TÉCNICO LISBOA

Automatic Identification of Regions of Interest in Dermoscopy Images Using Vision Transformers and Multiple Instance Learning

Diogo José Pereira Araújo

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Carlos Jorge Andrade Mariz Santiago
Prof. Ana Catarina Fidalgo Barata

Examination Committee

Chairperson: Prof. Pedro Filipe Zeferino Aidos Tomás
Supervisor: Prof. Carlos Jorge Andrade Mariz Santiago
Member of the Committee: Prof. Chrysoula Zerva

November 2023

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

This work was created using LaTeX typesetting language
in the Overleaf environment (www.overleaf.com).

Acknowledgments

I would like to express my deepest gratitude to Prof. Catarina Barata and Prof. Carlos Santiago. Their constant guidance and support have been instrumental in shaping this thesis. I am truly fortunate to have had supervisors who not only provided valuable insights, but also encouraged engaging discussions that significantly influenced the development of this work. Under their supervision, I have grown considerably both personally and professionally, and have learned so much about the fields of deep learning and computer vision.

I would like to express my sincere gratitude to my friends, Alceu, André, Miguel, Ricardo, Rita, and Tiago, who have shared this academic journey with me. Their constant support and willingness to help in any situation made this experience much more enjoyable than it otherwise would have been without them.

Finally, I would like to express my deepest gratitude to my family and friends for their continued understanding and constant support throughout all these years.

Abstract

Skin cancer is a growing public health concern. Early detection of the lesion plays a critical role in ensuring successful treatment of the cancer. Dermatologists traditionally use criteria like the 7-point checklist, which focuses on specific dermoscopic characteristics without considering their spatial distribution in the lesion. Multiple Instance Learning (MIL) is a weakly supervised learning technique that serves as an approximation to this criterion in the field of deep learning. In contrast to these methods, Vision Transformers (ViTs) have recently shown remarkable promise, while at the same time using spatially aware information from all the patches in the image. This contrast motivates us to address two questions in dermoscopy image analysis: (1) the understanding of whether all patches are relevant for skin cancer diagnosis, and (2) the influence of the spatial arrangement of the patches on diagnostic accuracy. To address these questions, we introduce a two-branch framework that combines a ViT-based architecture with a MIL model. We tackle both binary classification (melanoma vs. nevus) and multi-class classification (with eight skin disease types). Our work presents a novel two-stage MIL formulation oriented towards binary classification, and we extend it to a three-stage approach for multi-class classification. Our results consistently demonstrate the competitive performance of these formulations in both binary and multi-class contexts. Our findings reveal that only certain patches are critical for correct classification, and that adding spatial information slightly improves classification accuracy.

Keywords

Skin Cancer; Vision Transformers; Multiple Instance Learning.

Resumo

O cancro da pele é uma preocupação crescente em termos de saúde pública. A deteção atempada da lesão desempenha um papel fundamental para garantir o sucesso do tratamento do cancro. Os dermatologistas utilizam critérios como a lista de verificação de 7 pontos, que se baseia em características dermatoscópicas específicas sem considerar a sua distribuição espacial na lesão. Multiple Instance Learning (MIL) é uma técnica de aprendizagem com supervisão fraca que serve de aproximação a este critério no domínio da aprendizagem profunda. Em contraste com estes métodos, os Vision Transformers (ViTs) mostraram recentemente uma potencialidade notável, embora utilizem, ao mesmo tempo, as características espaciais de todas as partes da imagem. Este contraste motiva-nos a abordar duas questões no centro da análise de imagens dermatoscópicas: (1) a compreensão de se todas as zonas são relevantes para a classificação da imagem, e (2) a influência da localização espacial dessas zonas na exatidão da classificação. Para abordar estas questões, introduzimos uma estrutura de dois componentes que combina uma arquitetura baseada no ViT com um modelo de MIL. Abordamos tanto a classificação binária (melanoma vs. nevo) como a classificação multi-classe (com oito tipos de cancros da pele). O nosso trabalho apresenta uma nova formulação MIL de duas etapas orientada para a classificação binária, e estendemo-la a uma abordagem de três etapas para a classificação multi-classe. Os nossos resultados demonstram consistentemente o desempenho competitivo destas formulações. As nossas conclusões revelam que apenas determinadas áreas são essenciais para uma classificação acertada e que a informação espacial melhora a exatidão na classificação de cancro da pele.

