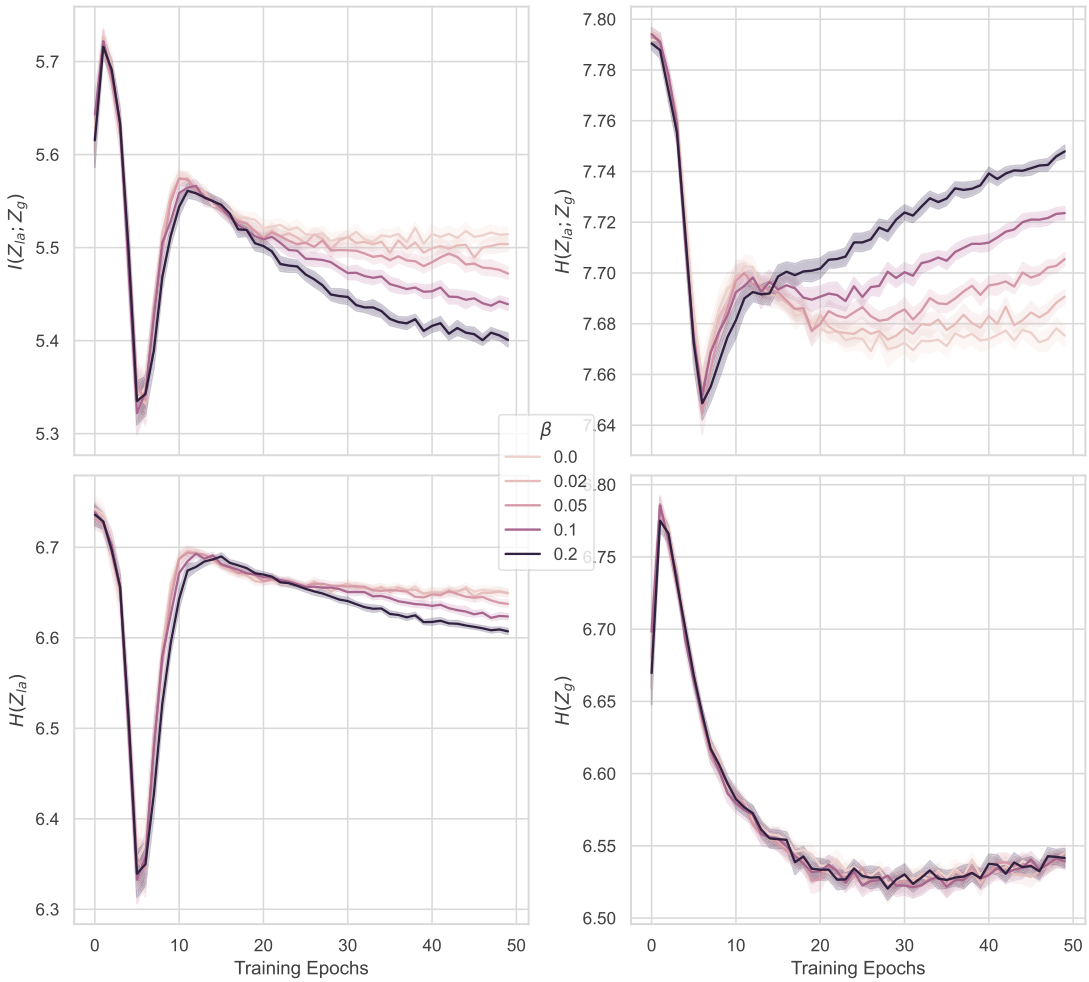
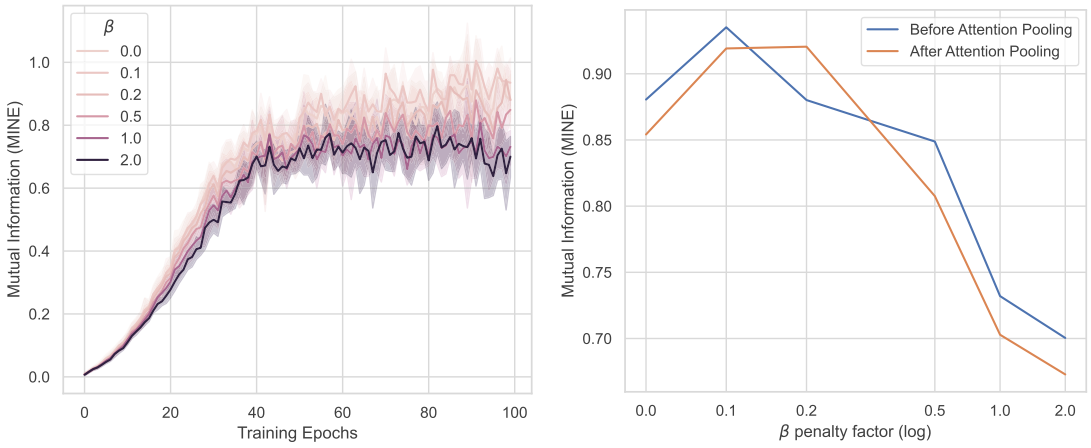


Figure 6.6: Individual components of the Rényi estimation during training on SOP. Minimization was performed using the measurement by Rényi.

6.2.3 Relation to Local Feature Attention

Since the backpropagation of gradients from the local feature path is stopped before the CNN backbone and the global descriptor, there are only two places in the model where mutual information reduction can influence local features: The optional dimensionality reduction convolution layer and in the attention module (recall figure 4.5).

Figure 6.7 shows measurements of mutual information before the attention module and compares them with those after attention pooling. While measurements are noisy, they essentially follow the data processing inequality. Both measurements decrease at about the same rate with increasing β, indicating that neither the dimensionality reduction nor the attention module is solely responsible for the reduction.



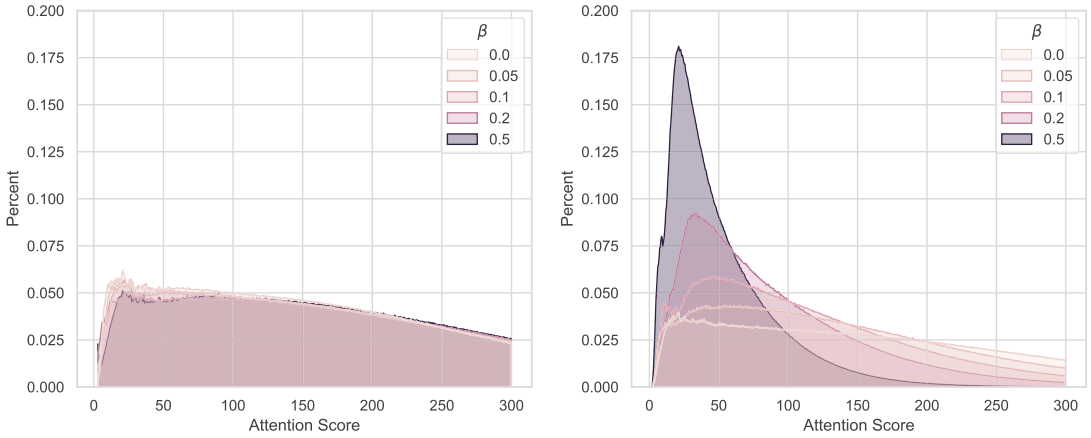
(a) Training History Before Pooling

(b) Comparison MI Before/After Pooling

Figure 6.7: Measurements of MI before Attention Pooling on CIFAR100. Subfigure a) shows the training history of the measurement and can be compared to figure 6.3b. Subfigure b) compares the measurement of the last epoch (depending on β) with the measurements after attention pooling (aforementioned figure).

