ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

UGQE:
UNCERTAINTY GUIDED QUERY EXPANSION IN IMAGE RETRIEVAL

M.Sc. THESIS

Fırat ÖNCEL

Department of Computer Engineering

Computer Engineering Programme

JUNE 2022

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

UGQE: UNCERTAINTY GUIDED QUERY EXPANSION IN IMAGE RETRIEVAL

M.Sc. THESIS

Fırat ÖNCEL
(504191518)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Prof. Dr. Gözde ÜNAL

JUNE 2022

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BYSG:
GÖRÜNTÜ ERİŞİMİNDE BELİRSİZLİK YÖNLENDİRMELİ SORGU GENİŞLETME

YÜKSEK LİSANS TEZİ

Fırat ÖNCEL
(504191518)

Bilgisayar Mühendisligi Anabilim Dalı

Bilgisayar Mühendisligi Programı

Tez Danışmanı: Prof. Dr. Gözde ÜNAL

HAZİRAN 2022

Fırat ÖNCEL, a M.Sc. student of ITU Graduate School student ID 504191518 successfully defended the thesis entitled "UGQE: UNCERTAINTY GUIDED QUERY EXPANSION IN IMAGE RETRIEVAL", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Gözde ÜNAL** İstanbul Technical University	.....
Jury Members :	**Prof. Dr. Uluğ BAYAZIT** İstanbul Technical University	.....
	Assoc. Prof. Dr. Aykut ERDEM Koç University	.....
		.....

Date of Submission : **03 June 2022**
Date of Defense : **22 June 2022**

To my family,

FOREWORD

First of all, I would like to thank my advisor Prof. Dr. Gözde ÜNAL for her endless support and guidance throught this thesis. Also, I would like to thank my dear friend Mehmet AYGÜN who helped me in this study even though we always worked remotely. My dear friend Gülçin BAYKAL CAN has put a lot effort and mental support during this study. I would like to thank my family; my mother Mukaddes, my father Hasan and my brother Tuna, for their endless support throughout my entire life. They have always supported my all decisions. Also, I would like to thank my all teachers from primary school to university for their support. Finally, I want to express my deepest gratitude to my other half, Bengisu, for her endless support, without her this thesis would not be completed. I would like to thank all of the members of ITU Vision Laboratory and my friends.

I am supported by 2210A-National Scholarship Programme for MSc Students 2019/2 from the Scientific and Technological Research Council of Turkey (TÜBİTAK) and I am grateful for that.

June 2022

Fırat ÖNCEL
Computer Engineer

TABLE OF CONTENTS

ABBREVIATIONS

CNN	**:** Convolutional Neural Network
DNN	**:** Deep Neural Network
ANN	**:** Artificial Neural Network
NLP	**:** Natural Language Processing
NMT	**:** Neural Machine Translation
QE	**:** Query Expansion
LAttQE	**:** Learnable Attention-based Query Expansion
UGQE	**:** Uncertainty Guided Query Expansion
mAP	**:** Mean Average Precision

SYMBOLS

$\mathbf{q}$	**:** Query Vector
ϕ	**:** CNN Feature Extractor

LIST OF TABLES

LIST OF FIGURES

UGQE:
UNCERTAINTY GUIDED QUERY EXPANSION IN IMAGE RETRIEVAL

SUMMARY

Image Retrieval is one of the important subproblems in Computer Vision domain. A typical image retrieval pipeline consists of a feature extractor and a search operation in image database with a given similarity measure. With the dominance of deep learning, hand-crafted feature extraction techniques are replaced with Convolutional Neural Network (CNN) based feature extractors. Images are represented with those extracted features.

Sometimes, when a query is made in an image database, some of the retrieved images may be irrelevant to the query image. Those images should be eliminated in order to improve the performance of the image retrieval systems. Query expansion is one of the ways to perform that operation. Query expansion can be considered as making a second search after the retrieved images of the first search are aggregated with the query image. The aggregation can be done in several ways such as taking an average or a weighted average. However, classical query expansion techniques have some drawbacks such as indistinctness between relevant and irrelevant neighbors or monotonic weight assignments. Existing approaches in query expansion did not consider reliability of neighbors in selecting and executing the expansion operation. Reliability per se is not straightforward to measure, however, it can be estimated as inversely proportional to the amount of uncertainty inherent in the neighbor selection. With the advent of neural network based function approximators, an uncertainty quantification can be integrated into standard neural networks that adds an ability of saying "I do not know" or "I am not certain" about this outcome.

In this thesis we integrate a pair-wise uncertainty quantification into the query expansion process in order to generate new features via a novel Uncertainty Guided Transformer Encoders (UGTE) method. Those newly generated features are concatenated with original features to enrich the overall feature representations. Then those feature representations are fed into the Learnable Attention Based Transformer Encoders (LABTE) to assign weights to neighbors. Our method consists of UGTE and LABTE: first we generate new features with UGTE, then assign new weights to the neighbors with LABTE.

Experimental results show that our proposed method increases the performance of the system relative to the baseline method which consists of only the LABTE framework, over standard image retrieval benchmarks. We utilize a CNN feature extractor, which is trained on Google Landmarks dataset. To extract the features of the transformer encoder, the train dataset that is utilized is rSfM120k, while the method is tested with datasets: rOxford5k, rParis6k and 1 M Distractors.

BYSG:
GÖRÜNTÜ ERİŞİMİNDE
BELİRSİZLİK YÖNLENDİRMELİ SORGU GENİŞLETME

ÖZET

Görüntü erişimi, bilgisayarla görü alanının önemli alt problemlerinden birisidir. Bu problemde amaç görüntüleri vektör düzeyinde temsil etmek ve daha sonra bu vektörler arasındaki olması gereken yakınlık, aynı nesnenin farklı açılardan edinilmiş görüntüleri arasında kurulması gereken bağlantı, ve uzaklık, farklı nesneler arasında kurulması gereken bağlantı, özelliklerinden yararlanılarak verilen bir görüntünün bir görüntü veri tabanınından benzer görüntünün elde edilmesi olarak tanımlanabilir. Derin Öğrenme öncesi dönemde görüntülerin vektör düzeyinde ifade edilmesi için genellikle geleneksel olarak nitelendirilebilecek, kelime çantası modeli veya yerel öznitelik eşleştirme gibi, yöntemler makine öğrenmesi yöntemleri ile birleştirilerek problem giderilmeye çalışılmıştır.

Derin öğrenme yöntemlerinin yaygınlaşmasıyla birlikte birçok alanda olduğu gibi bilgisayarla görü alanında klasik yöntemlerin yerini derin öğrenme tabanlı yöntemler almaya başlamıştır. Derin öğrenme tabanlı yöntemlerde de görüntüler için en çok kullanılan yöntem Evrişimli Sinir Ağları (ESA) içermektedir. ESA'lar görüntülerdeki örüntüleri bulmada ve öznitelik çıkarmada en çok kullanılan yöntemdir. ESA'lar görüntülerdeki uzamsal bağlantıları kullanarak başarılı çalışmaktadırlar. ESA'lar sınıflandırma, bölütleme gibi temel bilgisayarla görü problemlerinin yanında görüntü erişimi probleminde de başarıyla çalışmaktadır.

Bu tezdeki görüntü erişimi problemi ele alınırken kent simgelerini içeren veri kümeleri kullanılmıştır. Google Landmarks [3] adı verilen ve tüm dünyadan 1 milyonun üzerinde görüntü içeren veri kümesi kullanılmıştır. Bu görüntü kümesi kullanılarak önce ESA katmanlarının bittiği kısımdan çıktılar alınmış, örneğin 7x7x2048 büyüklüğünde, ve bu çıktılar daha havuzlama yöntemiyle, GeM [4], birlikte 2048 boyutlu vektörlerle görüntülerin temsili sağlanmıştır. Daha sonra bu görüntü temsilleri sağlanarak seçilen bir eğitim demetinde, sorgu, pozitif örnek ve negatif örnekler, karşılaştırmalı hata fonksiyonu kullanılarak ESA eğitilmiştir ve öznitelik çıkarıcı diyeceğimiz ağ yaratılmıştır. Bu çalışmada öznitelik çıkarıcı ağ olarak ResNet101 [5] kullanılmıştır ve bu ağ tezin ele aldığı esas problemde kullanılmak üzere kaydedilmiştir.

Sorgu genişletme tekniği, görüntü erişiminde; ilk sorgunun görüntü veri tabanındaki en yakın komşularını elde ettikten sonra, bu en yakın komşuların öznitelik vektörlerinin kullanılarak oluşturulun geliştirilmiş sorgunun tekrar görüntü veri tabanında aranmasıyla oluşturulur. Sorgu genişletme tekniğinde amaç ilk sorguda olmayan niteliklerin görüntü veri tabanından gelen en yakın komşu niteliklerle birlikte zenginleştirilerek daha yüksek başarımlı sorgu sonuçlarına ulaşılmasına dayanmaktadır.

Sorgu genişletme tekniğinde ilk sorgu sonrasında elde edilen en yakın komşu görüntülerin öznitelik vektörlerinin ne şekilde birleştirileceği önemli bir araştırma konusu olmuştur. Temel olarak kullanılan yöntem sorgunun ve elde edilen k tane en yakın komşu vektörünün ortalaması alınmasıdır. Bunun yanı sıra komşuların azalan oranda etki edilmesi ve benzerlik oranları ile çarpılarak ağırlıklı ortalamasının alınması diğer yöntemlerdir. Bu yöntemler öznitelik çıkarıcı ağın kalitesine, sorgunun yapıldığı görüntü veri tabanındaki görüntülerin zorluğuna bağlı olarak elde edilen komşuların aslında gerçekten komşu olmadığı durumlarda yetersiz kalmıştır. Komşu olmayan vektörlerin genişleştilmiş vektöre katılması sistemin performansını düşürmektedir. Ayrıca klasik yöntemlerle yapılan sorgu genişletme işlemlerinin bir diğer yetersiz kaldığı konu da ağırlıkların en yakın komşudan en uzak komşuya doğru azalarak gitmesidir, bunun dışında bir ağırlıklandırmaya izin vermemeleridir.

Sorgu genişletme tekniğindeki bu sorunu çözmek adına Gordo ve arkadaşları 2020 yılında "Dikkat-Tabanlı Sorgu Genişletmeyi Öğrenme" adlı çalışmayla birlikte sorgu ve en yakın komşu vektörlerini bir araya getirirken kullanılan ağırlıkların dönüştürücü kodlayıcı model ile birlikte öğrenilmesi amaçlanmıştır. Bu çalışmadaki amaç aslında komşu olmadığı halde yakın öznitelik vektörleri bakımından yakın gözüken vektörlerin genişletilmiş sorgunun oluşturulması sırasında devre dışında bırakılmasdır. Ayrıca önerdikleri bu sistemde en yakın komşulara atanan ağırlıklar azalarak gitmemekte, herhangi bir kısıt bulunmamaktadır.

Sınıflandırma probleminde belirsizlik ölçme yöntemi, yani bir yapay sinir ağının daha önce görmediği bir sınıf hakkında bilmiyorum diyebilme özelliği kazandırma özelliği olarak değerlendirilebilir. Standart hata fonksiyonları ile eğitilen yapay sinir ağları bu noktada yetersiz kalacağı için Şensoy ve arkadaşları 2018 yılında Kanıtsal Derin Öğrenme tekniğini önererek yapay sinir ağlarına belirsizlik ölçümünde prensipli bir yöntem kazandırmıştır.

Bu teze konu olan çalışmada "Dikkat-Tabanlı Sorgu Genişletmeyi Öğrenme" yöntemi temel alınarak bir sorgu ve onun en yakın komşuları arasındaki belirsizlik faydalanılarak var olan öznitelik vektörlerine en olarak yeni öznitelikler üreten bir model kurulmuştur. Bunu yapmak için öznitelik çıkarıcı ESA alınarak rSfM120k veri kümesindeki tüm görüntülerin öznitelikleri çıkarılmıştır. Daha sonra bu veri kümesindeki sorgu, pozitif ve negatif örnekler demeti kullanılarak, "Belirsizlik Yönlendirmeli Dönüştürücü Kodlayıcı" adı verilen model Kanıtsal Derin Öğrenme yöntemi kullanılarak eğitilmiş ve bu model tarafından çıkarılan yeni öznitelikler var olan öznitelik vektörlerine eklenmiştir. Bu artırılmış öznitelik vektörleri kullanılarak daha sonra "Öğrenilebilen Dikkat Tabanlı Sorgu Genişletme" modeli eğitilmiştir.

Uçtan uca eğitilen bu iki modelin birleştirilmesiyle birlikte olan yönteme de "Belirsizlik Yönlendirmeli Sorgu Genişletme" yöntemi ismi verilmiştir. rOxford5k ve rParis6k test veri kümelerinde, öznitelik vektörü çıkaran ESA ve Dönüştürücü Kodlayıcı eğitiminde kullanılan rSfM120k veri kümelerinde yer almayan resimlerden oluşmakta, yapılan deneylerle birlikte klasik yöntemlerden daha iyi sonuç verdiği deneysel olarak gösterilmiştir. Ayrıca yeni öznitelikler eğitilmeden eğitilen "Öğrenilebilen Dikkat Tabanlı Sorgu Genişletme" modeline nazaran, belirsizlik tabanlı yeni özniteliklerin eklendiği artırılmış öznitelik vektörleri ile eğitilen "Öğrenilebilen Dikkat Tabanlı Sorgu Genişletme" modelinin daha iyi sonuç verdiği deneysel olarak ortaya konmuştur.

Ayrıca rOxford5k ve rParis6k test görüntü veri kümelerine eklenen 1 milyon dağıtıcı görüntünün eklenmesiyle ve tüm görüntü veri kümesi vektörlerinin yine görüntü veri kümesinde arama yapılarak oluşturulan görüntü kümesi tarafında artırma yöntemiyle yapılan deneylerde de "Belirsizlik Yönlendirmeli Sorgu Genişletme" modelinin üstün olduğu deneysel olarak gösterilmiştir.

1. INTRODUCTION

Image retrieval methods are mainly built on the transformation of a high dimensional input image to its low dimensional latent representation. Due to several sources of error such as occlusions, viewpoint variations, background clutters or loss of information during transformation to the vector space, the image search is enriched with a query expansion idea that depends on constructing a latent model, which is based on aggregating a collection of responses from an initial query [6,7]. On the other hand, in the recent years, image retrieval methods are dominated by Convolutional Neural Networks (CNNs), which replace hand-crafted features in feature extraction phase of image retrieval systems [3,4,8,9].