Palavras Chave

Cancro da Pele; Vision Transformers; Multiple Instance Learning.

Contents

List of Figures

List of Tables

Acronyms

Acc	Accuracy
AK	Actinic keratosis
BA	Balanced Accuracy
BCC	Basal cell carcinoma
BKL	Benign keratosis
CLS	Classification
CV	Computer Vision
CNN	Convolutional Neural Network
CE	Cross Entropy
DEiT	Data-efficient Image Transformer
DL	Deep Learning
DN-169	DenseNet-169
DF	Dermatofibroma
EN-B3	EfficientNet-B3
EViT	Expediting Vision Transformer
FN	False Negatives
FP	False Positives
FNN	Feedfoward Neural Network
FC Layer	Fully Connected Layer
GRU	Gated Recurrent Unit
Kr	Keep rate
LN	Layer Normalization

LSTM	Long Short-Term Memory
MVT	Medical Vision Transformer
NV	Melanocytic nevus
MEL	Melanoma
MSA	Multi-head Self-Attention
MLP	Multi-layer Perceptron
MIL	Multiple Instance Learning
NLP	Natural Language Processing
NN	Neural Network
RNN	Recurrent Neural Network
RGB	Red,Green and Blue
ROI	Region of Interest
RN-18	ResNet-18
RN-50	ResNet-50
SA	Self-Attention
Seq2Seq	Sequence to Sequence
SCC	Squamous cell carcinoma
SVM	Support Vector Machine
TN	True Negative
TP	True Positives
VASC	Vascular lesion
ViT	Vision Transformer
WSI	Whole Slide Image

1

Introduction

Contents

1.1 Motivation

Skin cancer is a growing global health concern. It is the most common type of cancer, with nearly 3.3 million cases diagnosed annually in the U.S. alone [6]. Early detection of the cancer is critical to improving the chances of successful treatment. The five-year survival rate for melanoma, which is the deadliest type of skin cancer, is 99% when detected early [7]. However, survival rates are usually low after the disease has spread beyond the skin [8], which emphasizes the importance of diagnosing the disease at an early stage.

Currently, the most common method for early detection of skin cancer is through visual inspection by a specialist. In clinical practice, dermatologists often rely on the identification of specific dermoscopic criteria to diagnose melanoma lesions. For example, the 7-point checklist criterion [9] relies on the identification of features such as atypical pigment networks, irregular streaks, irregular pigmentation, and more, to diagnose a lesion as melanoma. To provide a visual representation of some of the dermoscopic criteria used by physicians when analyzing dermoscopic images, Figure 1.1 is included. Given the challenges in distinguishing malignant from non-malignant lesions, drawing conclusions from these dermoscopy criteria can be a complex and arduous task.

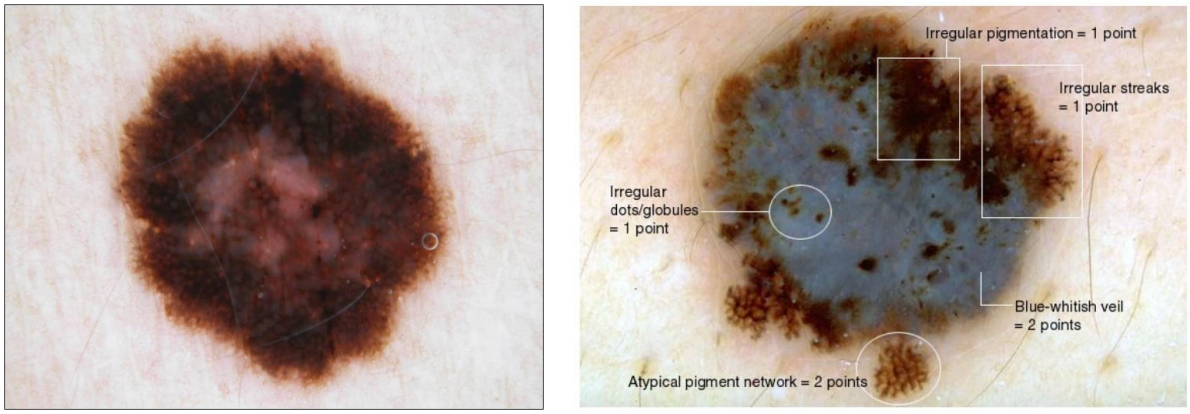

Figure 1.1: Visual display of two dermoscopic images. The left image shows a melanocytic lesion obtained from the ISIC 2019 dataset [3–5]. The right image shows a melanocytic lesion with several dermoscopic criteria commonly assessed by specialists during dermoscopic image analysis [9].