Figure 6.8 compares two histograms of local feature attention scores by the method of mutual information minimization. Table 6.3 shows summary statistics for the same data. Interestingly, MINE and Rényi have diverging effects: MINE increases the median attention score and the standard deviation. Rényi strongly decreases the median score and also the standard deviation. The difference of the two methods is especially apparent in the histograms.

Figure 6.9 compares the median attention score of local features (as depicted in table 6.3) with the parameter σ used in the calculation of Rényi entropy. Recall that matrix-based Rényi's α-order functional needs two hyperparameters: α and the kernel width σ. We automatically calculate σ as the average of the mean L2 distances of the $k = 10$ nearest neighbors in the local feature space. As such, it is directly proportional to the magnitude of the local features after attention pooling. Since attention pooling is just a weighted



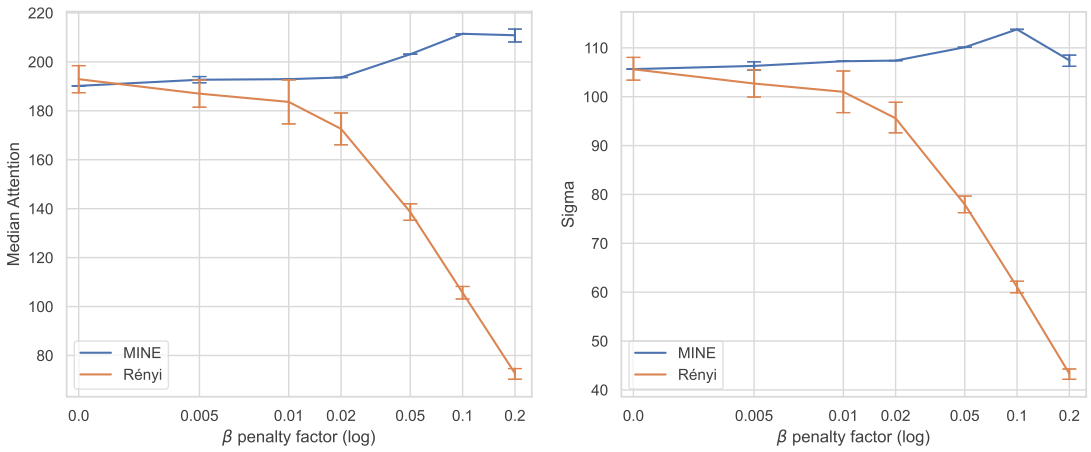
(a) MINE (b) Rényi

Figure 6.8: Histograms of local feature attention scores depending on the β penalty factor. Data from the experiments on SOP. a) shows the results when minimizing using the measurement of MINE while b) shows the same for Rényi.

	MINE			Renyi		
β	Mean	Median	Std.	Mean	Median	Std.
0.00	187.7	159.4	141.6	187.3	158.8	141.3
0.05	196.1	165.7	148.4	150.9	126.6	112.4
0.10	202.2	170.3	153.1	121.7	100.1	91.1
0.20	206.3	175.1	154.8	91.8	73.1	69.1
0.50	207.2	177.2	150.9	51.5	34.0	39.2

Table 6.3: Summary statistics for local feature attention scores on SOP

average based on the respective attention scores, it is directly captured by the median attention score. Therefore, the minimization of mutual information directly impacts the hyperparameter σ used to estimate mutual information. This feedback loop could indicate that our heuristic method for σ is unsuitable and mutual information is not reduced. However, [79] point out that mutual information increases monotonically as σ decreases, which we do not observe.



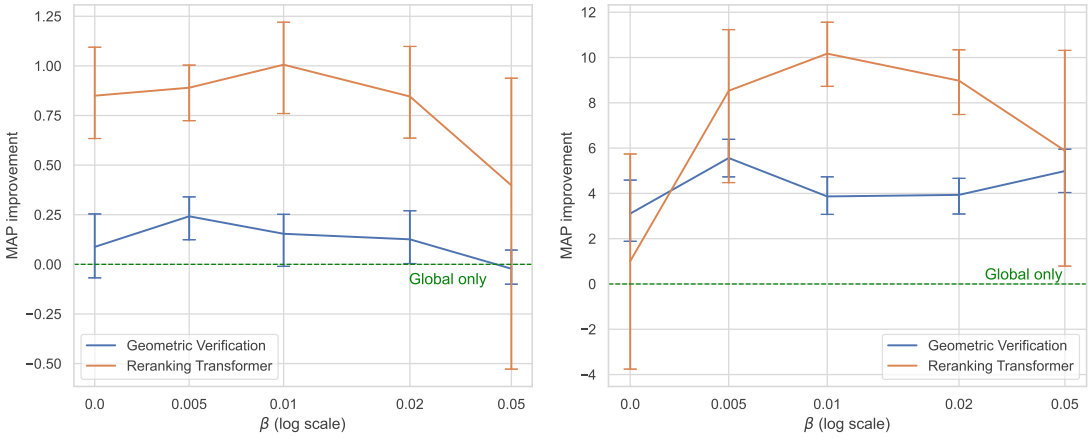
(a) Median Attention Score

(b) σ

Figure 6.9: Comparison between the median attention score and σ used for calculating the Rényi entropy

6.3 Performance Improvements of Re-Ranking

The details of the experimental setup are described in section 5.2.3. Experiments were repeated multiple times for different random seeds, and the re-ranking performance was evaluated using the geometric verification process described in DELG as well as with RRT[62]. The improvement of re-ranking is calculated by subtracting the mAP achieved by re-ranking with the mAP from retrieval using only global features from the same experimental run. *Rényi* was evaluated on SOP and $\mathcal{R}$Oxf/$\mathcal{R}$Par (for which training was performed on GLDv2). MINE was evaluated only on SOP. Appendix A contains the retrieval performance of all individual runs. Note that β values across datasets do not correspond exactly.



(a) Revisited Paris (Rényi)

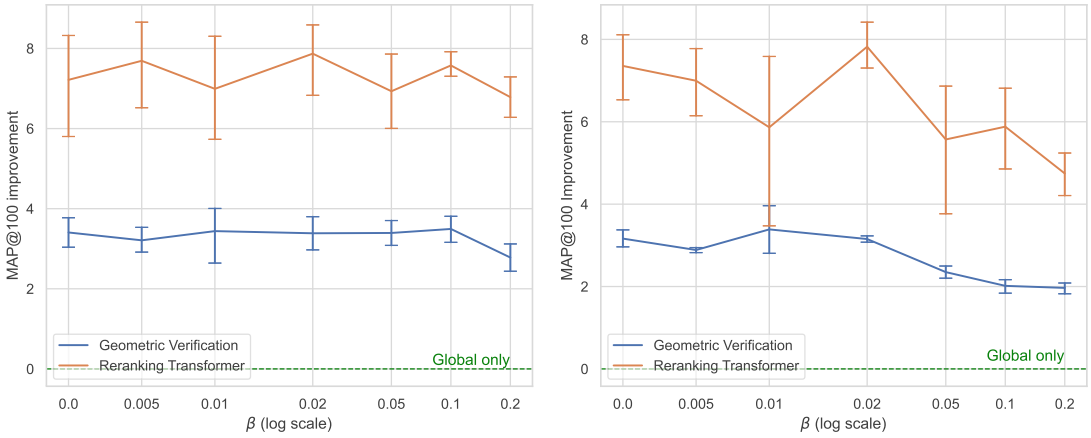
(b) Revisited Oxford (Rényi)

Figure 6.10: Performance improvements of re-ranking with MI estimation using Rényi. The Y-axis shows the mean improvement across different runs with a (non-parametric) 95% confidence interval. The X-axis shows the improvement depending on the mutual information penalty factor β between global and local features.