As important as a role feature extraction plays in performance of image retrieval systems, a set of related tools such as database side augmentation [10], query expansion [6], hashing [11] and so on play crucial roles in image retrieval systems. Particularly, Query Expansion (QE) is regarded as one of the most powerful tools, as it increases the performance of an image retrieval system no matter how the features are extracted [1,6]. The basic idea of a QE method is to enhance the quality of the query vector through an augmentation of the query vector space using some priors. The latter is provided by an initial search that results in a collection of vectors that lead to a richer latent feature representation of the query. One of the main limitations of QE algorithms [4,6,9,12] is that the assigned weights to neighbors of the query are in monotonically decreasing order. In the recent work of Gordo et al. [1], a self-attention mechanism was used via transformer encoders [2] in order to assign weights which do not have to be in monotonically decreasing order so that irrelevant neighbors, which are not true neighbors to the query, can be eliminated via assigning them smaller weights.

While Deep Neural Networks (DNNs) have the flexibility to create and assign different weights in QE algorithms, they are famous for their overconfidently wrong predictions [13], which might hurt the quality of the expanded query feature vector. In recent years,

there has been a lot of interest on how to incorporate uncertainty estimation into deep neural network models [14]–[17] to alleviate their problem in making overconfident predictions. These works enable integrating the ability of saying "I am not sure about this prediction".

In this thesis, we integrate a dedicated pairwise uncertainty estimation between a query and each of its neighbors in the creation of an enriched image representation. The latter is in terms of image features relative to the original extracted features which in combination provide a tool in generating more powerful expanded queries. To that end, we design an uncertainty-guided transformer encoder model, which relies on and expands on the self-attention-based Learnable Attention-based Query Expansion (LAttQE) model by [1] via incorporating pairwise uncertainty that is estimated with the Evidential Deep Learning (EDL) framework [17].

Our proposed new module, which is called the Uncertainty Guided Query Expansion (UGQE), first takes the feature vector of a query and feature vectors of its retrieved top-k neighbors and estimates the pairwise uncertainty in whether the neighbors are from the same landmark with the query or not, using the EDL framework within a transformer encoder architecture. Next, our model utilizes the obtained uncertainty information in order to generate a new feature representation via concatenating transformed features and original features. Finally, the LAttQE model is used to form the expanded query by aggregating these new generated features of the nearest neighbors.

Our main contributions in this thesis can be summarized as follows:

- A method to create uncertainty guided features to enrich the original image representations.
- A demonstration of how to use EDL for quantifying the pair-wise uncertainty via a transformer encoder model.
- An overall uncertainty-guided query expansion method that improves over the baselines.

Our experiments demonstrate that the proposed UGQE method increases the performance of the traditional QE methods and outperforms the LAttQE model when

the uncertainty is integrated, in most settings. Our method is not only applied to the recently proposed LAttQE model but also to the classical QE techinques such Average QE (AQE) [6], Average QE with Decay (AQEwD) [9] and α-weighted QE (α-QE) [4] techniques.

2. BACKGROUND AND LITERATURE REVIEW

2.1 Preliminaries

2.1.1 Convolutional neural networks

CNNs have become very popular with their powerful capability of capturing the patterns in images with limited number of parameters compared to fully connected networks by exploiting the spatial relationships. In 2012, Krizhevsky et al [18] won the ImageNet challenge with their AlexNet architecture. Subsequently, numerous ubiquitous architectures are proposed such as GoogLeNet [19], VGG [20], ResNet [5] and EfficientNet [21]. In image retrieval CNNs are widely used to extract feature vectors that are representative of given images. In general, a CNN that is pre-trained on ImageNet, is fine-tuned with a metric learning setting, which requires a query, a positive sample and negative samples, to construct feature representations.

2.1.2 Pooling

Image retrieval systems make use of the pooling mechanism that is popularly used within CNN architectures. A pooling layer is typically used after a convolutional layer k output, X_k, which is of size Height (H) $\times$ Width (W), in order to reduce its dimensions. This operation is done to reduce the number of parameters to a single scalar or smaller matrices depending on the kernel size (F) and stride (S) of the pooling operation. There are several types of pooling that involve either fixed or learnable parameters. Max pooling, given in equation 2.1, where f is k dimensional feature representation of an image, and average pooling, given in equation 2.2, are the most commonly used fixed pooling layers. In the first method, max pooling, maximum value of given $H \times W$ patch is used in feature representation while in the second method, average pooling, average of the $H \times W$ patch is used. Generalized Mean Pooling (GeM) [4], is another pooling technique which has a learnable parameter p. GeM is formulated in equation 2.3. When

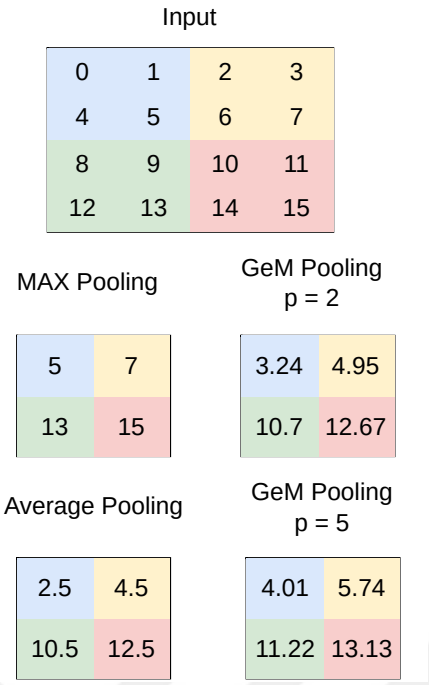

Figure 2.1 : Pooling examples.

$p = 1$, GEM reduces to average pooling, and when $p \to \infty$, it becomes max pooling. In image retrieval, usage of GeM pooling is common practice in order to extract features at the end of the last convolutional layer and it's optimized directly within the model. Suppose the output of last convolutional layer is of size Batch-Size (B) $\times$ Dimension (K) $\times$ Height (H) $\times$ Width (W). After applying the pooling operation, we get $B \times K$, K-dimensional feature vector for each element in the batch. Numerical examples with $F = 2$ and $S = 2$ are given in Figure 2.1.

$$f = [f_1, f_2, ..., f_k], f_k = \max_{x \in X_k} x \tag{2.1}$$

$$f = [f_1, f_2, ..., f_k], f_k = \frac{1}{|X_k|} \sum_{x \in X_k} x \tag{2.2}$$

$$f = [f_1, f_2, ..., f_k], f_k = (\frac{1}{|X_k|} \sum_{x \in X_k} x^{p_k})^{\frac{1}{p_k}} \tag{2.3}$$

2.1.3 Self attention

The transformer architecture, which appeared first in 2017 [2], has been very popular for the last five years. It is originally proposed for sequence-to-sequence models in neural machine translations. It comprises of a self-attention mechanism, given in equation 2.4, which simply calculates the dependencies between a query (Q) and key (K)/value (V) feature vectors through an attention distribution

$$Attention(Query(Q), Key(K), Value(V)) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.4}$$

where d_k refers to dimension of the query vector and it is used as a scaling factor.

MultiHead Attention, given in equations 2.5, 2.6 and 2.7, can be described as a concatenation of h identical attention heads as follows:

$$MultiHead(Q,K,V) = Concat(head_1, ..., head_h)W^O \tag{2.5}$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{2.6}$$

where

$$W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v} and W^Q \in \mathbb{R}^{hd_v \times d_{model}}. \tag{2.7}$$

Here, W^Q, W_K and W^V refer to the trained projection weights of query, key and value. W^0 refers to the trained projection weights of concatenation operation to calculate final attention.

To integrate order information, typically a positional encoding is added to feature vectors before feeding them into transformer encoder layers. Positional encoding can be chosen in both a learnable and a fixed way. For illustration, the attention mechanism; scaled dot-product calculation, multi-head attention and a transformer-encoder architecture, are depicted in Figures 2.2 and 2.3.

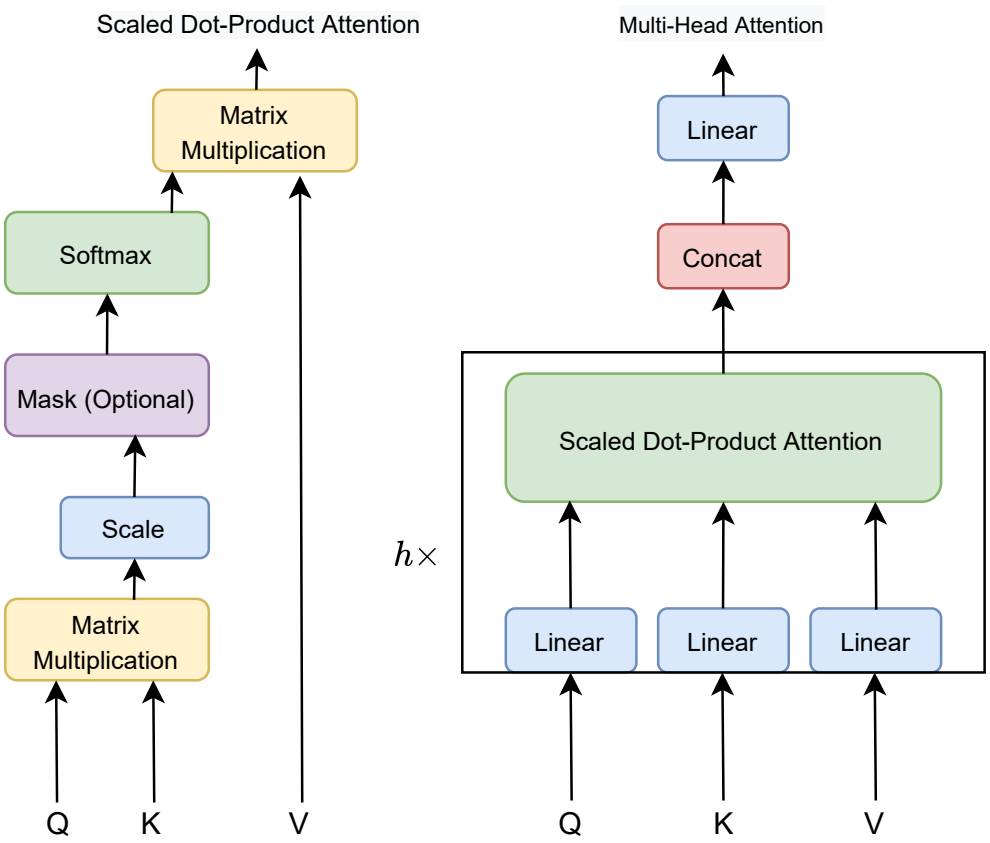

Figure 2.2 : Attention calculation [2] - self drawn.

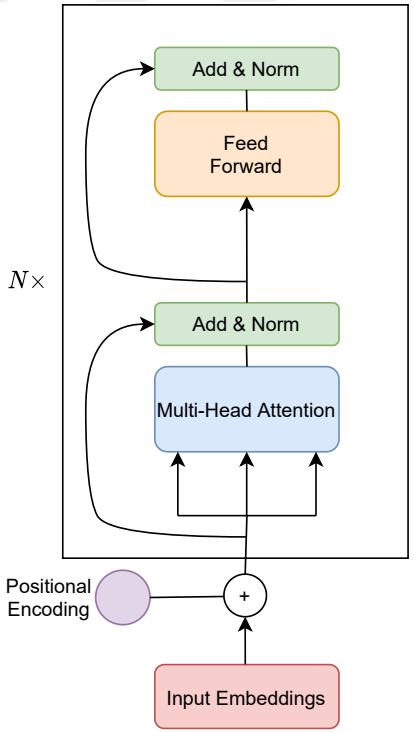

Figure 2.3 : Transformer encoder architecture [2] - self drawn.

2.2 Image Retrieval & Query Expansion

Image retrieval systems generally consist of two components:

- Extraction of a relatively compact representation or a feature vector of an image.
- Searching in the representation space, typically using a similarity measure, such as a cosine similarity.

The feature extraction component was largely based on hand-crafted feature engineering methods like SIFT [22], Fisher Vectors [23] and VLAD [24] before the deep learning era. Further extensions of those methods are also introduced for improving whole retrieval pipelines [12].

Today, CNNs are widely used in the feature extraction part of a deep image retrieval system, which is illustrated in Figure 2.4. Typically, the features are extracted from the output of the convolutional layers by applying a pooling layer, such as GeM pooling and these features are named as bottleneck features, or they are extracted from the projection of a linear layer. In the recent works, CNNs, which are typically pre-trained on Imagenet, are fine-tuned in both unsupervised [4] and supervised [3,9,25] settings for learning more efficient representations. These models are fine-tuned with a train set, which is disjoint from the test set. To improve the feature representations of query images, Chum et al. [6] introduced the idea of QE for image retrieval, which first appeared in text retrieval systems [26,27]. The QE idea is quite simple: after the first search operation on the test set, highly ranked nearest neighbor images are filtered with a spatial verification step to retain high quality results. Next, using the original query feature vector along with the feature vectors of retrieved presumably high quality results, a new expanded query representation is aggregated to enhance feature representations. For the aggregation of the original query feature vector and the retrieved feature vectors, approaches such as mean averaging and weighted averaging are utilized. While the QE methods were first introduced for hand-crafted features in image retrieval systems, deep learning-based image retrieval systems also has been using QE idea in their pipelines for improving their performance since QE methods do not rely on how features are extracted, as they are relevant to the aggregation between the original query and nearest

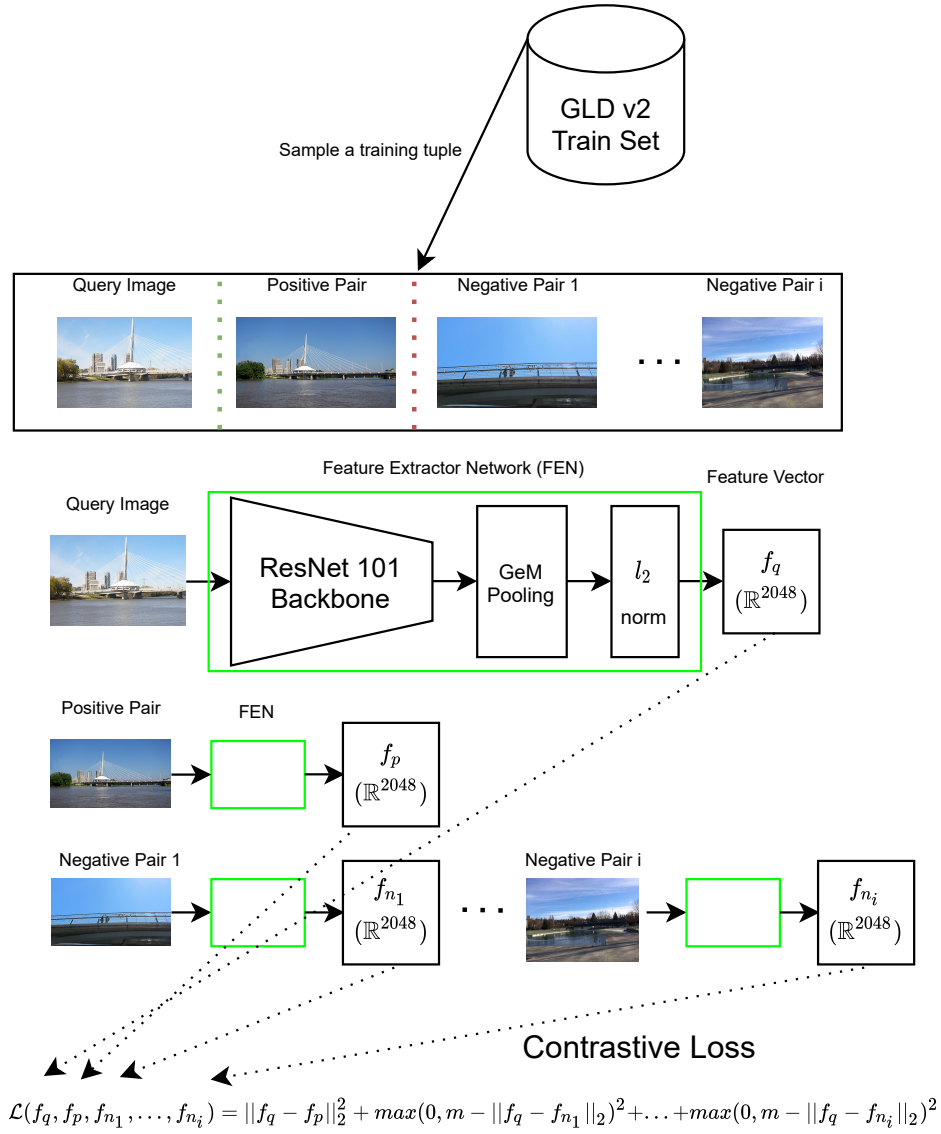

Figure 2.4 : Deep image retrieval pipeline (feature extractor).

neighbors. In situations where a nearest neighbor of a query is not an actual neighbor, it should not be included in the aggregation process. This can be considered as a drawback of classic QE methods. Moreover, lately, state-of-the-art results are obtained with QE methods that are combined with deep representations in the feature extraction phase [4,9,28], or more recently, with learning the QE weights [1] using the attention mechanism [2]. The latter method proposes a non-monotonically weighting scheme to alleviate the false neighbor problem, which is promising.