The application of Deep Learning (DL) methods, especially Convolutional Neural Networks (CNNs), to dermoscopic image analysis has increased significantly in recent years [10]. CNNs have long been at the forefront of various Computer Vision (CV) tasks, demonstrating outstanding performance in a wide range of medical image analysis applications. These networks are well suited for tasks such as lesion detection, segmentation, and classification [11, 12]. However, dermoscopic image classification poses specific challenges due to the complex and diverse nature of these images, especially when it comes to identifying the *key patches* that embody the different Regions of Interest (ROIs) within the dermoscopic

image. Consequently, the ability of deep models to identify the most relevant patches in a dermoscopy image has the potential to provide valuable clinical insight to physicians during the diagnostic process.

1.2 Problem Formulation

In clinical practice, dermatologists diagnose skin cancer using established criteria, such as the 7-point checklist criterion [9]. These criteria determine whether a skin lesion is a melanoma based on specific dermoscopic characteristics present in the lesion. Notably, the 7-point criterion [9] focuses solely on the presence or absence of these dermoscopic features within the lesion, regardless of their spatial arrangement. DL models have the potential to identify ROIs that correspond to these dermoscopic criteria, thereby providing dermatologists with significant assistance in their diagnostic process [13].

Multiple Instance Learning (MIL) is a framework in which an image is treated as a "bag", and each patch within the image is an "instance". The key idea in MIL is that the classification of the image depends on the presence or absence of "key instances". In the context of the melanoma vs. nevus binary problem, a "key instance" is a patch that belongs to the melanoma class. If all patches in the bag are from the nevus class (indicating no "key instances"), the image is classified as a nevus. Conversely, if there's at least one "key instance" in the bag (indicating that there's at least one melanoma patch), the entire image is classified as melanoma. This MIL approach resembles the 7-point checklist method in that both rely on the presence or absence of certain regions in the image to diagnose the lesion, regardless of the spatial location of these regions. Therefore, we can think of the MIL framework as an approximation of the 7-point checklist criterion in the world of DL.

Most deep MIL techniques applied to medical image analysis currently rely on CNN-based backbones. However, Vision Transformers (ViTs) have gained immense popularity in CV for their ability to match and even surpass state-of-the-art CNN models. ViTs use a mechanism known as Multi-head Self-Attention (MSA) to extract complex features based on the pairwise relationships between different patches in the image, while taking into account their spatial location within the image. This approach differs from that used by dermatologists. As mentioned above, the 7-point checklist criterion [9] focuses only on the presence or absence of specific dermoscopic features, regardless of their location on the image. This contrast highlights a significant difference between ViTs and clinical diagnostic methods.

Thus, we are faced with two opposing paradigms. On one side is the ViT architecture, which uses correlations between patches in the image to make predictions, taking into account spatial information. On the other is the MIL framework, analogous to the 7-point criterion [9], where specific patches are crucial for classification, regardless of their spatial location. This contrast raises the question of whether all patches are relevant for skin cancer diagnosis and whether the spatial arrangement of the patches in the image has an influence on diagnostic accuracy.

1.3 Objectives

The relationship between the ViT and MIL serves as the primary motivation for this work, which aims to address two key questions in the dermoscopic image analysis:

1. Are all patches within a dermoscopic image equally relevant for a correct diagnosis of skin cancer, or do deep models primarily rely on the identification of specific ROIs?

2. Does the spatial arrangement of patches within the image have a significant impact on the accurate classification of a dermoscopic image?

1.4 Contributions

To address these two fundamental questions, we introduce a two-branch framework. One branch features an Expediting Vision Transformer (EViT) architecture [14], while the other branch accommodates a MIL model. This work focuses on two different challenges: a binary classification task distinguishing melanoma from nevus, and a multi-class problem covering eight different skin disease types.

In the binary classification scenario, we propose a novel two-stage MIL formulation suitable for dermoscopy image processing. We extend the two-step approach to a three-step method that addresses the multi-class problem.