Figure 6.10 shows the improvement of re-ranking on $\mathcal{R}$Oxf and $\mathcal{R}$Par. For both evaluation datasets $\beta = 0.01$ performs the best when re-ranking using RRT and $\beta = 0.005$ for geometric verification. $\mathcal{R}$Oxf shows a far bigger improvement of re-ranking than $\mathcal{R}$Par which is consistent with results reported in [12].

Figure 6.11 compares the two MI estimation methods of estimating MI on SOP. Both methods perform the best on $\beta = 0.02$ when re-ranking using RRT. For geometric verification, β has no noticeable effect for smaller values.

RRT repeatedly performs better than geometric verification in all experiments independently of the mi-penalty factor β, validating the result in [62]. An exception is on SOP using MINE, which is the only experiment using *ResNet-50* as the cnn backbone (figure 6.12), where geometric verification performs consistently better. For higher β values, re-ranking actually decreases the performance (compared to only global retrieval). We



(a) SOP (Rényi) (b) SOP (MINE)

Figure 6.11: Comparison of re-ranking performance improvements on SOP between *Rényi* and MINE for MI estimation.

suspect this is due to the dimensionality of the feature types. All our experiments use local descriptors of 128 dimensions but the global descriptor has 512 dimensions for ResNet-18 and 2048 for ResNet-50. As such, the global descriptor for ResNet-50 can contain a lot more information, which explains the comparatively good performance of global-only retrieval. However, RRT exhibits a higher variance in its performance metrics, with some runs exhibiting unsatisfactory performance.

Across all experiments, RRT performs best for a moderate (out of all tested) MI penalty-factor β. However, the retrieval performance exhibits a high statistical uncertainty even after repeated training runs. Only the evaluation on $\mathcal{R}$Oxf (figure 6.10b) exhibits a significant increase in re-ranking performance for $beta = 0.01$.

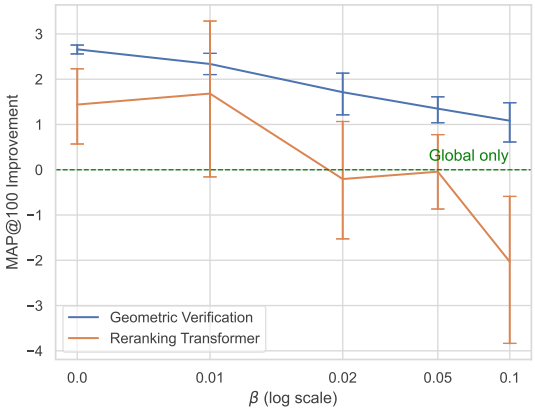


Figure 6.12: Re-ranking performance improvements on SOP using MINE for MI estimation and ResNet-50 backbone, as opposed to ResNet-18 in figure 6.11b.

6.3.1 Influence of RRT Sequence Length

Given the input image size of 224, the ResNet backbones used for our experiments always produce $14 \times 14 = 196$ local features per image. We follow the approach in [12] and select only features with an attention score greater than the median score of the last training epoch. Since the median score is calculated over the whole training epoch, different images will use a different number of local features (even when the overall number is close to half of all possible features). Figure 6.13 shows that distribution.

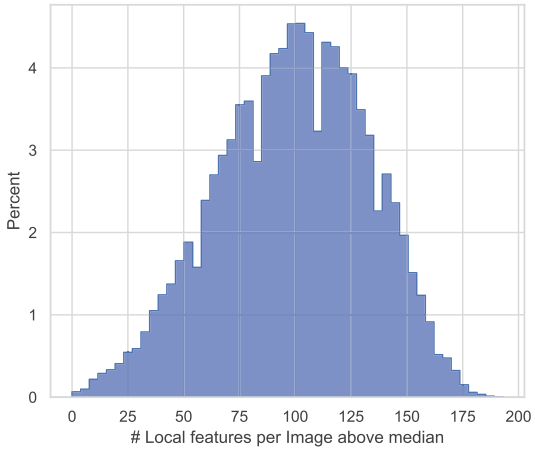
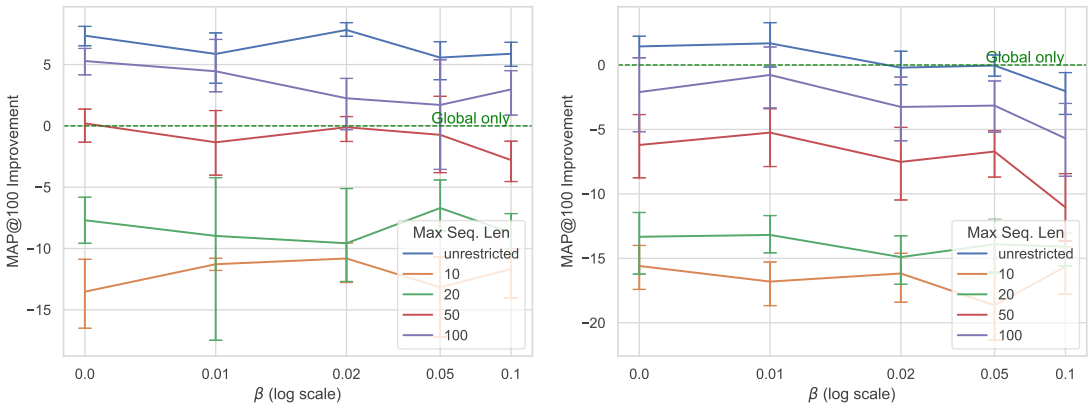


Figure 6.13: Distribution of the number of local features selected for re-ranking. For the SOP dataset without any MI estimation.

Figure 6.14 shows how restricting the maximum sequence length (and thus the number of local features) that RRT can use influences its retrieval performance. Unsurprisingly, shorter sequences deteriorate performance as the model has less information to work with. Note that local features are exported in decreasing order of their attention scores, so the 'most important' features (as determined by the attention module of the feature extractor) are kept the longest in this experiment. Restricting the sequence to 50 tokens already leads to a re-ranking performance worse than global-only retrieval (for ResNet-18, subfigure 6.14a). As the global feature is included in the sequence, this suggests that RRT uses the information of the global feature very inefficiently since it encodes both feature types with only 128 dimensions (down from 512 for the global feature from ResNet-18).



(a) ResNet-18 (b) ResNet-50

Figure 6.14: Re-ranking performance improvements when re-ranking using RRT depending on the maximum usable sequence length. For the SOP dataset with minimization using MINE.

CHAPTER 7

Conclusion

A common image retrieval setup uses two types of representations for re-ranking: global and local features. This thesis proposes a novel system that explicitly models the information-theoretic relationship between these representations. This approach is related to the information bottleneck and multi-view representation learning. We integrate a mutual-information-based loss term into our model objective to reduce redundant information between global and local features.

We demonstrated how a retrieval system can measure and minimize mutual information between feature types. Two different methods of estimation MI and for re-ranking have been evaluated. We performed experiments on standard image retrieval benchmarks and showed that penalizing mutual information improves re-ranking performance on some datasets.

7.1 Research Questions

RQ1: How can we measure and minimize mutual information between global and local features for image retrieval?

Section 6.2 detailed the minimization of mutual information. We used MINE [8] and Rényi's α-order matrix-based functional [53] to estimate and minimize MI between global and local features.

Can we observe mutual information between global and local features?