AQE [6], AQEwD [9] and α-QE [4] are the most popular standard query expansion methods. In the AQE method, the query feature vector and its nearest neighbors' feature

vectors are mean-aggregated. In the AQEwD method, nearest neighbors' feature vectors are weighted according to their order in a decreasing manner such that the query feature vector receives 1 as its weight, and other neighbors receive their weights according to the rule: $\frac{k-i}{k}$ where k is the total number of aggregated feature vectors including the query and i is the relevance order. Finally, for the α-QE method, each neighbor is weighted by its similarity to the query, with a power to the α, $sim(query, neighbor_i)^{\alpha}$.

2.3 Transformers - Vision Transformers

The transformer [2] architecture has become the default choice for most of the NLP tasks like Neural Machine Translation (NMT) [29]–[31]. This architecture can capture the dependencies between the sequential inputs with a self attention mechanism including positional encodings. After its success over text data, the transformer architecture recently also has become popular in computer vision domain. Vision Transformer [32] treats images as patches and employs the transformer architecture in classification problems.

El-Nouby et al. [33] extend vision transformers to a metric learning setting to perform image retrieval. While this work focuses on the initial representation of images, different from them in our work we focus on how to expand the initial representations using transformer encoders in the QE framework.

Graph Attention Networks [34] are also used in the metric learning setting to weigh the feature vectors in a batch [35]. Most recent and relevant work in query expansion that includes the attention setting is by Gordo et al. [1]. They utilized a self attention mechanism to weigh the nearest neighbors of the query image for integrating their information. Our work expands on the transformer idea, strengthening it with the uncertainty information. The uncertainty that we introduce is captured from the neighborhood relationships between feature vectors, and integrated into generation of an expanded query feature vector.

2.4 Uncertainty in Deep Learning

Predicting the model uncertainty is an emerging field in deep learning. MC Dropout [14] is one of the earliest and widely utilized works in the related literature. In this method,

a dropout mechanism is applied not only in training time but also in test time, while a multitude of predictions are averaged to get the prediction, and the variance among predictions is used to calculate the uncertainty. This approach presents high computation time requirements since multiple forward passes are needed. Deep Ensembles [15] and their variants are also related to MC Dropout, bearing a similar approach in incorporating an indirect uncertainty estimation into deep neural networks. In contrast to deep ensemble methods, a line of recent research work, known as EDL directly estimates both data and model uncertainty. The latter, which is also known as epistemic uncertainty, deals with how certain a neural network can be when making predictions. Sensoy et al. [17] estimate the classification uncertainty collecting evidence for each class by placing a Dirichlet distribution on the class probabilities instead of a softmax to allow the neural network models to directly quantify the uncertainty in their outputs. Amini et al. [36] extend this idea into continuous outputs by placing an Inverse Gamma distribution at the end of the regression task. In our work, we integrate the uncertainty estimation between a query image and its top-k neighbors to generate new features. With these features, the image representations are enhanced while the estimated uncertainty is integrated into the query expansion procedure. To our knowledge, our work is the first to introduce an uncertainty guidance through gauging the reliability of neighboring feature vectors into the QE framework.

3. METHOD

In this thesis, as we deploy an uncertainty quantification through the EDL approach into the Attention-Based QE setting, we give a brief overview of both frameworks.

3.1 Attention-Based Query Expansion Learning

LAttQE [1] utilizes the self-attention mechanism to form the expanded query based on the original query and its k-nearest neighbors by predicting the weights for each of the respective feature vectors. They formulate the problem as a metric learning problem. Let ϕ be a CNN feature extractor, which takes an input image, and transforms it into a D-dimensional feature vector. The query image is denoted by q, and its feature vector is denoted by $\mathbf{q} = \phi(q)$. The positive pair, the i-th negative pair and the k-th nearest neighbor of the query image are denoted as $\mathbf{p}$, $\mathbf{n_i}$ and $\mathbf{d_k}$, respectively. $\hat{\mathbf{q}}$ can be formulated as in equation 3.1.

$$\hat{\mathbf{q}} = \mathbf{w_q q} + \sum_{\mathbf{n=1}}^{\mathbf{k}} \mathbf{w_{d_k} d_k} \tag{3.1}$$

where w_q and w_{d_k} are assigned weights to the query and its neighbors, respectively. Additionally, the transformer outputs whether the retrieved neighbors are from the same landmark as the query or not. To account for that, an additional variable y is defined such that y is set to 1 if we have a positive pair, and set to 0 otherwise. The corresponding contrastive loss can be formulated as in equation 3.2:

$$L_{qe}(\hat{\mathbf{q}}, \mathbf{f}, y) = y\,\|\hat{\mathbf{q}} - \mathbf{f}\| + (1-y)\max(0, m - \|\hat{\mathbf{q}} - \mathbf{f}\|), \tag{3.2}$$

where $\mathbf{f}$ is either $\mathbf{p}$ or $\mathbf{n_i}$, and m is a selected margin.

In Figure 3.1, general overview of base method's building block Learnable Attention Based Transformer Encoder ($LABTE$) is given.

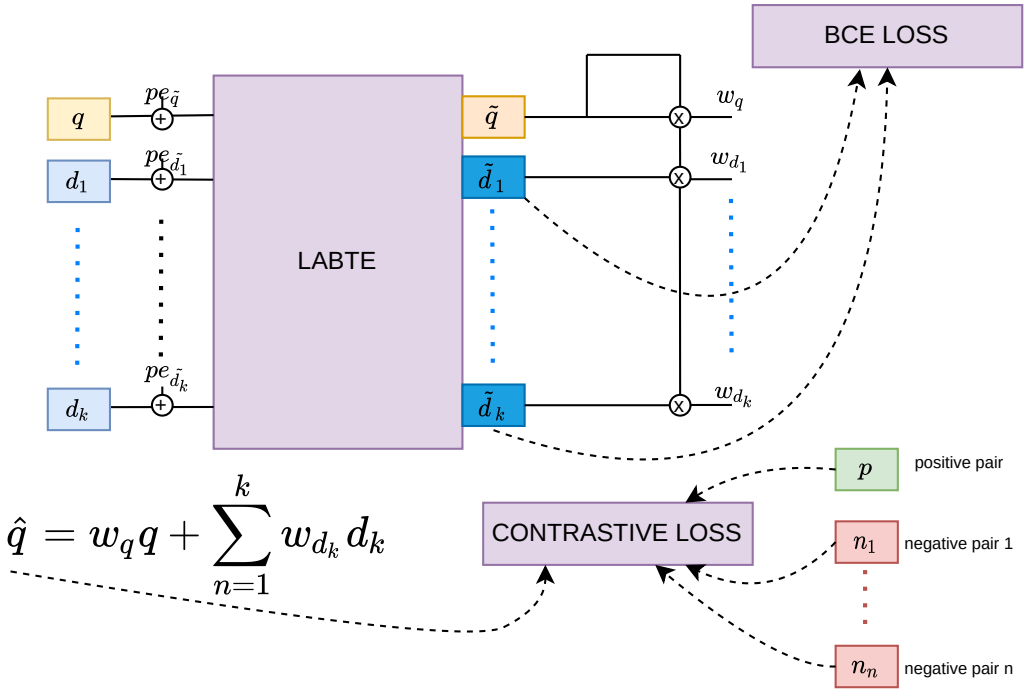

Figure 3.1 : Learnable Attention Based Transformer Encoder (*LABTE*): *LABTE* first takes the feature vectors then the positional encoding information is added. Self-normalized dot product operation is applied between output vector of query and other neighbors' vectors. Resulting expanded query vector is used in contrastive loss with positive pair and negative pairs. Output vectors are also used in Binary-Cross Entropy Loss.

3.2 Evidential Deep Learning

For estimating the uncertainty or inversely the reliability of a neighbor in contributing to the feature description of a query, we utilize the objective proposed in [17]. Let us denote the evidence collected from the k-th class of a multi-class (e.g. K-class) classification problem as e_k, and Dirichlet distribution parameters as $\alpha_k = e_k + 1$. Then the Dirichlet strength is given in equation 3.3.

$$S = \sum_{k=1}^{K} \alpha_k \tag{3.3}$$

Belief masses are defined as in equation 3.4, and the expected probability for the k-th class is given in equation 3.5.

$$b_k = \frac{e_k}{S} \tag{3.4}$$

$$\hat{p}_k = \frac{\alpha_k}{S} \tag{3.5}$$

Finally, the uncertainty in the prediction can be calculated as the residual belief remaining when we subtract the sum of our beliefs from unity, given in equation 3.6.

$$u = 1 - \sum_{k=1}^{k} b_k \tag{3.6}$$

Given N input samples and labels, the classification uncertainty can be quantified by minimizing the following objective function in equation 3.7:

$$\mathscr{L}(\Theta) = \sum_{i=1}^{N} \mathscr{L}_i(\Theta) = \sum_{i=1}^{N} \sum_{j=1}^{K} \left(y_{ij} - \hat{p}_{ij}\right)^2 + \frac{\hat{p}_{ij}\left(1 - \hat{p}_{ij}\right)}{(S_i + 1)} \tag{3.7}$$

where Θ denotes neural network parameters, y_i is a one-hot vector encoding of ground-truth of sample x_i with $y_{ik} = 0$ and $y_{ij} = 1$ for all $k \neq j$, and i denotes the index of a data sample for $i = 1, ..., N$. Here, the objective entails minimization of the sum of the squared prediction errors and an estimate of the variance in the second term to obtain the evidential distribution parameters. Although this is a non-Bayesian neural network approach, by placing evidential priors over the classification output, this framework outputs the uncertainty u in the form of what is left beyond our beliefs and collected evidence that are based on the assumed evidential distribution.

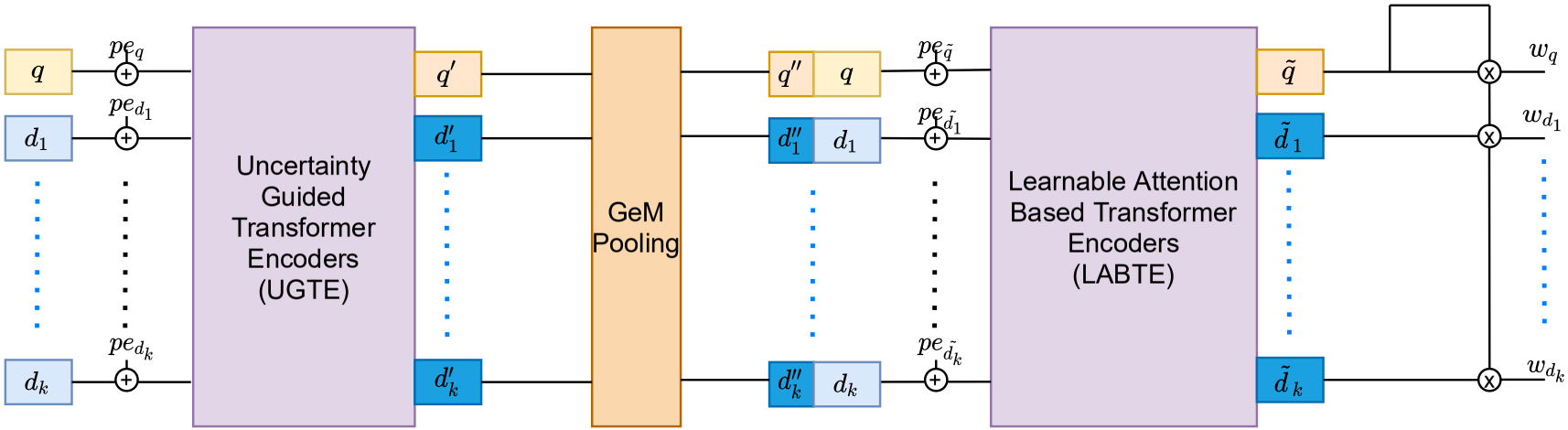

Figure 3.2 : UGTE takes input features, calculates the uncertainty between the query and each neighbor, and outputs the uncertainty guided features. Then, LABTE takes a combination of new features and original features, and outputs the weights for the query and its nearest neighbors to generate the expanded query.

3.3 Uncertainty Guided Query Expansion Learning

Leveraging on the evidential distribution idea, we propose the Uncertainty Guided Query Expansion (UGQE) model, which adapts and fuses ideas from both the EDL and LAttQE to create an improved attention-based architecture that enables and exploits uncertainty learning in query expansion. Our complete UGQE model is depicted in Figure 3.2. UGQE involves two transformer encoders, the first one generates the uncertainty guided features. The first transformer encoder integrates a pairwise uncertainty quantification, by employing a $K = 2$-class evidential classification, which outputs a target label y that is 1 if the i-th neighbor $\mathbf{d_i}$ of the query $\mathbf{q}$ is relevant, and 0 otherwise. The first transformer is trained end-to-end with the second transformer encoder, i.e. the LAttQE architecture, to produce the final weights that are used in the query expansion.