In short, the contributions of this thesis are diverse and include the proposal of a comprehensive framework that merges a ViT-based architecture and the MIL framework to understand the importance of ROIs and spatial arrangement in dermoscopy image diagnosis. We present innovative deep MIL approaches that address both binary and multi-class scenarios that are particularly well suited for dermoscopy image analysis. Our work is complemented by an extensive series of experiments using state-of-the-art backbones, providing valuable insights into the performance of our model.

1.5 Document Organization

This thesis consists of six chapters designed to provide a structured and logical progression of ideas. Chapter 1 sets the scene by addressing the motivation for research in skin cancer diagnosis, highlighting the importance of detecting ROIs in dermoscopy images and the challenges faced by dermatologists. Chapter 2 introduces the core concepts of ViTs and MIL and presents related work in the field of dermoscopy image processing. In chapter 3, our approach, which combines ViTs with MIL, is presented in detail. Chapter 4 provides insights into the experimental set-up, covering datasets, model configurations, and other critical parameters. Chapter 5 presents the experimental results, analyzing the performance of

the model in the binary and multi-class contexts. Finally, Chapter 6 summarises the conclusions drawn from this thesis work and outlines possible future directions.

2

State Of The Art Review

Contents

Chapter 2 is intended to introduce the ViT model and lay the foundation for understanding the concept of MIL. It will also explore the practical applications of both ViT and MIL in the field of medical imaging, with a particular focus on their relevance to skin cancer diagnosis.

2.1 The Vision Transformer

In recent years, the Transformer architecture, first introduced by Vaswani *et al.* [15], has gained prominence in the field of Natural Language Processing (NLP). However, this success was not immediately transferred to the field of CV until Dosovitskiy *et al.* [1] introduced the ViT model [16]. Figure 2.1 provides an overview of the ViT architecture.

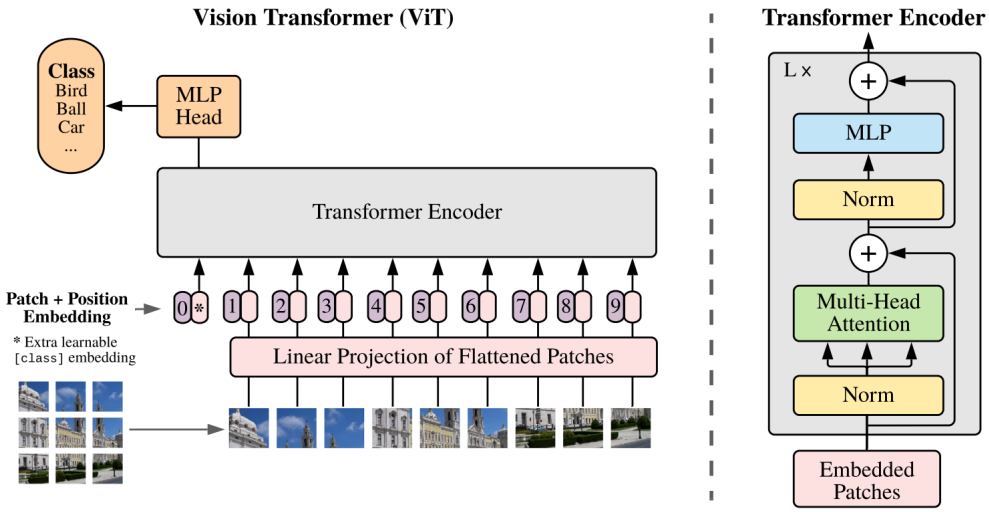

Figure 2.1: Overview of the ViT model [1]. On the left, the diagram illustrates the process of transforming an input image into a sequence of flattened patch embeddings, denoted as $\mathbf{z_0} \in \mathbb{R}^{(N+1,D)}$. This sequence is then fed into the Transformer encoder. The representation of the Classification (CLS) token at the output of the last encoder block is used as input for an Multi-layer Perceptron (MLP) classifier to perform image classification. On the right, an overview of the Transformer encoder introduced by Vaswani *et al.* [15], is presented.