Yes, to validate the premise of this thesis, we already showed measurements of mutual information with figure 1.1 in the introduction.

Can (neural) mutual information estimation adequately minimize mutual information?

Both methods are capable of minimizing mutual information. Since MINE optimizes a lower bound, its training objective is opposed to our mi minimization objective. While we showed that it is possible to use it for our application, increasing β values makes the reduction less and less efficient. We discussed these caveats in section 6.2.1. This can also be observed in our direct comparison on the SOP dataset (figure 6.2b) where Rényi is able to minimize MI more efficiently for higher β values.

A further downside of MINE is its far slower training time. Our implementation requires multiple training epochs for the auxiliary network for each epoch of the actual feature extractor. Training MINE can be unstable, and its estimates exhibit a high variance. Additionally, since we use it for minimization and not for maximization, we need multiple epochs for the signal to recover sufficiently. We discussed this in our implementation section 4.3.1.

In section 6.3, we showed improvements in re-ranking with local descriptors that have minimized mutual information with the global descriptor. A direct comparison between the two methods was made on the SOP dataset (figure 6.11. The results are inconclusive, and both methods perform at a similar level. However, our experiment on $\mathcal{R}$Oxf and $\mathcal{R}$Par when MI is estimated using Rényi showed promising results (figure 6.10). Training for these benchmarks was performed on GLDv2, the largest dataset we used. Due to computational restraints, we did not evaluate MINE on this dataset.

In summary, Rényi's α-order matrix-based functional is better able to minimize mutual information, performs on par with MINE in terms of re-ranking performance, and is able to do so with $\sim 20\times$ faster training for our experiments.

RQ2: How does minimizing mutual information between global and local features affect the local features?

In our implementation, we use the attention-pooled version of the local features to minimize mutual information. Section 6.2.3 showed how the attention module behaves with increasing MI penalty. MINE had a moderate, and Rényi strongly affected the attention scores associated with local features.

RQ3: How does minimizing mutual information between global and local features affect re-ranking performance?

We compared two approaches to re-ranking: Geometric verification, which calculates the final ordering based solely on local features, and RRT [62], which also encodes the global feature vector into its input sequence. We hypothesized that geometric verification is affected negatively by minimizing MI since it does not use information from the global descriptor. In section 6.3, we presented the results of our re-ranking experiments.

Does minimizing mutual information affect the two approaches differently?

RRT generally performs better than geometric verification (validating the results of [62]) but exhibits more variance in its performance. However, both methods exhibit similar behavior with an increasing β penalty factor: First, the re-ranking improvement increases and then declines after a peak is reached. Figure 6.10 shows this well for the $\mathcal{R}$Oxf and $\mathcal{R}$Par benchmark datasets. For the SOP dataset, the results are inconclusive. This invalidates our hypotheses about the importance of the global descriptor for our tested re-ranking methods. Restricting the number of local features usable for re-ranking with RRT suggests that RRT is not able to efficiently use the information of the global descriptor either (discussed in section 6.3.1). This could additionally explain the similar behavior of both methods and can be considered a limitation of our experimental setup.

Does minimizing mutual information improve re-ranking performance?

Figure 6.10 shows that a moderate penalty on mutual information improves re-ranking performance. Increasing the penalty further deteriorates it.

7.2 Limitations & Future Work

Our experimental setup is easily extendable to other methods of mutual information estimation. E.g., NWJ [42] or NCE [65] can be considered as drop-in replacements for MINE [8] but were not evaluated due to computational restraints.

Using lower bounds to minimize MI works but is suboptimal. It requires multiple training epochs of the auxiliary estimation networks not to overpower the estimation networks' objective. [15] propose a promising upper bound instead, simplifying joint training with our feature extractor.

Rényi's α-order matrix-based functional [53] cannot be applied to CNNs in a meaningful way [80]. As such, we only applied the method to the attention-pooled local features, which is a single high-dimensional vector. While our general methodology is agnostic to model architecture, the univariate Rényi's estimation limits our implementation to architectures similar to DELG [12]. [74] propose an approach based on tensor kernels that can measure mutual information of CNN feature maps. This would allow removing the attention module as an intermediary step and to gain further insights into the relationships of mutual information inside the feature extractor. Instead of optimizing on pooled local features, they could instead be selected by the objective $I(Z_{lc}; Y) - \beta I(Z_{lc}; Z_g)$. [80] propose something similar for the information bottleneck.

We follow the approach in DELG [12] and block the propagation of gradients from the local feature path into the backbone. As a result, reducing mutual information can only affect the attention module and the dimensionality reduction layer. We discussed this in section 6.2.3. A more powerful model for the local feature head would allow for a greater influence of our method on the local feature representations.

While the re-ranking approach of RRT includes the global descriptor, it does not use this information efficiently as it embeds the descriptor into a much smaller feature space. An improved architecture could increase the re-ranking performance.

A limitation of our experimental setup is the exportation step between our feature extractor and re-ranking with RRT. The features are fixed for training the re-ranking model because data augmentation cannot be applied to the training data, which decreases the performance and generalizability of the re-ranking model. A joint architecture would further allow fine-tuning of the feature extractor, which has been shown to improve performance [62].

APPENDIX A

Retrieval Results for Individual Runs

Table A.1: Retrieval results for individual runs on $\mathcal{R}$Oxf and $\mathcal{R}$Par using Rényi.

	Seed		MAP ROxf			MAP RPar		
β	Extractor	RRT	Global	GV	RRT	Global	GV	RRT
0.0	0	0	40.90	43.49	48.63	75.92	75.92	76.62
0.0	0	1	40.90	43.49	35.23			
0.0	1	0	47.00	47.93	52.43	77.79	77.62	78.32
0.0	1	1	47.00	47.93	37.93			
0.0	2	0	40.19	47.53	48.68	75.99	76.32	77.22
0.0	2	1	40.19	47.53	34.00			
0.0	3	0	42.65	45.29	54.96	74.29	74.29	75.37
0.0	3	1	42.65	45.29	38.83			
0.0	4	0	43.94	45.97	52.08	75.10	75.38	75.81
0.0	4	1	43.94	45.97	36.64			
0.005	0	0	39.53	46.66	49.08	75.18	75.48	75.75
0.005	0	1	39.53	46.66	50.57			
0.005	1	0	42.07	46.29	48.91	75.16	75.32	76.04
0.005	1	1	42.07	46.29	51.93			
0.005	2	0	40.51	47.76	32.79	74.97	75.28	75.96
0.005	2	1	40.51	47.76	50.28			
0.005	3	0	40.51	45.44	53.63	74.51	74.56	75.54
0.005	3	1	40.51	45.44	54.06			
0.005	4	0	41.92	46.20	51.07	74.33	74.72	75.31
0.005	4	1	41.92	46.20	52.09			
0.01	0	0	44.86	48.33	50.69	75.51	75.77	76.03

Table A.1: Retrieval results for individual runs on $\mathcal{R}$Oxf and $\mathcal{R}$Par using Rényi.