Algorithm 1: Uncertainty Guided Query Expansion (UGQE)

input : Learnable Attention Based Transformer Encoder ($LABTE$)
Uncertainty Guided Transformer Encoder ($UGTE$)
features of query and k-nearest neighbors: $F = \{\mathbf{q}, \mathbf{d_1}, \mathbf{d_2}, ..., \mathbf{d_k}\}$
positional encodings: $PE = \{\mathbf{pe_q}, \mathbf{pe_{d_1}}, \mathbf{pe_{d_2}}, ..., \mathbf{pe_{d_k}}\}$
output : expanded query: $\hat{\mathbf{q}}$

1 $F' \leftarrow UGTE([F; PE])$
2 $F' \leftarrow GeMPooling(F')$
3 $\{\tilde{\mathbf{q}}, \tilde{\mathbf{d_1}}, \tilde{\mathbf{d_2}}, ..., \tilde{\mathbf{d_k}}\} \leftarrow LABTE([F'; F])$
4 **for** $i \leftarrow 1$ **to** K **do**
5 $\quad w_i \leftarrow normalizeddotproduct(\tilde{q}, \tilde{d}_i)$
6 **end**
7 $\hat{\mathbf{q}} \leftarrow \mathbf{q}$
8 **for** $i \leftarrow 1$ **to** K **do**
9 $\quad \hat{\mathbf{q}} = \hat{\mathbf{q}} + w_i \mathbf{d_i}$
10 **end**
11 **return** $\hat{\mathbf{q}}$

The details of the UGQE algorithm is described in Algorithm 1. Input feature vectors, which are the query ($\hat{\mathbf{q}}$) and its top-k nearest neighbors ($\mathbf{n_1}, \mathbf{n_2}, ..., \mathbf{n_k}$), are 2048-Dimensional. The output vectors of this model also have a size of 2048, same as the input size. Our model takes these features (line 1 in Algorithm 1), as inputs and then sends the output vectors to the GeM Pooling layer [4], which reduces the dimensions of vectors from 2048-Dimensional (2048D) to 512D (line 2 in Algorithm 1). As the point

of this model is to learn informative features, we observe that the reduced features are able to retain the useful information in the original features. These 512D features, which are later concatenated to the original features, are fed to a Multilayer Perceptron (MLP). Then the EDL Loss, given in equation 3.7, is minimized, and an uncertainty estimate is produced for each neighbor, which is a quantity that signifies a kind of confidence in the neighbor status of each neighbor. Then, the obtained uncertainty feature vectors are concatenated to the original features to be input to the second transformer, i.e. the LAttQE model (line 3 in Algorithm 1). LAttQE is trained end-to-end with the first transformer encoder, and the output of the overall model are the weights of the query and its nearest neighbors that provide construction of the final expanded query ($\hat{\mathbf{q}}$) (lines 4-10 in Algorithm 1).

In Figure 3.3, our proposed Uncertainty Guided Transformer Encoder ($UGTE$) block is given.

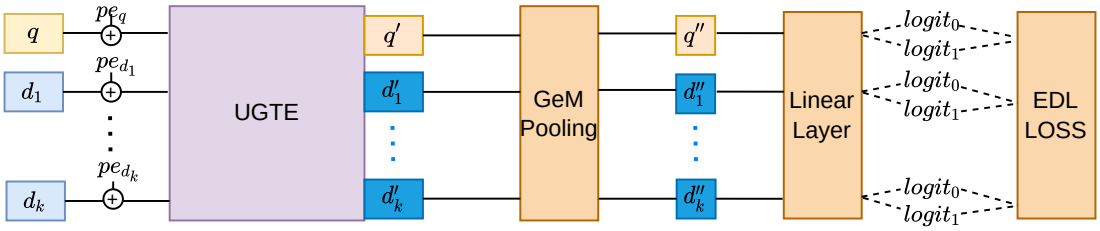

Figure 3.3 : Uncertainty Guided Transformer Encoder ($UGTE$): $UGTE$ takes feature vectors and produces output vectors which are later GeM pooled. Pooled vectors are used in EDL loss and also concatenated with the original features to be later used in LABTE.

3.4 Implementation Details

UGTE model has 2 attention layers (each one with 64 heads) and created features are 512D. Only the EDL Loss is used to train this model. We use Adam optimizer [37] with an initial learning rate of $1e^{-4}$, a weight decay of $1e^{-6}$, and an exponential learning rate scheduler with a decay of 0.99. Then we concatenate the original features with the created ones to train the second self-attention model, which is the LAttQE, which has 3 attention layers (each one with 64 heads). We use the contrastive loss given in equation 3.2, with a margin parameter of 2.1, and the binary-cross entropy loss to train this model. In the implementation of the EDL, we use the exponential activation layer, instead

of softmax which is used in standard neural networks, at the end of our uncertainty MLP block to ensure positive output. ReLU and Softplus can be also used, too. We use 2048 queries per epoch and 15 negative samples, which is chosen empirically, for each positive sample, selected from a pool of 40000 samples. Choice of the number of negative samples for a positive sample and the margin is essential as the loss is usually 0 with a low number, i.e. 5 or 10, of negative samples, and a low margin, i.e. 0.1. Queries and the pool of samples are updated at every epoch. We also use Adam optimizer with an initial learning rate of $1e^{-4}$, a weight decay of $1e^{-6}$, and an exponential learning rate scheduler with a decay of 0.99 for the second transformer. We train our setting for 300 epochs with a batch size of 64, and 127 neighbors. We do not use neighborhood dropping, which is proposed in [1], as it does not improve the scores in our experiments. Both models have learnable positional encodings to integrate the positional information.

As there is no publicly available official implementation of [1] [1], we implemented the LAttQE architecture, which constitutes our baseline. Using the original hyperparameters reported in the paper[2], slightly lower performance values are obtained than the reported results. We report both results in our paper, however, due to reproducibility concerns, we take our implementation as the baseline to compare it with the proposed UGQE.

[1] https://github.com/filipradenovic/cnnimageretrieval-pytorch/issues/68

[2] Unfortunately it was not possible to reproduce the performance results reported in the paper. This is also reported by researchers via issues opened in the github repo of the paper.

4. EXPERIMENTS AND RESULTS

4.1 Training Datasets

4.1.1 Google landmarks dataset

Google Landmarks Dataset v1 [38] is used in the training of the selected Convolutional Neural Network, which is used as a feature extractor. In order to make a comparison to our baseline model, we utilize the pre-trained ResNet101 [5] from Python Image Retrieval Toolbox [1], which was trained on the Google Landmarks Dataset v1. The latter consists of nearly 1.2 million train images with over 10 000 landmarks all around the world. With the index set, it has 2.3 million images with over 30 000 landmarks in total. This dataset is very diverse and extensive for training good feature extractors for image retrieval. Later in 2019, a second version of this dataset, Google Landmarks Dataset v2 [39], is published. The second version is more comprehensive, as it comprises 4.1 million train images and 5 million images with over 200 000 landmarks in total. Sample images from the dataset are given in Figure 4.1.

4.1.2 rSfM120k

rSfM120k dataset [4] consists of nearly 100 000 images. 91 642 images from 551 landmarks belong to the train set while 6 403 images from 162 landmarks, which are distinct from the train landmarks, belong to the validation set. ResNet101 model, trained with the Google Landmarks Dataset v1, with GeM Pooling and whitening layer is used to extract features of rSfM120k dataset. This model produces 2048-D feature vectors. We perform two types of extraction. First one is single scale in which the input image is fed in to the model at the original scale. Second one is multi scale in which the input image is fed in to the model at 3 scales which are $1024 \times (1, \sqrt{2}, 1/\sqrt{2})$, then these feature vectors are mean-aggregated. Resulting vectors are l_2 normalized. Python

[1] https://github.com/filipradenovic/cnnimageretrieval-pytorch

Figure 4.1 : Images from Google Landmark Dataset.

Image Retrieval Toolbox is used to extract all features and the transformer encoder models are trained with these feature representations. Our experiments are conducted with both single scale and multi scale vectors. We observe that training with single scale vectors gives slightly better results. Sample images from the dataset are given in Figure 4.2.

4.2 Test Datasets

Revisited Oxford ($\mathscr{R}$Oxford) and Revisited Paris ($\mathscr{R}$Paris) datasets are standard test benchmarks in image retrieval. In 2018, Radenovic et al. [40] extended the original Oxford and Paris datasets to revisited versions by adding new queries, 15 per dataset. Sample images from the dataset are given in Figure 4.3. As the standard evaluation protocols are saturated, they proposed new evaluation protocols: Easy, Medium and Hard. As the easy protocol is saturated, only medium and hard protocols are reported in recent works. Easy images can be considered as clear images without visual challenges such as occlusion, viewpoint variation or background clutter. Hard images can be

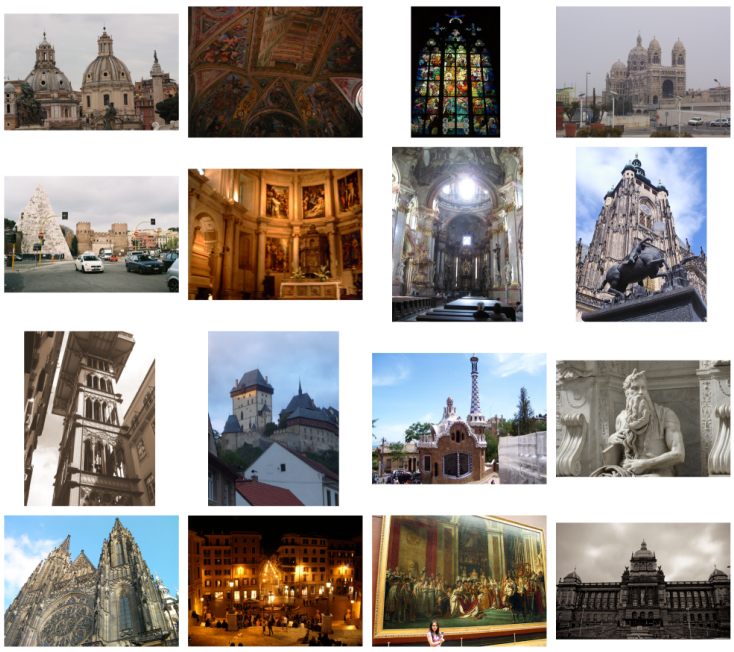

Figure 4.2 : Images from rSfM120k.

considered as difficult images containing visual challenges. Easy protocol considers easy images as positive, medium protocol considers easy and hard images as positive and hard protocol considers only hard images as positive. In order to conduct large scale comparisons on methods, $\mathscr{R}$1M, the distractor dataset, was also introduced in [40]. This dataset contains irrelevant images from the $\mathscr{R}$Oxford and $\mathscr{R}$Paris datasets. Our tests are conducted over these datasets using the suggested test protocols as described in [40]. Test results on Medium and Hard protocols are reported. Summary of test datasets are as follows:

- Revisited Oxford ($\mathscr{R}$Oxford ($\mathscr{R}$Oxf)): 4993 images with 70 query images.
- Revisited Paris ($\mathscr{R}$Paris ($\mathscr{R}$Par)): 6322 images with 70 query images.
- Distractor set (1 million distractors ($\mathscr{R}$1M)): 1 million distractors, which are irrelevant from ($\mathscr{R}$Oxf) and ($\mathscr{R}$Par) datasets, are added to the two datasets to evaluate the performance in the harder setup.

All test feature vectors are extracted at the multiscale setting, as described in the training dataset part to make a fair comparison with the existing methods. We use the

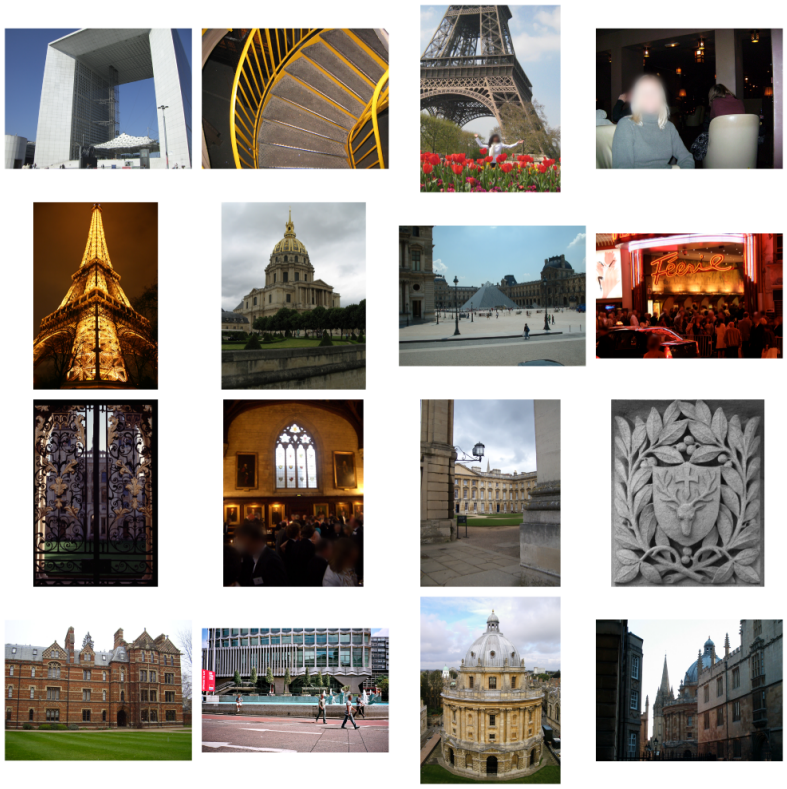

Figure 4.3 : Images from Oxford and Paris Datasets.

commonly used mean Average Precision (mAP) [41] to evaluate the performance.

4.3 Comparison to State-of-the-Art

All experimental results are presented in Table 4.1. Results of AQE [6], AQEwD [9], DQE [12], αQE [4] and LAttQE (depicted in gray) are given as reported in [1]. Our implementation of the LAttQE is given by LAttQE*. In both settings, the UGQE improves the baseline method [1] in terms of the reported performance measure. On the ($\mathscr{R}$Oxf) dataset, the UGQE performs better than other existing methods, and gives similar performance on the ($\mathscr{R}$Par). Furthermore, compared to the traditional methods and the LAttQE, the UGQE gives more balanced results as the mean mAP outperform the traditional methods. There are some mixed performance outputs where the αQE outperformed the transformer based techniques in some scenarios. A possible explanation is that after some point as the number of neighbors taken into account in the expansion increases, the performance in ($\mathscr{R}$Oxford) decreases while the performance in ($\mathscr{R}$Paris) increases. This indicates that there is a trade-off in the performance of the

two test datasets using traditional methods. However, even then it can be observed that the addition of the uncertainty guidance in the self-attention transformer framework helped UGQE surpass the performance results of the baseline in all settings. Although the results of LAttQE reported in [1] are not reproducable[2], the proposed UGQE outperforms the reproducable baseline method. Some sample visual results are given in Figure 4.4, where the UGQE tends to retrieve irrelevant images (red framed) in later ranks than the others when it is compared to the base model, LAttQE.

4.4 Database Side Augmentation

We employ a Database-side Augmentation (DBA) approach that is similar to that of [1], which entails dividing the weights by the temperature parameter T that regularizes the softmax inputs. Hence, the softmax function is applied to the regularized logits. After calculating the expanded query with the tempered weights, the resulting vector is l_2 normalized. For a given query, the weights and the expanded query are calculated as given in equations 4.1 and 4.2.

$$\mathbf{w} = \text{Softmax}\left(\text{normalizeddotproduct}\left(\tilde{\mathbf{q}}, \left[\tilde{\mathbf{d}}_0, \tilde{\mathbf{d}}_1, \ldots, \tilde{\mathbf{d}}_k\right]\right)/T\right) \tag{4.1}$$

$$\hat{\mathbf{q}} = \mathbf{w_q}\mathbf{q} + \sum_{\mathbf{n=1}}^{\mathbf{k}} \mathbf{w_{d_k}}\mathbf{d_k}, \hat{\mathbf{q}} = \frac{\hat{\mathbf{q}}}{||\hat{\mathbf{q}}||_2} \tag{4.2}$$

All database vectors are augmented beforehand *offline* as described above. This strategy gives the best results for database side augmentation. In Table 4.1, DBA+QE section shows the performance results when the described DBA procedure was applied before all QE models. The DBA-added UGQE method outperforms its baseline in all scenarios. Again, there are mixed results where for $\mathscr{R}$Paris ($\mathscr{R}$Par), the DBA-added αQE outperforms the others.

[2]We release our codes and trained models for the reproducable baseline (LAttQE) as well as the UGQE: https://github.com/ituvisionlab/ugqe

		$\mathcal{R}$Oxf		$\mathcal{R}$Oxf + $\mathcal{R}$1M		$\mathcal{R}$Par		$\mathcal{R}$Par + $\mathcal{R}$1M		
		M	H	M	H	M	H	M	H	Mean
No QE										
—		67.3	44.3	49.5	25.7	80.6	61.5	57.3	29.8	52.0
QE										
[6]	AQE	72.3	49.0	57.3	30.5	82.7	65.1	62.3	36.5	56.9
[9]	AQEwD	72.0	48.7	56.9	30.0	83.3	65.9	63.0	37.1	57.1
[12]	DQE	72.7	48.8	54.5	26.3	83.7	66.5	64.2	38.0	56.8
[4]	αQE	69.3	44.5	52.5	26.1	**86.9**	**71.7**	**66.5**	**41.6**	57.4
[1]	LAttQE†	73.4	49.6	58.3	31.0	86.3	70.6	67.3	42.4	59.8
[1]	LAttQE*	73.2	49.7	57.1	30.0	84.3	67.2	63.9	37.8	57.9
	UGQE	**73.3**	**50.1**	**58.3**	**31.0**	86.2	70.8	65.0	39.3	**59.2**
DBA + QE										
[6]	ADBA + AQE	71.9	53.6	55.3	32.8	83.9	68.0	65.0	39.6	58.8
[9]	ADBAwD + AQEwD	73.2	53.2	57.9	34.0	84.3	68.7	65.6	40.8	59.7
[12]	DDBA + DQE	72.0	50.7	56.9	32.9	83.2	66.7	65.4	39.1	58.4
[4]	αDBA + αQE	71.7	50.7	56.0	31.5	**87.5**	**73.5**	**70.6**	**48.5**	61.3
[1]	LAttQE + LAttDBA†	74.0	54.1	60.0	36.3	87.8	74.1	70.5	48.3	63.1
	LAttQE + LAttDBA*	73.8	54.4	57.8	33.0	85.8	70.6	67.2	42.7	60.7
	UGQE + UGQEDBA	**75.5**	**56.3**	**58.0**	**31.6**	87.3	73.3	67.7	43.7	**61.7**

Table 4.1 : Mean average precision (mAP) of $\mathcal{R}$Oxford ($\mathcal{R}$Oxf) and $\mathcal{R}$Paris ($\mathcal{R}$Par) with and without 1 million distractors ($\mathcal{R}$1M). Our uncertainty guided query expansion method outperforms both traditional and learning based expansion methods on most of the settings. (* with our implementation, gray lines with † as reported in [1], and all other scores also taken from [1]).

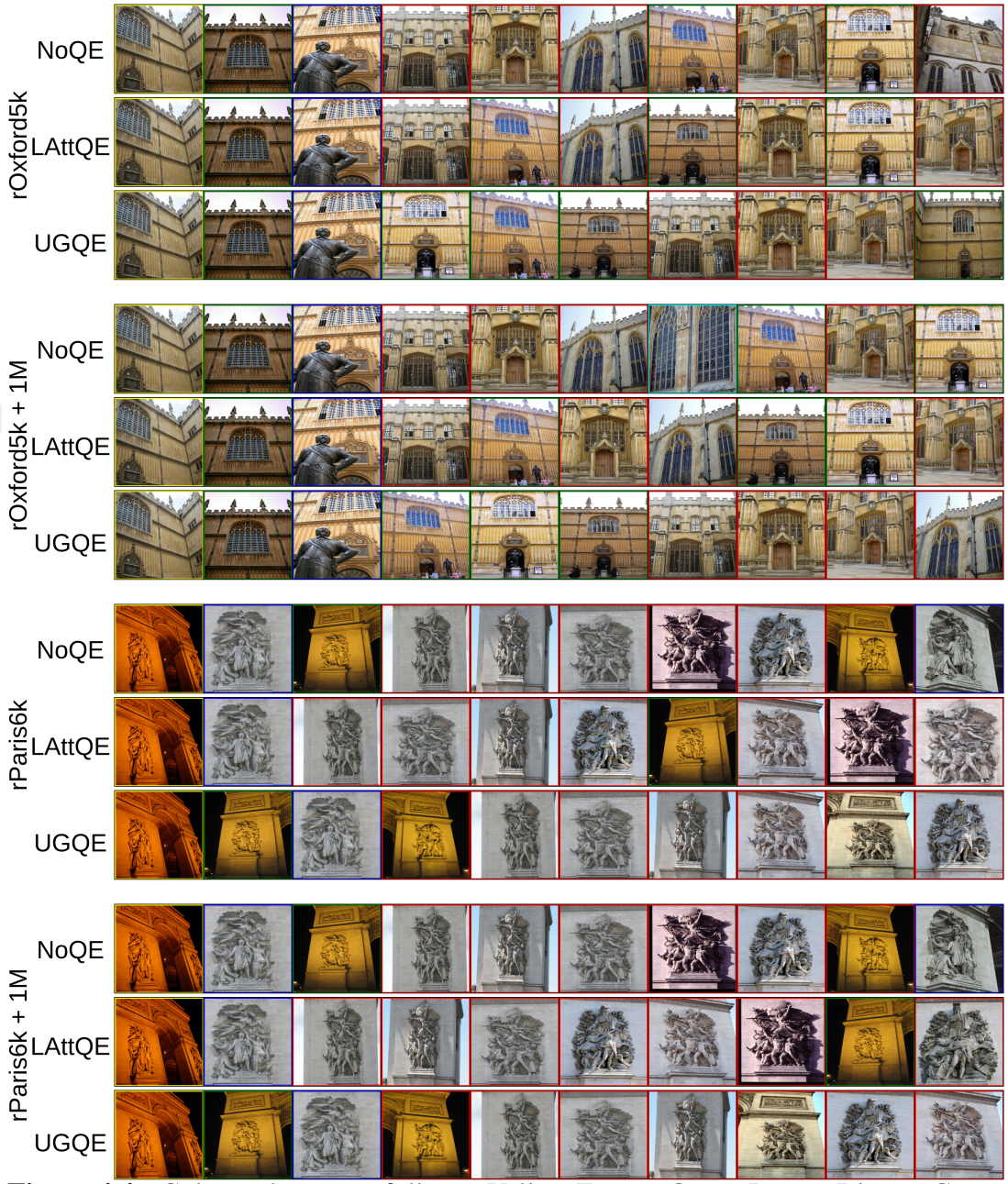

Figure 4.4 : Color codes are as follows: Yellow Frame: Query Image, Blue or Green Frame: Relevant Image, Red Frame: Irrelevant Image, Cyan Frame: Distractor Image. As it can be seen from examples, uncertainty information helps to remove some of the irrelevant retrievals.

4.5 Analysis

To interpret the results, two types of analysis are conducted. The first one is about the number of neighbors used for query expansion in each method. Result is given Figure 4.5. The second one is about the self-attention probabilities of the transformer-encoder module and the resulting weights for each neighbor used for query expansion. Results are given in Figures 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13, 4.14 and 4.15.

As it can be seen from Figure 4.5, some classical methods, AQE, AQEwD, and DQE, where the weights are the same or decreasingly monotonic are bound to fail when the neighbors for query expansion increase. When the α value is higher in αQE, this method gives a good performance but our method, UGQE, gives the best performance with the varying number of neighbors. When our method is compared to the base method, LAttQE, it also gives better performance than the base method. The base method struggles to obtain a satisfactory result when the number of neighbors is not quite high. The reason for this can be explained with the second self-attention probability analysis.

In the second analysis, self-attention probabilities in each layer for each model, UGQE and LAttQE, are plotted as a heat-map for the first queries in each dataset. Also, the final weights used in the expanded query are also plotted. This is done for varying numbers of neighbors, 8, 16, 32, 64, and 128. It can be summarized as follows:

- The baseline method, LAttQE, tends to give relatively high weights to the first neighbor in each setting, and the performance degradation depicted in Figure 4.5 may be a result of it.
- Self-attention probabilities in the first layers of each method, especially in UGTE part, are nearly uniformly distributed. From the first layer of the LABTE part of the UGQE method to the last layer, the farthest neighbors have increasing attention scores. But in the LAttQE method, the first and the last layers have similar attention scores and this could be the reason for the performance difference between the two methods.

- When the plots of all queries are examined, the same trend, in which LAttQE weights the first neighbor relatively high and the information from other neighbors are nearly neglected, is observed. It can be said that this trend is not occurring in our method.
- When the number of neighbors is increasing, the effect of the previously mentioned trend is decreased so that the performance of the base method increases. But in UGQE, performance is stable and it has its peak performance near 120 neighbors.
- The results for base method we implemented from scratch and the reported in [1] are not similar, this may be due to the hyperparameters we used different than the original ones.

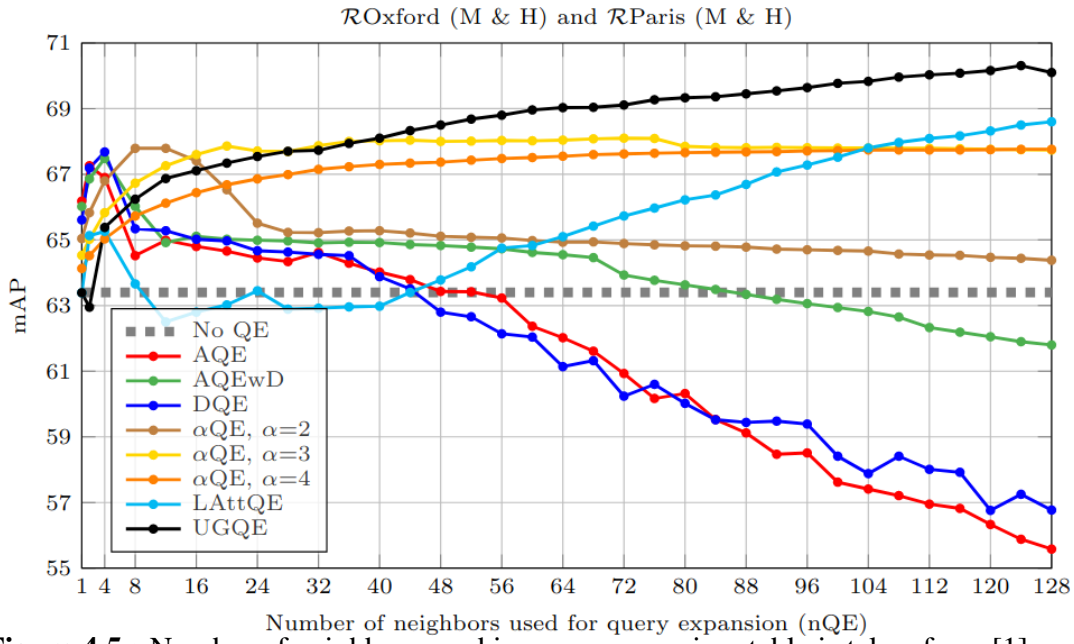

Figure 4.5 : Number of neighbors used in query expansion, table is taken from [1] and our results are added.

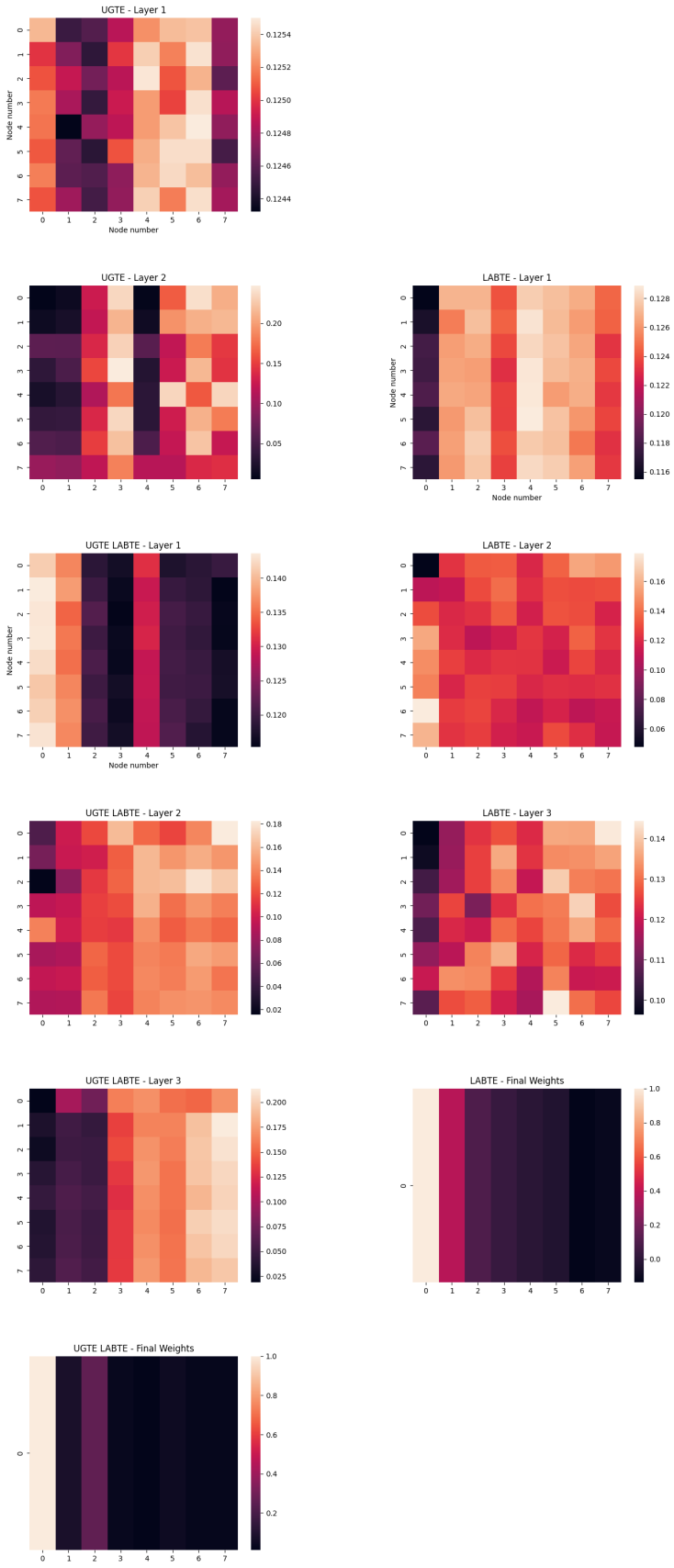

Figure 4.6 : Self-Attention Probability Analysis with 8 vectors in rOxford5k Dataset.