	Seed		MAP ROxf			MAP RPar		
β	Extractor	RRT	Global	GV	RRT	Global	GV	RRT
0.01	0	1	44.86	48.33	53.97			
0.01	1	0	43.29	45.81	52.93	74.43	74.63	75.64
0.01	1	1	43.29	45.81	52.08			
0.01	2	0	42.52	45.30	54.10	75.69	75.91	77.00
0.01	2	1	42.52	45.30	50.70			
0.01	3	0	42.82	47.22	54.20	74.17	74.43	75.13
0.01	3	1	42.82	47.22	53.33			
0.01	4	0	39.46	45.62	53.44	75.49	75.32	76.52
0.01	4	1	39.46	45.62	52.19			
0.02	0	0	41.58	46.71	50.68	74.64	75.00	75.93
0.02	0	1	41.58	46.71	47.84			
0.02	1	0	42.29	47.57	52.72	73.45	73.66	74.42
0.02	1	1	42.29	47.57	54.79			
0.02	2	0	42.01	45.42	52.64	75.24	75.32	75.79
0.02	2	1	42.01	45.42	46.64			
0.02	3	0	41.20	45.20	52.51	75.56	75.47	76.22
0.02	3	1	41.20	45.20	48.60			
0.02	4	0	43.75	45.60	52.53	74.76	74.83	75.52
0.02	4	1	43.75	45.60	52.49			
0.05	0	0	44.04	48.99	54.80	75.22	75.22	75.94
0.05	0	1	44.04	48.99	35.72			
0.05	1	0	40.57	48.02	50.76	74.54	74.42	75.36
0.05	1	1	40.57	48.02	53.88			
0.05	2	0	43.57	46.23	51.60	76.57	76.73	77.62
0.05	2	1	43.57	46.23	43.66			
0.05	3	0	39.75	44.75	52.38	73.14	73.10	73.97
0.05	3	1	39.75	44.75	49.35			
0.05	4	0	41.83	46.70	33.88	77.94	77.83	76.51
0.05	4	1	41.83	46.70	52.37			

Table A.2: Retrieval results for individual runs on SOP using Rényi

		MAP@100			R@1			R@10		
β	Seed	Global	GV	RRT	Global	GV	RRT	Global	GV	RRT
0.0	0	61.75	64.71	69.32	72.0	74.7	77.38	85.4	87.5	88.44
0.0	1	59.77	63.53	68.67	71.3	74.4	76.66	85.1	87.9	88.10
0.0	2	60.97	63.92	65.56	72.4	74.3	75.14	86.1	88.8	87.05
0.0	3	60.40	64.39	67.75	72.9	74.6	75.93	85.7	88.4	87.93
0.0	4	61.25	64.62	68.92	72.8	74.7	76.65	85.6	88.6	88.58
0.005	0	60.64	64.16	69.66	71.8	74.1	77.25	85.9	88.4	88.77

Table A.2: Retrieval results for individual runs on SOP using Rényi

		MAP@100			R@1			R@10		
β	Seed	Global	GV	RRT	Global	GV	RRT	Global	GV	RRT
0.005	1	60.53	64.25	67.85	72.5	75.1	76.61	85.4	86.9	87.91
0.005	2	61.51	64.55	66.93	73.7	74.8	75.73	86.3	88.8	87.92
0.005	3	61.17	64.16	69.33	72.5	74.1	77.37	86.6	87.6	88.49
0.005	4	60.92	63.70	69.46	72.8	73.8	77.04	85.8	88.3	88.59
0.01	0	62.36	64.21	68.34	74.3	74.1	76.73	86.4	88.6	88.14
0.01	1	60.54	64.35	68.90	71.8	74.5	77.12	84.8	87.5	88.44
0.01	2	60.64	64.87	67.06	72.6	75.7	75.66	85.5	89.2	87.80
0.01	3	61.18	64.61	66.32	72.1	74.7	75.13	86.4	88.3	87.21
0.01	4	60.18	64.06	69.24	70.6	73.5	76.69	85.3	88.3	88.36
0.02	0	60.33	63.39	68.81	71.5	72.2	76.55	85.1	87.6	88.21
0.02	1	60.65	64.51	68.36	72.1	74.3	76.58	85.4	88.3	88.22
0.02	2	60.94	64.34	66.87	72.1	74.4	75.58	86.2	88.5	87.66
0.02	3	60.49	64.43	69.07	71.5	74.2	76.87	85.5	88.6	88.48
0.02	4	60.43	63.10	69.08	71.8	73.2	77.26	85.5	87.6	88.27
0.05	0	61.49	65.01	67.47	72.6	75.1	76.29	85.0	88.9	87.86
0.05	1	60.45	64.38	68.70	71.7	74.6	76.45	85.7	88.1	88.42
0.05	2	60.74	63.66	66.52	73.1	73.4	75.16	85.4	88.0	87.44
0.05	3	61.65	65.22	68.15	72.5	74.6	76.41	86.9	88.3	88.17
0.05	4	60.86	63.89	69.01	72.0	73.3	76.76	84.9	87.8	88.41
0.1	0	61.98	64.81	69.49	73.8	74.6	77.46	85.4	88.2	88.44
0.1	1	60.45	64.36	68.70	71.9	73.7	76.92	85.7	88.2	88.39
0.1	2	60.62	64.05	68.23	71.5	73.7	76.34	85.0	88.4	88.22
0.1	3	60.68	64.08	67.86	72.4	74.4	75.99	85.8	88.3	88.14
0.1	4	60.51	64.41	67.84	72.2	74.4	76.50	85.7	88.6	88.16
0.2	0	60.92	64.07	68.59	71.9	73.5	76.55	85.2	87.7	88.25
0.2	1	61.08	63.96	67.19	73.0	73.9	76.15	85.8	88.9	87.98
0.2	2	60.96	63.43	67.20	72.1	72.3	75.68	86.6	87.5	87.83
0.2	3	61.38	64.59	68.08	73.1	74.7	76.35	86.0	88.6	88.06
0.2	4	61.06	63.25	68.26	72.7	72.9	76.34	84.7	87.2	88.21

Table A.3: Retrieval results for individual runs on SOP using MINE

	Seed		MAP@100			R@1			R@10		
β	Extr.	RRT	Glob.	GV	RRT	Glob.	GV	RRT	Glob.	GV	RRT
0.0	0	0	61.60	64.60	69.95	72.6	74.0	77.72	85.9	88.6	88.76
0.0	0	1	61.60	64.60	68.15	72.6	74.0	77.00	85.9	88.6	88.25
0.0	0	2	61.60	64.60	70.81	72.6	74.0	78.50	85.9	88.6	89.08
0.0	1	0	60.95	64.55	67.08	71.9	75.0	76.12	86.2	88.2	87.84
0.0	1	1	60.95	64.55	68.56	71.9	75.0	76.88	86.2	88.2	88.42
0.0	1	2	60.95	64.55	69.24	71.9	75.0	77.39	86.2	88.2	88.63