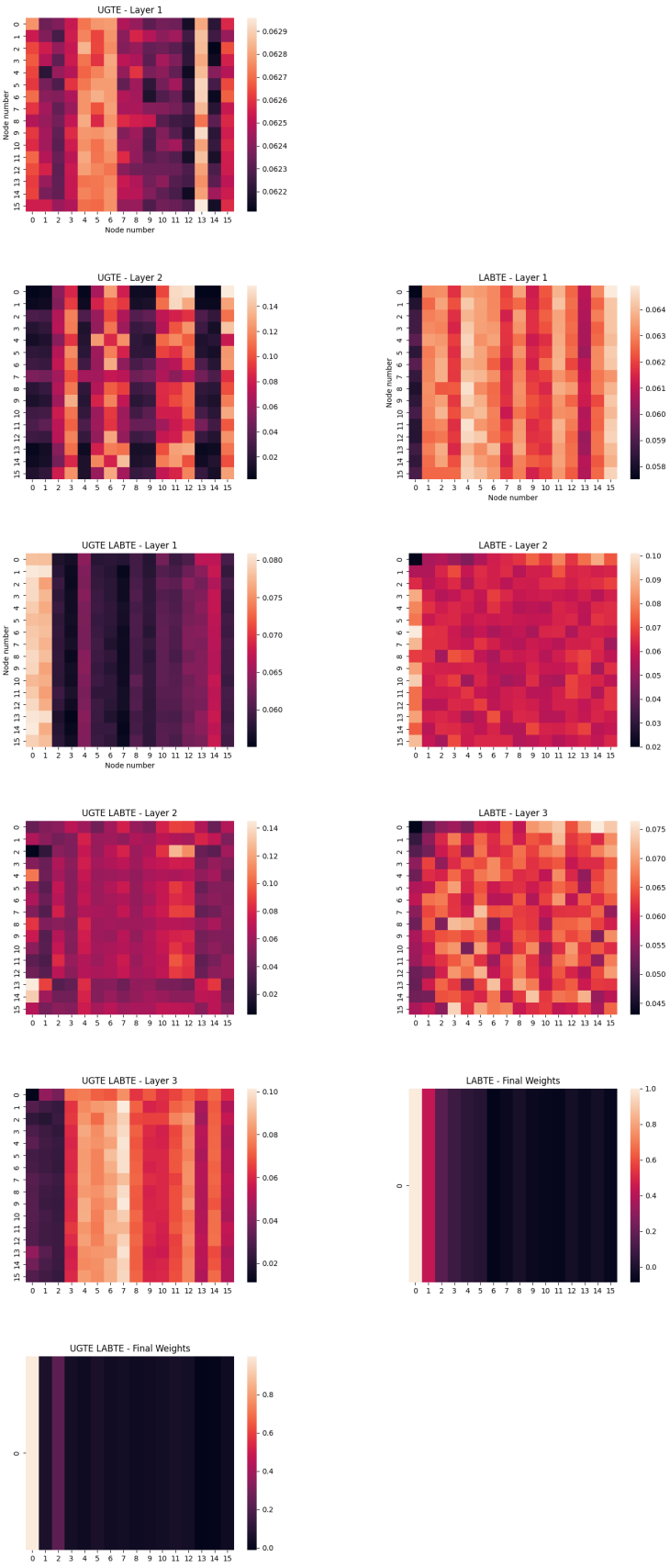

Figure 4.7 : Self-Attention Probability Analysis with 16 vectors in rOxford5k Dataset.

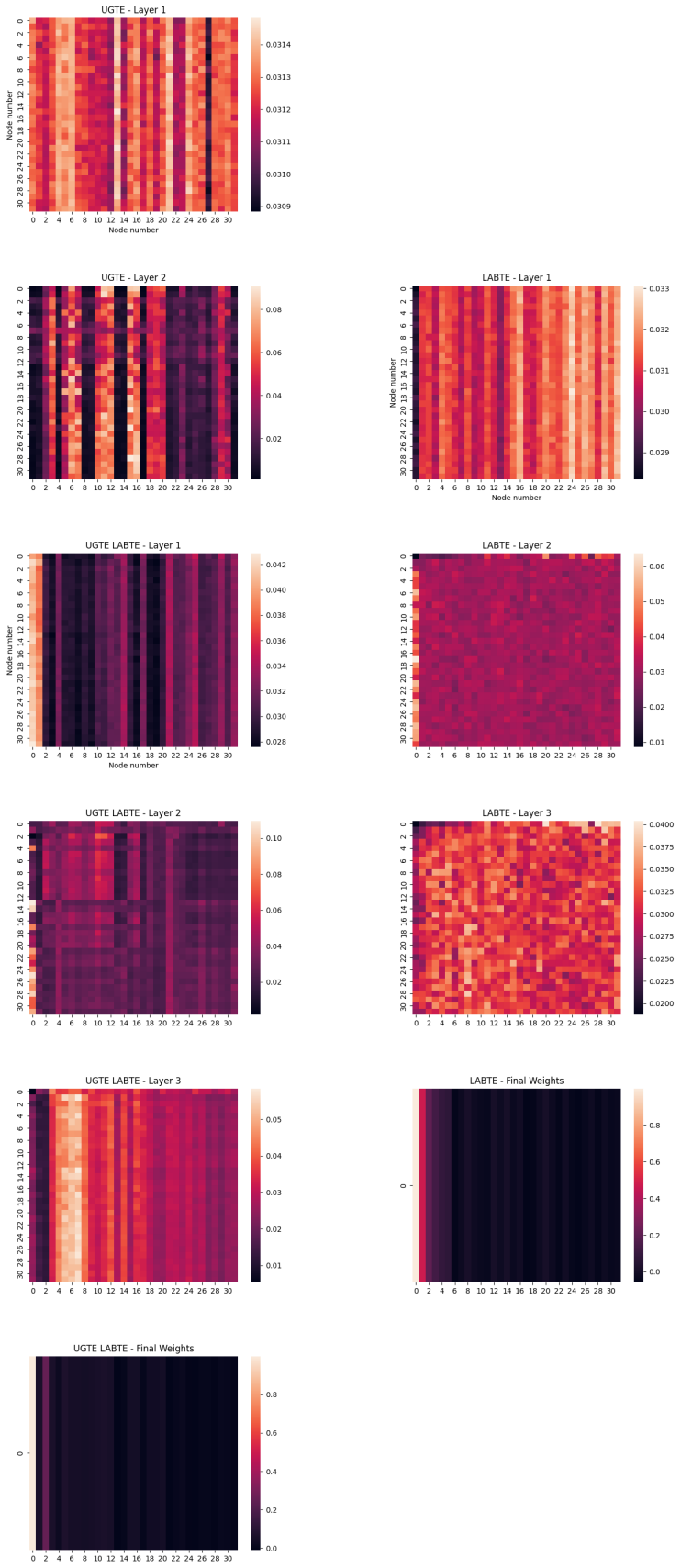

Figure 4.8 : Self-Attention Probability Analysis with 32 vectors in rOxford5k Dataset.

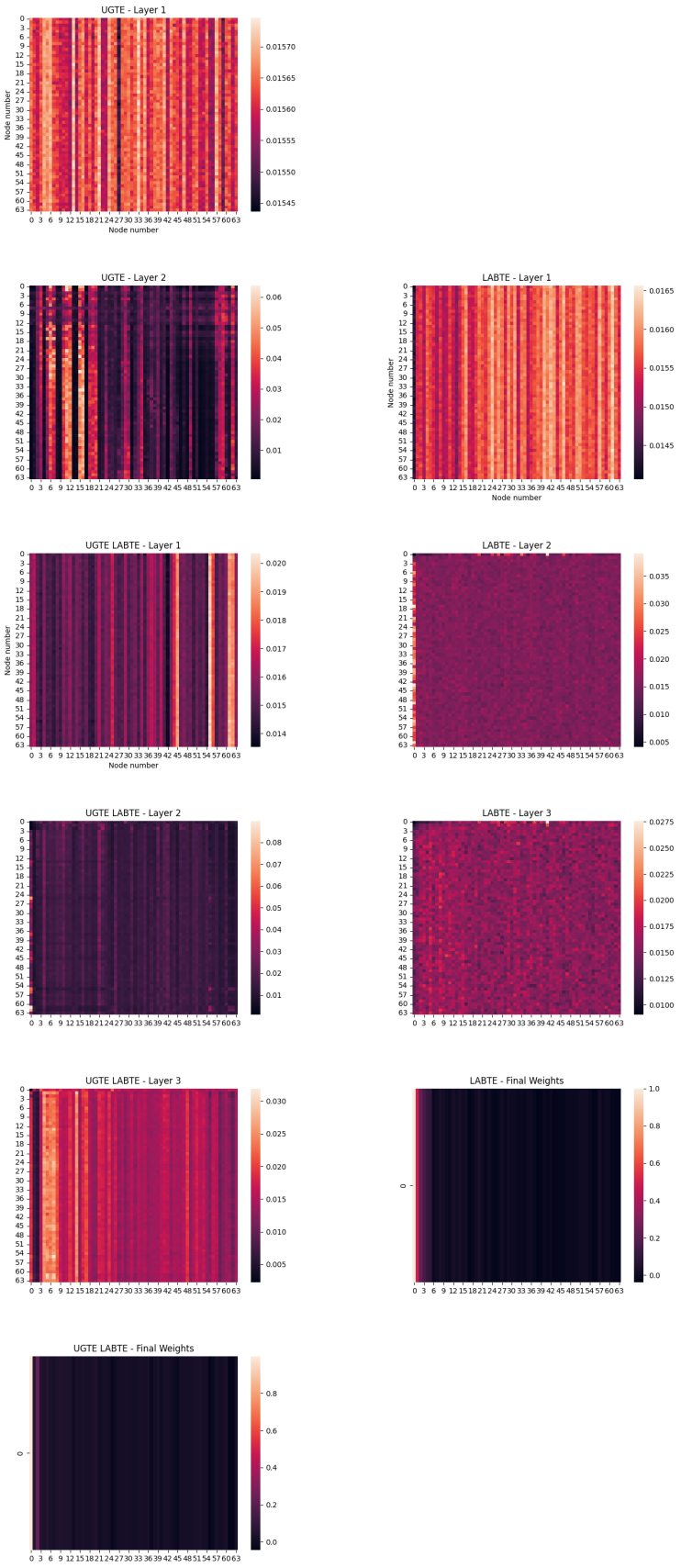

Figure 4.9 : Self-Attention Probability Analysis with 64 vectors in rOxford5k Dataset.

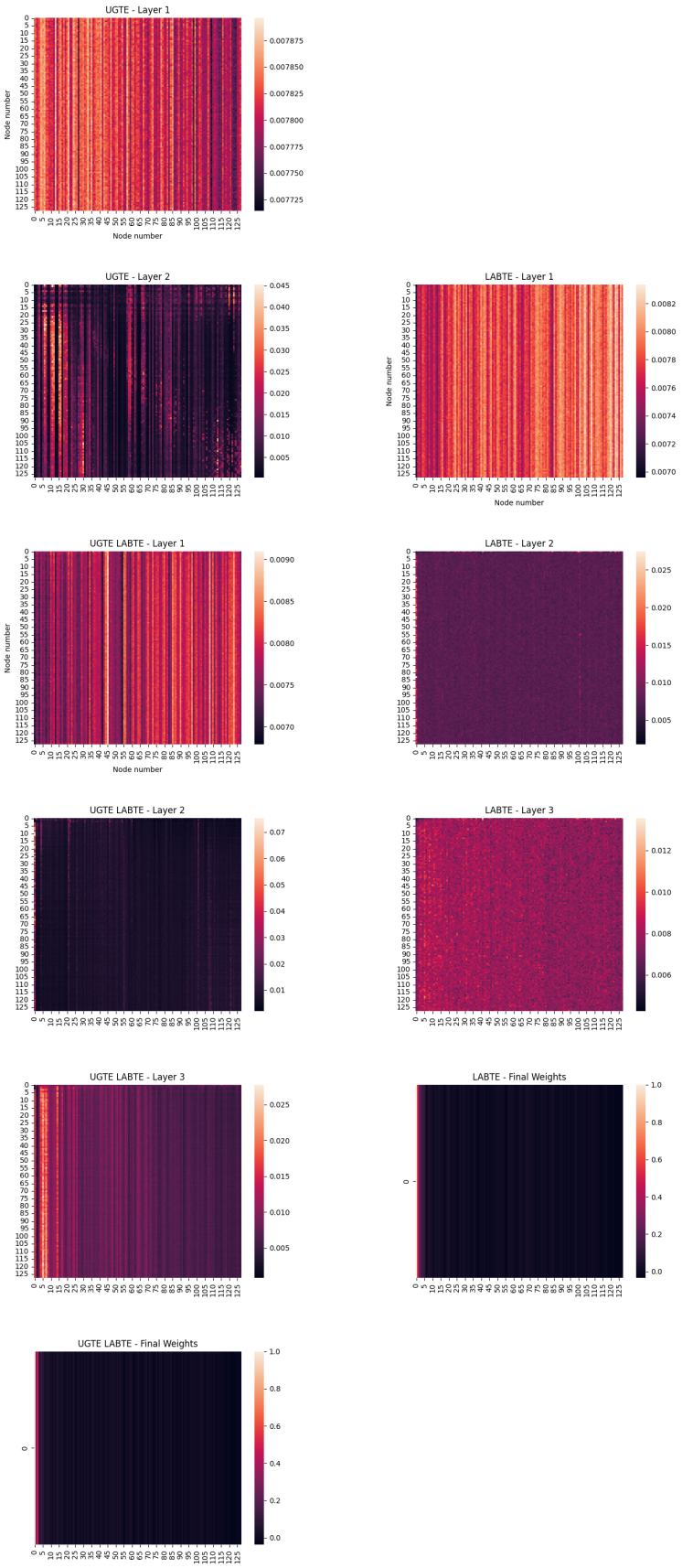

Figure 4.10 : Self-Attention Probability Analysis with 128 vectors in rOxford5k Dataset.