Table A.3: Retrieval results for individual runs on SOP using MINE

	Seed		MAP@100			R@1			R@10		
β	Extr.	RRT	Glob.	GV	RRT	Glob.	GV	RRT	Glob.	GV	RRT
0.0	42	0	61.80	64.69	67.22	73.7	74.6	76.22	85.6	89.6	87.70
0.0	42	1	61.80	64.69	69.98	73.7	74.6	77.77	85.6	89.6	88.88
0.0	42	2	61.80	64.69	68.26	73.7	74.6	76.77	85.6	89.6	88.10
0.005	0	0	61.44	64.43	69.58	71.7	74.1	77.57	86.6	89.1	88.70
0.005	0	1	61.44	64.43	66.68	71.7	74.1	75.51	86.6	89.1	87.65
0.005	0	2	61.44	64.43	69.58	71.7	74.1	77.48	86.6	89.1	88.88
0.005	1	0	60.54	63.31	67.49	72.9	73.7	76.05	86.3	88.6	88.00
0.005	1	1	60.54	63.31	69.42	72.9	73.7	77.72	86.3	88.6	88.64
0.005	1	2	60.54	63.31	68.23	72.9	73.7	76.85	86.3	88.6	88.59
0.005	42	0	62.30	65.20	68.39	73.1	74.7	76.99	85.4	87.8	88.15
0.005	42	1	62.30	65.20	67.46	73.1	74.7	76.46	85.4	87.8	87.94
0.005	42	2	62.30	65.20	68.97	73.1	74.7	77.41	85.4	87.8	88.58
0.01	0	0	61.54	64.94	66.55	72.4	73.7	75.59	86.4	88.9	87.67
0.01	0	1	61.54	64.94	68.05	72.4	73.7	76.97	86.4	88.9	87.94
0.01	0	2	61.54	64.94	66.32	72.4	73.7	75.43	86.4	88.9	87.62
0.01	1	0	60.79	65.21	66.91	72.4	76.8	75.99	85.4	88.3	87.74
0.01	1	1	60.79	65.21	67.77	72.4	76.8	76.38	85.4	88.3	88.07
0.01	1	2	60.79	65.21	70.45	72.4	76.8	78.15	85.4	88.3	89.05
0.01	42	0	61.69	64.03	69.18	74.6	74.0	77.53	86.1	89.2	88.57
0.01	42	1	61.69	64.03	70.05	74.6	74.0	77.68	86.1	89.2	88.88
0.01	42	2	61.69	64.03	59.57	74.6	74.0	69.49	86.1	89.2	85.50
0.02	0	0	61.36	64.59	69.32	72.9	73.8	77.52	85.7	88.8	88.54
0.02	0	1	61.36	64.59	68.43	72.9	73.8	77.16	85.7	88.8	88.21
0.02	0	2	61.36	64.59	71.25	72.9	73.8	78.67	85.7	88.8	89.31
0.02	1	0	60.63	63.86	67.70	72.2	74.8	76.47	85.2	88.1	88.05
0.02	1	1	60.63	63.86	69.07	72.2	74.8	77.22	85.2	88.1	88.50
0.02	1	2	60.63	63.86	68.56	72.2	74.8	76.86	85.2	88.1	88.48
0.02	42	0	61.71	64.71	69.11	72.9	74.7	77.21	85.9	88.5	88.32
0.02	42	1	61.71	64.71	68.52	72.9	74.7	76.74	85.9	88.5	88.38
0.02	42	2	61.71	64.71	69.54	72.9	74.7	77.41	85.9	88.5	88.61
0.05	0	0	61.49	64.13	66.68	72.9	73.7	75.70	86.9	88.2	87.48
0.05	0	1	61.49	64.13	61.10	72.9	73.7	71.14	86.9	88.2	85.86
0.05	0	2	61.49	64.13	68.65	72.9	73.7	76.98	86.9	88.2	88.36
0.05	1	0	61.51	63.83	68.99	72.9	74.1	77.29	86.3	89.1	88.55
0.05	1	1	61.51	63.83	67.59	72.9	74.1	76.54	86.3	89.1	88.11
0.05	1	2	61.51	63.83	69.57	72.9	74.1	77.51	86.3	89.1	88.84
0.05	42	0	61.60	63.69	68.08	73.3	74.3	76.80	86.2	88.3	88.18
0.05	42	1	61.60	63.69	65.87	73.3	74.3	75.40	86.2	88.3	87.42
0.05	42	2	61.60	63.69	67.40	73.3	74.3	75.97	86.2	88.3	88.07
0.1	0	0	61.75	63.85	67.70	73.3	74.3	76.68	85.7	87.8	87.93

Table A.3: Retrieval results for individual runs on SOP using MINE

	Seed		MAP@100			R@1			R@10		
β	Extr.	RRT	Glob.	GV	RRT	Glob.	GV	RRT	Glob.	GV	RRT
0.1	0	1	61.75	63.85	67.02	73.3	74.3	75.98	85.7	87.8	87.86
0.1	0	2	61.75	63.85	68.89	73.3	74.3	77.37	85.7	87.8	88.61
0.1	1	0	61.21	63.48	65.58	72.6	73.7	74.99	86.2	88.3	87.40
0.1	1	1	61.21	63.48	67.94	72.6	73.7	76.38	86.2	88.3	88.28
0.1	1	2	61.21	63.48	64.33	72.6	73.7	73.86	86.2	88.3	87.02
0.1	42	0	61.88	63.56	70.04	73.3	73.4	77.71	86.5	87.8	89.10
0.1	42	1	61.88	63.56	68.93	73.3	73.4	77.02	86.5	87.8	88.67
0.1	42	2	61.88	63.56	67.02	73.3	73.4	76.09	86.5	87.8	87.77
0.2	0	0	60.81	62.95	65.41	72.4	73.2	75.28	86.8	88.9	87.35
0.2	0	1	60.81	62.95	65.59	72.4	73.2	74.98	86.8	88.9	87.68
0.2	0	2	60.81	62.95	66.66	72.4	73.2	75.56	86.8	88.9	88.12
0.2	1	0	61.13	63.20	66.70	72.5	74.1	76.03	86.0	88.0	88.01
0.2	1	1	61.13	63.20	65.87	72.5	74.1	75.05	86.0	88.0	87.94
0.2	1	2	61.13	63.20	66.77	72.5	74.1	75.75	86.0	88.0	88.09
0.2	42	0	61.48	63.17	65.12	72.6	74.2	75.19	86.0	87.3	87.29
0.2	42	1	61.48	63.17	64.96	72.6	74.2	74.50	86.0	87.3	87.48
0.2	42	2	61.48	63.17	65.88	72.6	74.2	74.95	86.0	87.3	87.74

Übersicht verwendeter Hilfsmittel

Während des Schreibprozesses habe ich die Anwendung ChatGPT[1] in Form des GPT-4 KI Modells verwendet. Die Verwendung fand in eingeschränkter Form als erste Orientierung und Inspiration statt. Ich habe die Anwendung nicht für das Ausfindigmachen von Literatur verwendet. Um das Problem von "Halluzinationen" und falschen Antworten einzuschränken, habe ich zusätzlich zu textuellen Anfragen an die Anwendung die zu verwendenden Quellen direkt als Datei hochgeladen. Mir nicht geläufige Fakten habe ich stets mit diesen Quellen abgeglichen und im Einklang mit dem üblichen akademischen Prozesses gekennzeichnet.

Des Weiteren habe ich zur Prüfung von Rechtschreibung und Grammatik die Anwendung *Grammarly*[2] verwendet. Da diese auch die Umformulierung von Sätzen unterstützt, führe ich sie hier als Generative KI Anwendung an. Die explizit als "Generative AI" gekennzeichnete Funktion der Anwendung habe ich jedoch nicht verwendet. Diese würde das Generieren und Umschreiben von Textbausteinen basierend auf einer textuellen Eingabe unterstützen.