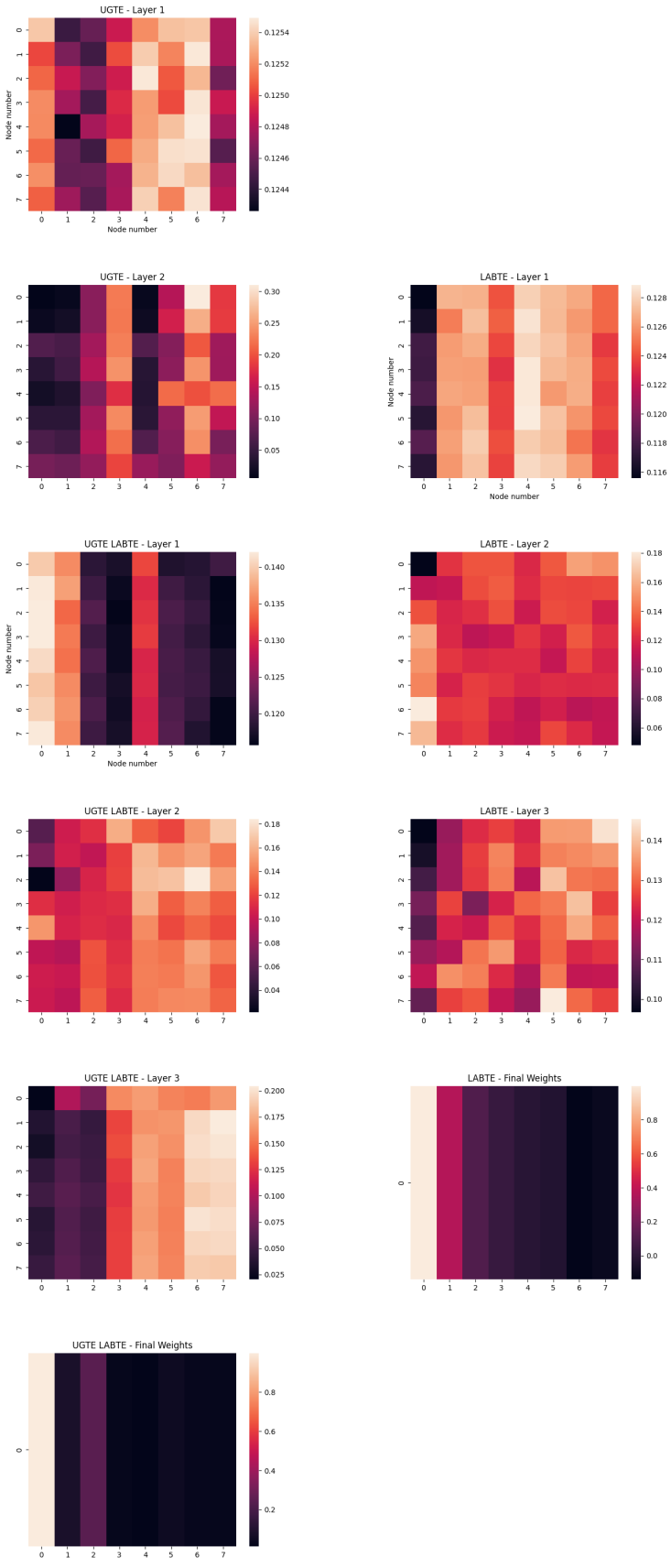

Figure 4.11 : Self-Attention Probability Analysis with 8 vectors in rParis6k Dataset.

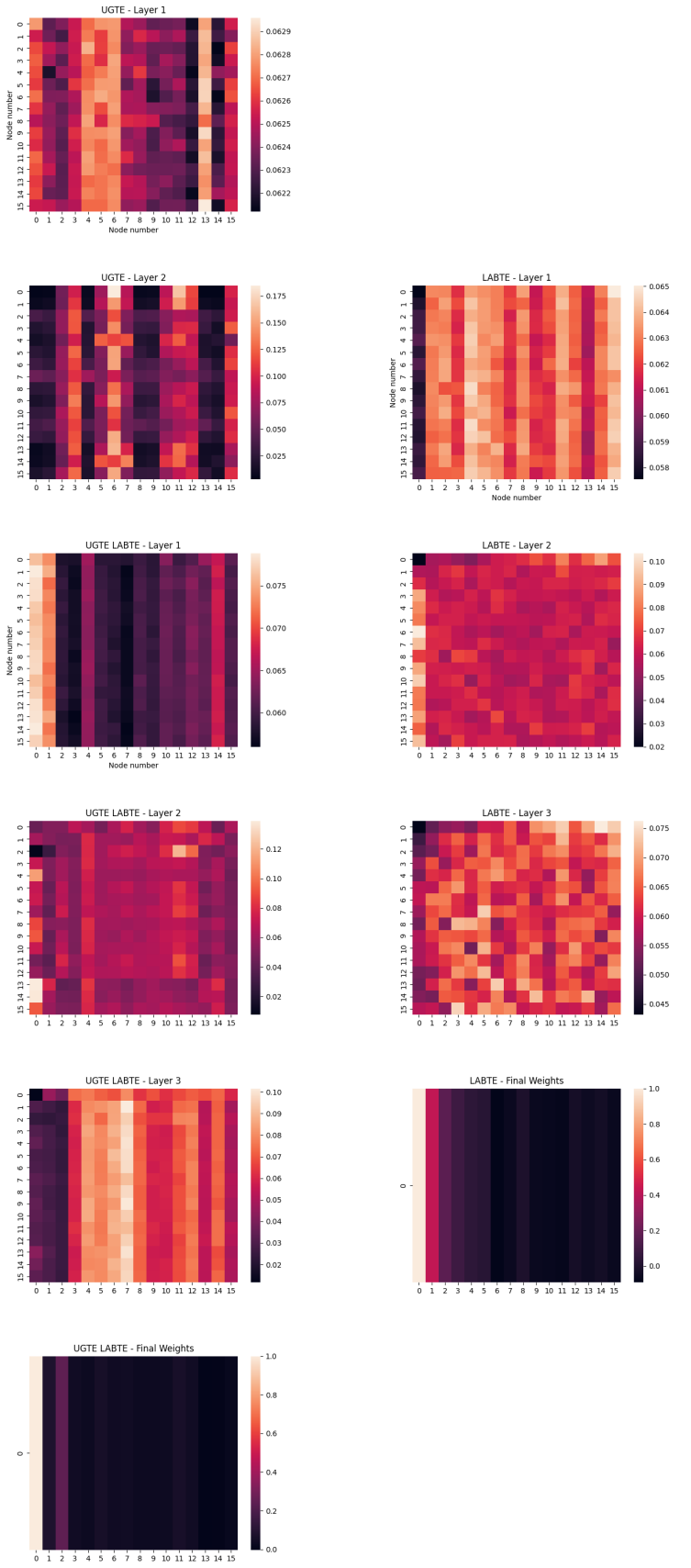

Figure 4.12 : Self-Attention Probability Analysis with 16 vectors in rParis6k Dataset.

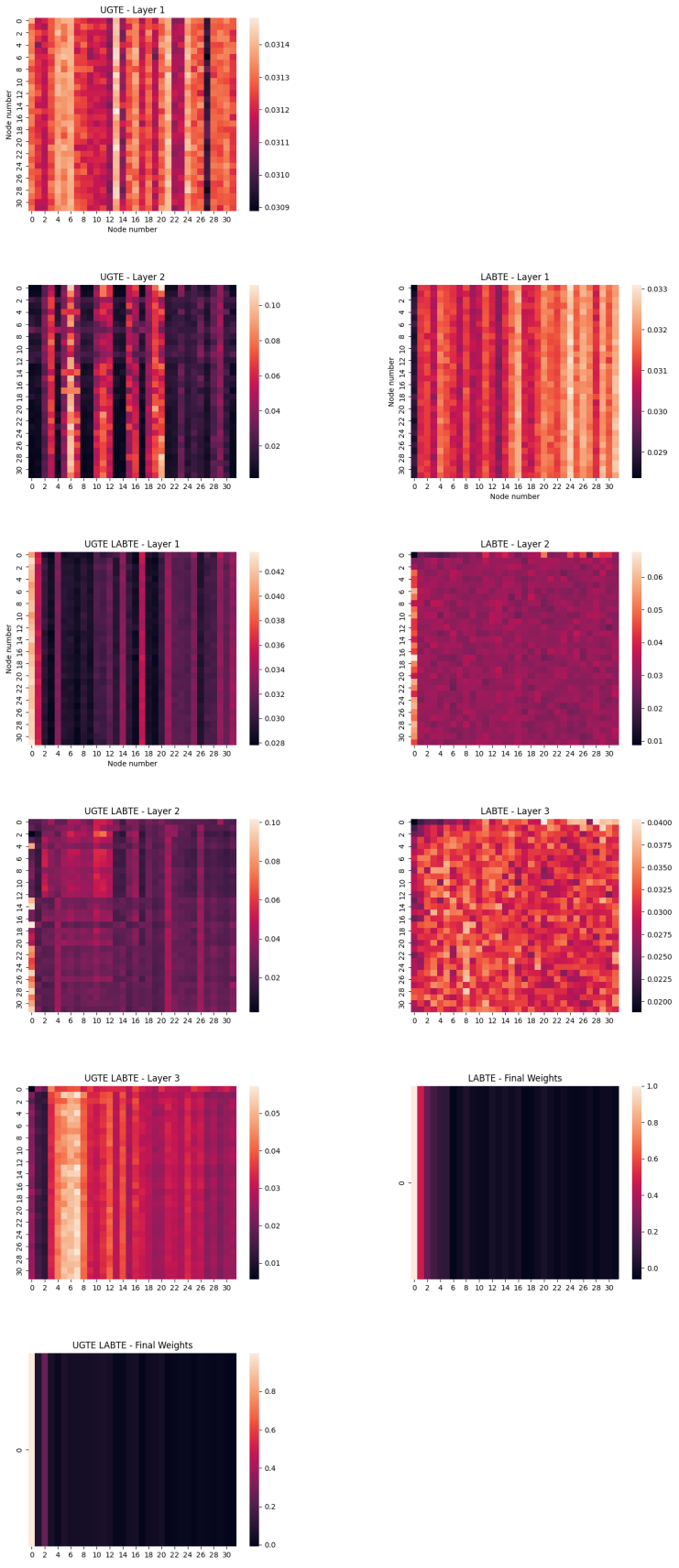

Figure 4.13 : Self-Attention Probability Analysis with 32 vectors in rParis6k Dataset.

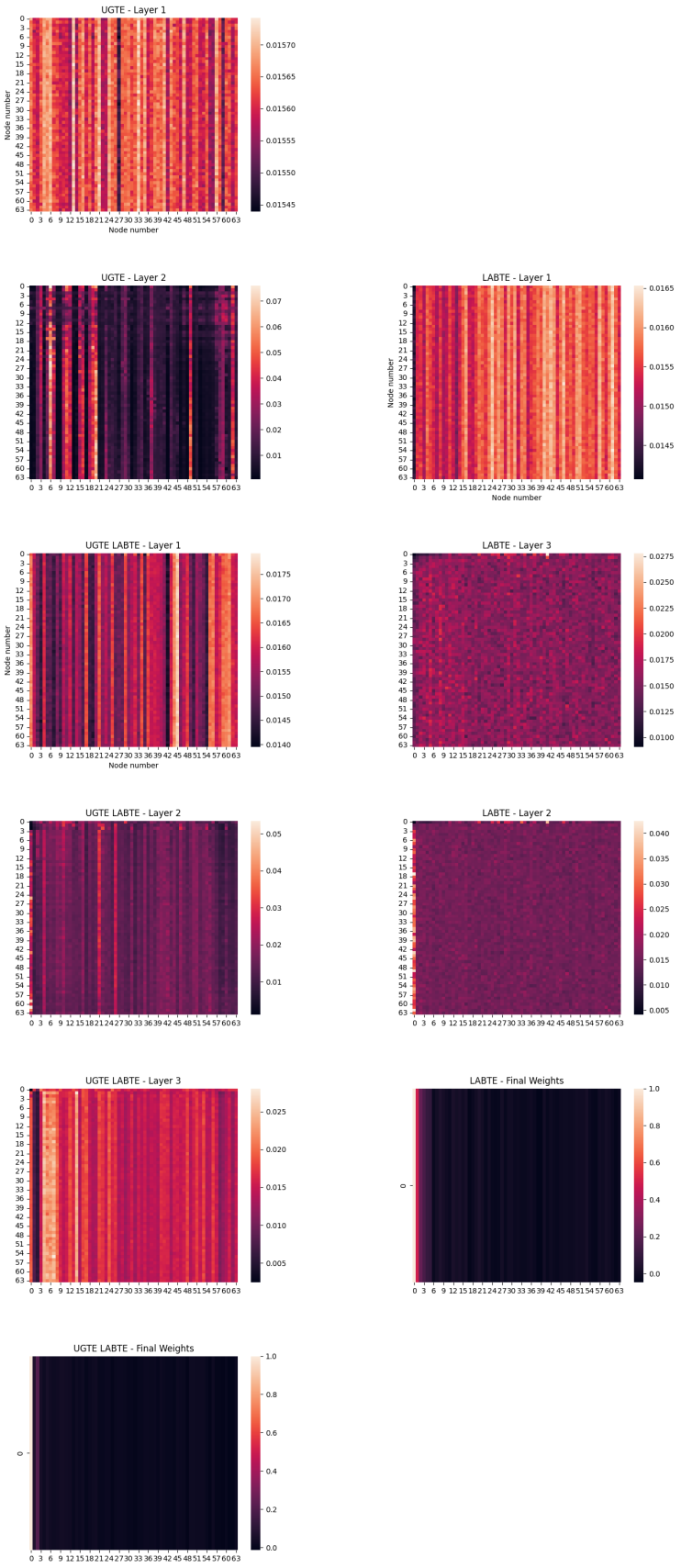

Figure 4.14 : Self-Attention Probability Analysis with 64 vectors in rParis6k Dataset.