[1] https://chatgpt.com/
[2] grammarly.com

List of Figures

1.1 Mutual Information between Global and Local Descriptors for CIFAR10 and CIFAR100 . . . 3

2.1 Image retrieval re-ranking process . . . 13
2.2 Overview of feature space similarity measures . . . 14

4.1 Schematic overview of information-theoretic objectives for the proposed feature extractor . . . 25
4.2 Model architecture of DELG . . . 27
4.3 Model architecture of RRT . . . 29
4.4 Model architecture without MI estimation . . . 32
4.5 Training of our model with MINE . . . 34

5.1 Distribution of the number of training images per class in the datasets . . . 41
5.2 Process of Re-Ranking Experiment . . . 44

6.1 Effect of the β penalty factor on MI between global and local descriptors 47
6.2 Comparison of methods showing the effect of β on SOP dataset . . . 47
6.3 Comparison β between main and auxiliary MINE network during training 48
6.4 Comparison β between main and auxiliary MINE network (big β values) . 49
6.5 Effect of the β penalty factor on MI between global and local descriptors during training on SOP . . . 50
6.6 Individual components of the Rényi estimation during training on SOP . 51
6.7 Measurements of MI before Attention Pooling . . . 52
6.8 Histograms of local feature attention scores . . . 53
6.9 Comparison of σ and median attention score . . . 54
6.10 Re-ranking performance improvements on Revisited Paris/Revisited Oxford using *Rényi* for MI estimation . . . 55
6.11 Comparison of re-ranking performance improvements on SOP between *Rényi* and MINE for MI estimation . . . 56
6.12 Re-ranking performance improvements on SOP using MINE for MI estimation and ResNet-50 backbone . . . 56
6.13 Distribution of the number of local features selected for re-ranking . . . 57
6.14 Re-ranking performance depending on usable sequence length . . . 58

List of Tables

5.1 Overview of the used datasets 39

6.1 DELG implementation benchmark results 45
6.2 RRT experiment benchmark results 46
6.3 Summary statistics for local feature attention scores on SOP 53

A.1 Retrieval results for individual runs on $\mathcal{R}$Oxf and $\mathcal{R}$Par using Rényi. . . . 63
A.1 Retrieval results for individual runs on $\mathcal{R}$Oxf and $\mathcal{R}$Par using Rényi. . . . 64
A.2 Retrieval results for individual runs on SOP using Rényi 64
A.2 Retrieval results for individual runs on SOP using Rényi 65
A.3 Retrieval results for individual runs on SOP using MINE 65
A.3 Retrieval results for individual runs on SOP using MINE 66
A.3 Retrieval results for individual runs on SOP using MINE 67

Acronyms

$\mathcal{R}$Oxf Revisited Oxford. 39, 40, 42, 43, 45, 55, 56, 60, 61, 63, 64, 71, 73

$\mathcal{R}$Par Revisited Paris. 39, 40, 42, 43, 45, 55, 60, 61, 63, 64, 71, 73

AP Average Precision. 38

AP@k Average Precision at rank k. 38

BCE Binary Cross-Entropy. 31

CCA Canonical Correlation Analysis. 19, 20

CE Cross-Entropy. 15, 16, 33

CNN Convolutional Neural Network. 12, 18, 22, 26, 36, 41, 42, 46, 61

DELG Deep Local and Global features. 1, 26, 31–33, 41–43, 46, 55, 61, 73

DNN Deep Neural Network. 9, 12

DV Donsker-Varadhan. 16–18

GeM Generalized Mean Pooling. 26, 31

GLDv2 Google Landmarks Dataset v2. 39, 41–43, 47, 55, 60

IB Information Bottleneck. 3, 9, 16, 20, 21

KL Kullback-Leibler. 8, 15, 16, 48

mAP Mean Average Precision. 40, 45, 55

mAP@k Mean Average Precision at rank k. 38

MHA Multi-Head Attention. 30

MI Mutual Information. 4, 8, 15, 16, 34, 36, 37, 42, 43, 46–50, 55–57, 59–61, 71

MINE Mutual Information Neural Estimation. 2–4, 15, 16, 18, 33–37, 41, 42, 46–50, 53, 55, 58–61, 65–67, 73

MSE Mean squared error. 28

P@k Precision at rank k. 37, 38

PCA Principal Pomponent Analysis. 28

R@k Recall at rank k. 38

RANSAC Random Sample Consensus. 11, 12

RRT Reranking Transformer. 3, 4, 22, 26, 29–32, 37, 39, 41–43, 46, 55–58, 60–62, 73

SOP Stanford Online Products. 31, 32, 40–43, 46, 47, 50, 51, 53, 55–58, 60, 61, 64–67, 71, 73

Bibliography

[1] S. Akaho. A kernel method for canonical correlation analysis. In *International Meeting of Psychometric Society*. arXiv, Feb. 2007.

[2] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations*, 2017.

[3] G. Andrew, R. Arora, J. Bilmes, and K. Livescu. Deep Canonical Correlation Analysis. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1247–1255. PMLR, May 2013.

[4] J. Ansel, E. Yang, H. He, N. Gimelshein, A. Jain, M. Voznesensky, B. Bao, P. Bell, D. Berard, E. Burovski, G. Chauhan, A. Chourdia, W. Constable, A. Desmaison, Z. DeVito, E. Ellison, W. Feng, J. Gong, M. Gschwind, B. Hirsh, S. Huang, K. Kalambarkar, L. Kirsch, M. Lazos, M. Lezcano, Y. Liang, J. Liang, Y. Lu, C. K. Luk, B. Maher, Y. Pan, C. Puhrsch, M. Reso, M. Saroufim, M. Y. Siraichi, H. Suk, S. Zhang, M. Suo, P. Tillet, X. Zhao, E. Wang, K. Zhou, R. Zou, X. Wang, A. Mathews, W. Wen, G. Chanan, P. Wu, and S. Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pages 929–947, La Jolla CA USA, Apr. 2024. ACM.

[5] R. Arora and K. Livescu. Kernel CCA for multi-view learning of acoustic features using articulatory measurements. In *Proc. MLSLP 2012*, pages 34–37, 2012.

[6] Y. Avrithis and G. Tolias. Hough Pyramid Matching: Speeded-Up Geometry Reranking for Large Scale Image Retrieval. *International Journal of Computer Vision*, 107(1):1–19, Mar. 2014.

[7] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer.

[8] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, and D. Hjelm. Mutual Information Neural Estimation. In *Proceedings of the 35th International Conference on Machine Learning*, pages 531–540. PMLR, July 2018.

[9] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, Aug. 2013.

[10] M. Boudiaf, J. Rony, I. M. Ziko, E. Granger, M. Pedersoli, P. Piantanida, and I. B. Ayed. A Unifying Mutual Information View of Metric Learning: Cross-Entropy vs. Pairwise Losses. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 548–564, Cham, 2020. Springer International Publishing.

[11] H. Cantzler. Random sample consensus (ransac). *Institute for Perception, Action and Behaviour, Division of Informatics, University of Edinburgh*, 3, 1981.

[12] B. Cao, A. Araujo, and J. Sim. Unifying Deep Local and Global Features for Image Search. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 726–743, Cham, 2020. Springer International Publishing.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-End Object Detection with Transformers. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing.

[14] W. Chen, Y. Liu, W. Wang, E. M. Bakker, T. Georgiou, P. Fieguth, L. Liu, and M. S. Lew. Deep Learning for Instance Retrieval: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7270–7292, June 2023.

[15] P. Cheng, W. Hao, S. Dai, J. Liu, Z. Gan, and L. Carin. CLUB: A Contrastive Log-ratio Upper Bound of Mutual Information. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1779–1788. PMLR, Nov. 2020.

[16] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct. 2007.

[17] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On Kernel Target Alignment. In J. Kacprzyk, D. E. Holmes, and L. C. Jain, editors, *Innovations in Machine Learning*, volume 194, pages 205–256, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[18] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694, June 2019.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[20] Z. Dong, C. Jing, M. Pei, and Y. Jia. Deep CNN based binary hash video representations for face retrieval. *Pattern Recognition*, 81:357–369, Sept. 2018.

[21] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain markov process expectations for large time. IV. *Communications on Pure and Applied Mathematics*, 36(2):183–212, 1983.

[22] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, Oct. 2020.