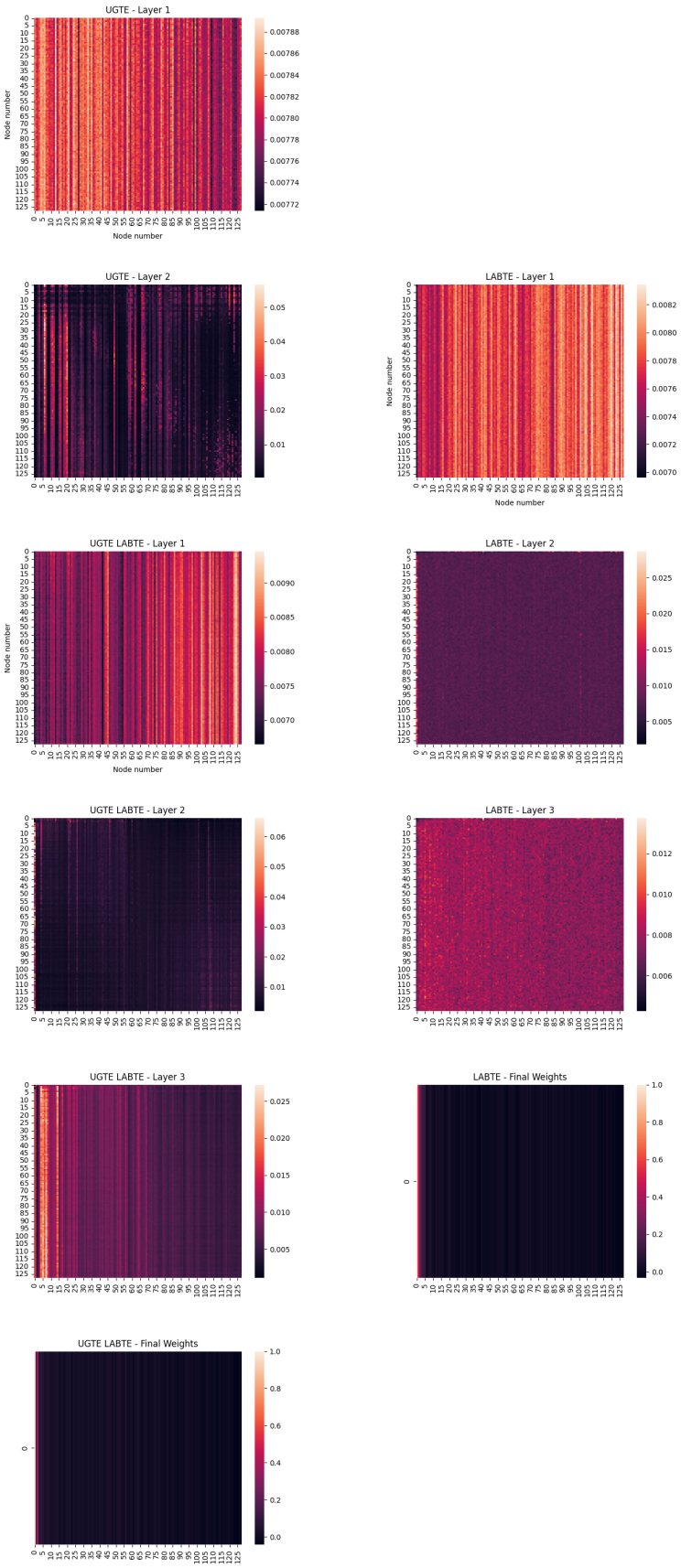

Figure 4.15 : Self-Attention Probability Analysis with 128 vectors in rParis6k Dataset.

5. CONCLUSION

In this thesis, we proposed an uncertainty guided self-attention mechanism to learn query expansion in an end-to-end fashion. We built a novel feature generation method that is compatible with the traditional methods such as the QE, AQEwD and the learning based LAttQE method. Our work provides evidence to our hypothesis that the integration of uncertainty information on the neighborhood relationships in image retrieval methods can lead to the creation of more robust retrieval systems.

In UGQE, our main contribution is to build a new feature generation mechanism with transformer-encoder blocks using EDL to help LAttQE to perform better while assigning the weights and experiments support that. Our results indicate that the newly generated features are enriching the original features by estimating the uncertainty in neighborhood relationships.

Our auxiliary findings can be summarized as follows: (i) Extraction of the training feature vectors, can be done in two ways: single scale, where only the original image size is used; or multi scale, where the original image size is resized with different scales then the feature vectors of each scale are aggregated. In our experiments, test feature vectors are multi scale to make a fair comparison with existing methods, however, we observe that using single scale vectors in training gives better performance. Our experiments imply that in the query expansion method, the way of training and test feature vectors' extraction is independent of each other as we are assigning weights to given vectors in our framework to construct expanded query, at the end. (ii) In the original LAttQE paper, in the training phase of transformer-encoder blocks neighborhood dropping method is employed to use different number of neighbors in each batch, however, in our experiments we observed no improvements in that aspect, and obtained the best performing result without dropping the neighbors. (iii) We conjecture that the type of the positional encoding, fixed or learnable, is a crucial component of the system. For the LAttQE baseline, as there was no code available (and no response to our inquiries

to the related github repository), no information about the initialization of the positional encodings was made available. Therefore, after trying different initializations, we observed that uniform initialization, between 0 and 1, performed the best, which we picked.

Finally, as a future work, fine-tuning the feature extractor network, ResNet101, using our setup in an end-to-end fashion can be done. In this way, we can utterly change the image representations by integrating uncertainty information. Also, new weighting schemes for the feature activations (outputs of CNN layers) can also be proposed. Weights can be assigned to every element of $H \times W$ feature activation map and then reduced to a scalar. Integrating the intermediate layer outputs of CNN to the expanded query can be considered as another way of improvement. We believe that this thesis particularly stimulates further research questions on how to incorporate uncertainty information into image retrieval models. Hence, investigation on other uncertainty measures for reliability of related neighbors to the query could be pursued.

REFERENCES

[1] **Gordo, A.**, **Radenovic, F. and Berg, T.** (2020). Attention-based query expansion learning, *European Conference on Computer Vision*, Springer, pp.172–188.

[2] **Vaswani, A.**, **Shazeer, N.**, **Parmar, N.**, **Uszkoreit, J.**, **Jones, L.**, **Gomez, A.N.**, **Łukasz Kaiser and Polosukhin, I.** (2017). Attention is all you need, *NeurIPS*.

[3] **Noh, H.**, **Araujo, A.**, **Sim, J.**, **Weyand, T. and Han, B.** (2017). Large-Scale Image Retrieval With Attentive Deep Local Features, *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

[4] **Radenovic, F.**, **Tolias, G. and Chum, O.** (2018). Fine-tuning CNN Image Retrieval with No Human Annotation, *TPAMI*.

[5] **He, K.**, **Zhang, X.**, **Ren, S. and Sun, J.** (2016). *CVPR*.

[6] **Chum, O.**, **Philbin, J.**, **Sivic, J.**, **Isard, M. and Zisserman, A.** (2007). Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval, *2007 IEEE 11th ICCV*.

[7] **Tolias, G. and Jégou, H.** (2014). Visual query expansion with or without geometry: refining local descriptors by feature aggregation, *PR*.

[8] **Babenko, A.**, **Slesarev, A.**, **Chigorin, A. and Lempitsky, V.S.** (2014). Neural Codes for Image Retrieval, *ECCV 2014*, volume8689 of *Lecture Notes in Computer Science*, Springer, pp.584–599.

[9] **Gordo, A.**, **Almazan, J.**, **Revaud, J. and Larlus, D.** (2017). End-to-end Learning of Deep Visual Representations for Image Retrieval, *IJCV*.

[10] **Turcot, T. and Lowe, D.G.** (2009). Better matching with fewer features: The selection of useful features in large database recognition problems, *ICCV Workshop*.

[11] **Datar, M. and Indyk, P.** (2004). Locality-sensitive hashing scheme based on p-stable distributions, *In SCG '04: Proceedings of the twentieth annual symposium on Computational geometry*, ACM Press, pp.253–262.

[12] **Arandjelovic, R. and Zisserman, A.** (2012). Three things everyone should know to improve object retrieval, *CVPR*.

[13] **Guo, C.**, **Pleiss, G.**, **Sun, Y. and Weinberger, K.Q.** (2017). On calibration of modern neural networks, *International Conference on Machine Learning*, pp.1321–1330.

[14] **Gal, Y. and Ghahramani, Z.** (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, *ICML*, PMLR, pp.1050–1059.

[15] **Lakshminarayanan, B.**, **Pritzel, A. and Blundell, C.** (2017). Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, *Advances in Neural Information Processing Systems*, volume 30.

[16] **Malinin, A. and Gales, M.** (2018). Predictive uncertainty estimation via prior networks, *arXiv preprint arXiv:1802.10501*.

[17] **Sensoy, M.**, **Kaplan, L. and Kandemir, M.** (2018). Evidential Deep Learning to Quantify Classification Uncertainty, *Advances in Neural Information Processing Systems*.

[18] **Krizhevsky, A.**, **Sutskever, I. and Hinton, G.E.** (2012). ImageNet Classification with Deep Convolutional Neural Networks, *Advances in Neural Information Processing Systems*, Curran Associates, Inc.

[19] **Szegedy, C.**, **Liu, W.**, **Jia, Y.**, **Sermanet, P.**, **Reed, S.**, **Anguelov, D.**, **Erhan, D.**, **Vanhoucke, V. and Rabinovich, A.** (2015). Going deeper with convolutions, *CVPR*.

[20] **Simonyan, K. and Zisserman, A.** (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition, *International Conference on Learning Representations*.

[21] **Tan, M. and Le, Q.** (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, *ICML*.

[22] **Lowe, D.** (1999). Object recognition from local scale-invariant features, *Proceedings of the Seventh IEEE ICCV*.

[23] **Perronnin, F.**, **Sánchez, J. and Mensink, T.** (2010). Improving the Fisher Kernel for Large-Scale Image Classification, ***K. Daniilidis, P. Maragos and N. Paragios**, editors, ECCV 2010*, volume6314 of *Lecture Notes in Computer Science*, Springer, pp.143–156.

[24] **Jégou, H.**, **Douze, M.**, **Schmid, C. and Pérez, P.** (2010). Aggregating local descriptors into a compact image representation, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.3304–3311.

[25] **Arandjelovic, R.**, **Gronat, P.**, **Torii, A.**, **Pajdla, T. and Sivic, J.** (2016). NetVLAD: CNN Architecture for Weakly Supervised Place Recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[26] **Buckley, C.** (1994). Automatic Query Expansion Using SMART : TREC 3, *In Proceedings of The third Text REtrieval Conference (TREC-3*, pp.69–80.

[27] **Salton, G. and Buckley, C.** (1990). Improving retrieval performance by relevance feedback, *Journal of the American Society for Information Science*, *41*(4), 288–297.

[28] **Tolias, G.**, **Sicre, R. and Jégou, H.** (2016). Particular object retrieval with integral max-pooling of CNN activations, ***Y. Bengio and Y. LeCun****, editors, 4th International Conference on Learning Representations, ICLR*.

[29] **Brown, T.**, **Mann, B.**, **Ryder, N.**, **Subbiah, M.**, **Kaplan, J.D.**, **Dhariwal, P.**, **Neelakantan, A.**, **Shyam, P.**, **Sastry, G.**, **Askell, A.**, **Agarwal, S.**, **Herbert-Voss, A.**, **Krueger, G.**, **Henighan, T.**, **Child, R.**, **Ramesh, A.**, **Ziegler, D.**, **Wu, J.**, **Winter, C.**, **Hesse, C.**, **Chen, M.**, **Sigler, E.**, **Litwin, M.**, **Gray, S.**, **Chess, B.**, **Clark, J.**, **Berner, C.**, **McCandlish, S.**, **Radford, A.**, **Sutskever, I. and Amodei, D.** (2020). Language Models are Few-Shot Learners, *Advances in Neural Information Processing Systems*, volume 33, pp.1877–1901.

[30] **Devlin, J.**, **Chang, M.W.**, **Lee, K. and Toutanova, K.** (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *NAACL*.

[31] **Liu, X.**, **Duh, K.**, **Liu, L. and Gao, J.** (2020). Very deep transformers for neural machine translation, *arXiv preprint arXiv:2008.07772*.

[32] **Dosovitskiy, A.**, **Beyer, L.**, **Kolesnikov, A.**, **Weissenborn, D.**, **Zhai, X.**, **Unterthiner, T.**, **Dehghani, M.**, **Minderer, M.**, **Heigold, G.**, **Gelly, S.**, **Uszkoreit, J. and Houlsby, N.** (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, *9th International Conference on Learning Representations, ICLR,*.

[33] **El-Nouby, A.**, **Neverova, N.**, **Laptev, I. and Jégou, H.** (2021). Training Vision Transformers for Image Retrieval, *CoRR*, *abs/2102.05644*.

[34] **Veličković, P.**, **Cucurull, G.**, **Casanova, A.**, **Romero, A.**, **Liò, P. and Bengio, Y.** (2018). Graph Attention Networks, *International Conference on Learning Representations*.

[35] **Seidenschwarz, J.**, **Elezi, I. and Leal-Taixé, L.** (2021). Learning Intra-Batch Connections for Deep Metric Learning, *38th International Conference on Machine Learning, (ICML)*.

[36] **Amini, A.**, **Schwarting, W.**, **Soleimany, A. and Rus, D.** (2020). Deep evidential regression, *Advances in Neural Information Processing Systems*, *33*.

[37] **Kingma, D.P. and Ba, J.** (2014). Adam: A method for stochastic optimization, *arXiv:1412.6980*.

[38] **Noh, H.**, **Araujo, A.**, **Sim, J.**, **Weyand, T. and Han, B.** (2017). Large-scale image retrieval with attentive deep local features, *ICCV*.

[39] **Weyand, T.**, **Araujo, A.**, **Cao, B. and Sim, J.** (2020). Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[40] **Radenović, F.**, **Iscen, A.**, **Tolias, G.**, **Avrithis, Y. and Chum, O.** (2018). Revisiting oxford and paris: Large-scale image retrieval benchmarking, *CVPR*.

[41] **Philbin, J.**, **Chum, O.**, **Isard, M.**, **Sivic, J. and Zisserman, A.** (2007). Object retrieval with large vocabularies and fast spatial matching, *CVPR*.

CURRICULUM VITAE

Name SURNAME: Fırat ÖNCEL

EDUCATION:

- **B.Sc.:** 2019, Istanbul Technical University, Faculty of Computer and Informatics, Department of Computer Engineering
- **M.Sc.:** 2022, Istanbul Technical University, Graduate School, Department of Computer Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2019-ongoing Research Assistant at Department of Computer Engineering, Istanbul Technical University

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Oncel, F.**, Aygün, M., Baykal, G., Unal, G. (2022). UGQE: Uncertainty Guided Query Expansion. In: El Yacoubi, M., Granger, E., Yuen, P.C., Pal, U., Vincent, N. (eds) Pattern Recognition and Artificial Intelligence. ICPRAI 2022. Lecture Notes in Computer Science, vol 13363. Springer, Cham. https://doi.org/10.1007/978-3-031-09037-0_10

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- Bozkurt M., **Oncel F.**, Gürpınar C., Kose H., Unal G., 2022. Facial Expressions Detection of Children with Hearing Impairment, 2022 30th Signal Processing and Communications Applications Conference (SIU), 1-4.