[23] A. El-Nouby, N. Neverova, I. Laptev, and H. Jégou. Training Vision Transformers for Image Retrieval, Feb. 2021.

[24] M. Federici, A. Dutta, P. Forré, N. Kushman, and Z. Akata. Learning Robust Representations via Multi-View Information Bottleneck. In *International Conference on Learning Representations*, Sept. 2019.

[25] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT press, 2016.

[26] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. End-to-End Learning of Deep Visual Representations for Image Retrieval. *International Journal of Computer Vision*, 124(2):237–254, Sept. 2017.

[27] A. Gordo, F. Radenovic, and T. Berg. Attention-Based Query Expansion Learning. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 172–188, Cham, 2020. Springer International Publishing.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[29] HAROLD. HOTELLING. RELATIONS BETWEEN TWO SETS OF VARIATES*. *Biometrika*, 28(3-4):321–377, Dec. 1936.

[30] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456. PMLR, June 2015.

[31] H. Jónsson, G. Cherubini, and E. Eleftheriou. Convergence Behavior of DNNs with Mutual-Information-Based Regularization. *Entropy*, 22(7):727, July 2020.

[32] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, Jan. 2017.

[33] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, 2009.

[34] Y. Li, M. Yang, and Z. Zhang. A Survey of Multi-View Representation Learning. *IEEE Transactions on Knowledge and Data Engineering*, 31(10):1863–1883, Oct. 2019.

[35] Y. Liu, L. Ding, C. Chen, and Y. Liu. Similarity-Based Unsupervised Deep Transfer Learning for Remote Sensing Image Retrieval. *IEEE Transactions on Geoscience and Remote Sensing*, 58(11):7872–7889, Nov. 2020.

[36] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, Sept. 1999.

[37] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[38] D. McAllester and K. Stratos. Formal Limitations on the Measurement of Mutual Information. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 875–884. PMLR, June 2020.

[39] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng. Multimodal deep learning. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, Icml '11, pages 689–696, New York, NY, USA, June 2011. ACM.

[40] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating Divergence Functionals and the Likelihood Ratio by Convex Risk Minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, Nov. 2010.

[41] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-Scale Image Retrieval with Attentive Deep Local Features. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3476–3485, Oct. 2017.

[42] S. Nowozin, B. Cseke, and R. Tomioka. F-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[43] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[44] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[45] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[46] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[47] B. Poole, S. Ozair, A. V. D. Oord, A. Alemi, and G. Tucker. On Variational Bounds of Mutual Information. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5171–5180. PMLR, May 2019.

[48] F. Radenovic, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5706–5715, June 2018.

[49] F. Radenović, G. Tolias, and O. Chum. Fine-Tuning CNN Image Retrieval with No Human Annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1655–1668, July 2019.

[50] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. [Paper] Visual Instance Retrieval with Deep Convolutional Networks. *ITE Transactions on Media Technology and Applications*, 4(3):251–258, 2016.

[51] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[52] A. Rényi. On Measures of Entropy and Information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, volume 4.1, pages 547–562. University of California Press, Jan. 1961.

[53] L. G. Sanchez Giraldo, M. Rao, and J. C. Principe. Measures of Entropy From Data Using Infinitely Divisible Kernels. *IEEE Transactions on Information Theory*, 61(1):535–548, Jan. 2015.

[54] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948.

[55] S. Shao, K. Chen, A. Karpur, Q. Cui, A. Araujo, and B. Cao. Global Features are All You Need for Image Retrieval and Reranking. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11002–11012, Oct. 2023.

[56] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, Aug. 2000.

[57] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Routledge, New York, Oct. 2017.

[58] O. Siméoni, Y. Avrithis, and O. Chum. Local Features and Visual Words Emerge in Activations. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11643–11652. IEEE Computer Society, June 2019.

[59] Sivic and Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1470–1477 vol.2, Oct. 2003.

[60] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep Metric Learning via Lifted Structured Feature Embedding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4004–4012, June 2016.

[61] S. Su, C. Zhang, K. Han, and Y. Tian. Greedy Hash: Towards Fast Optimization for Accurate Hash Coding in CNN. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[62] F. Tan, J. Yuan, and V. Ordonez. Instance-level Image Retrieval using Reranking Transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12085–12095, Oct. 2021.

[63] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37-Th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[64] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5, Apr. 2015.

[65] A. van den Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding, Jan. 2019.

[66] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. ukasz Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[67] Z. Wan, C. Zhang, P. Zhu, and Q. Hu. Multi-View Information-Bottleneck Representation Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(11):10085–10092, May 2021.

[68] Z. Wan, C. Zhang, P. Zhu, and Q. Hu. Multi-View Information-Bottleneck Representation Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10085–10092, May 2021.

[69] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille. NormFace: L2 Hypersphere Embedding for Face Verification. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 1041–1049, New York, NY, USA, Oct. 2017. Association for Computing Machinery.

[70] Q. Wang, C. Boudreau, Q. Luo, P.-N. Tan, and J. Zhou. Deep Multi-view Information Bottleneck. In *Proceedings of the 2019 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 37–45. Society for Industrial and Applied Mathematics, May 2019.

[71] W. Wang, R. Arora, K. Livescu, and J. Bilmes. On Deep Multi-View Representation Learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1083–1092. PMLR, June 2015.

[72] P. Weinzaepfel, T. Lucas, D. Larlus, and Y. Kalantidis. Learning Super-Features for Image Retrieval. In *International Conference on Learning Representations*, Oct. 2021.

[73] T. Weyand, A. Araujo, B. Cao, and J. Sim. Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2575–2584, 2020.

[74] K. K. Wickstrøm, S. Løkse, M. C. Kampffmeyer, S. Yu, J. C. Príncipe, and R. Jenssen. Analysis of Deep Convolutional Neural Networks Using Tensor Kernels and Matrix-Based Entropy. *Entropy*, 25(6):899, June 2023.

[75] C. Xu, D. Tao, and C. Xu. Large-Margin Multi-ViewInformation Bottleneck. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1559–1572, Aug. 2014.

[76] M. Yang, D. He, M. Fan, B. Shi, X. Xue, F. Li, E. Ding, and J. Huang. DOLG: Single-Stage Image Retrieval with Deep Orthogonal Fusion of Local and Global Features. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11752–11761, Oct. 2021.

[77] S. Yokoo, K. Ozaki, E. Simo-Serra, and S. Iizuka. Two-Stage Discriminative Re-Ranking for Large-Scale Landmark Retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1012–1013, 2020.

[78] S. Yu, F. Alesiani, X. Yu, R. Jenssen, and J. Principe. Measuring Dependence with Matrix-based Entropy Functional. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10781–10789, May 2021.

[79] S. Yu, L. G. S. Giraldo, R. Jenssen, and J. C. Príncipe. Multivariate Extension of Matrix-Based Rényi's $A\alpha$-Order Entropy Functional. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(11):2960–2966, Nov. 2020.

[80] S. Yu, K. Wickstrøm, R. Jenssen, and J. C. Príncipe. Understanding Convolutional Neural Networks With Information Theory: An Initial Exploration. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):435–442, Jan. 2021.

[81] Q. Zhang, S. Yu, J. Xin, and B. Chen. Multi-View Information Bottleneck Without Variational Approximation. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4318–4322, May 2022.

[82] J. Zhao, X. Xie, X. Xu, and S. Sun. Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54, Nov. 2017.

[83] W. Zhao, H. Luo, J. Peng, and J. Fan. Spatial pyramid deep hashing for large-scale image retrieval. *Neurocomputing*, 243:166–173, June 2017.