The ViT architecture consists primarily of three key components: an initial pipeline responsible for converting the input image into a sequence of embedded patches, a Transformer encoder block, and a final MLP head responsible for classifying the entire image. The Transformer encoder block remains true to the original design proposed by Vaswani *et al.* [15], which was originally intended to process sequences of tokens. In contrast, CV applications are primarily concerned with images rather than token sequences. To accommodate this fundamental shift in input data, substantial modifications are required to adapt the Transformer architecture for image processing. In the following sections, we will embark on a more comprehensive exploration of the ViT architecture, delving into the intricate details of its constituent blocks.

2.1.1 The Vision Transformer Input

The first step in the ViT framework is to transform the input image into a sequence of embedded patches. Figure 2.2 shows the process of partitioning the input image into patches of uniform size, which initiates the process of transforming the image into a sequence of patch embeddings.

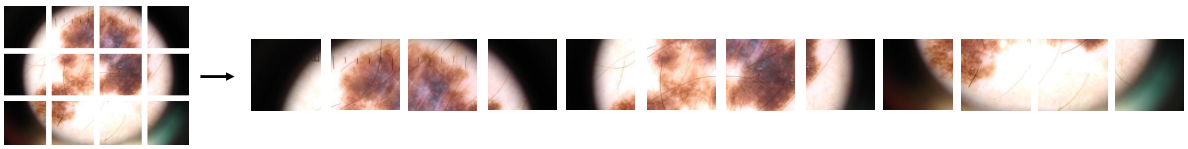

Figure 2.2: Dermoscopy image $X \in \mathbb{R}^{H \times W \times C}$ (where H is the height of the image, W is the width, and C is the number of channels). The first step in the ViT pipeline is to divide this input image into $N = \frac{HW}{P^2}$ patches. In this context, the n-th patch in the image can be described as $\mathbf{x_n} \in \mathbb{R}^{P \times P \times C}$, where P corresponds to the height and width of a patch [3–5].

Consider the task of classifying a given input image $X \in \mathbb{R}^{H \times W \times C}$, where (H, W) represents the image resolution, and C denotes the number of channels. Additionally, assume that each n-th patch is defined as $\mathbf{x_n} \in \mathbb{R}^{P \times P \times C}$, where (P, P) is the patch resolution.

The first step in the ViT pipeline is to transform the input image X into a sequence of flattened patches: $X \in \mathbb{R}^{H \times W \times C} \rightarrow X_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, more precisely:

$$X_p = \begin{bmatrix} \text{——} & \mathbf{x_1}^\top & \text{——} \\ \text{——} & \mathbf{x_2}^\top & \text{——} \\ & \vdots & \\ \text{——} & \mathbf{x_N}^\top & \text{——} \end{bmatrix} \in \mathbb{R}^{N \times (P^2 \cdot C)}, \tag{2.1}$$

where $\mathbf{x_n} \in \mathbb{R}^{P^2 \cdot C}$ $\forall_{n=1,\ldots,N}$ are column vectors.

Since the Transformer architecture designed by Vaswani *et al.* [15] uses token embeddings as input, it is necessary to map the flattened patches, $\mathbf{x_n}$ $\forall_{n=1,\ldots,N}$, into an embedding space denoted as D. To accomplish this transformation, a trainable linear matrix $\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$ is used. This matrix learns to linearly project each flattened patch into the D dimensional space. Thus, the sequence of flattened patch embeddings can be defined as follows:

$$X_p \cdot \mathbf{E} = \begin{bmatrix} \text{——} & \mathbf{x_1}^\top \mathbf{E} & \text{——} \\ \text{——} & \mathbf{x_2}^\top \mathbf{E} & \text{——} \\ & \vdots & \\ \text{——} & \mathbf{x_N}^\top \mathbf{E} & \text{——} \end{bmatrix} \in \mathbb{R}^{N \times D}, \quad \text{where: } \mathbf{x_n}^\top \mathbf{E} \in \mathbb{R}^{1 \times D} \; \forall_{n=1,\ldots,N}. \tag{2.2}$$

Similar to the BERT model [17], a trainable CLS token, denoted as $\mathbf{x_{class}} \in \mathbb{R}^D$, is concatenated to the sequence of patch embeddings. The state of this CLS token at the output of the final ViT encoder

block serves as a representation of the entire image, which is then used for the classification task. This approach is motivated by the fact that, through the operations of MSA, the CLS token learns a representation that can be seen as an aggregation of all the patch embedding representations [17, 18]. Consequently, it is useful to perform image classification based on the information contained in this CLS embedding.

Until the introduction of the Transformer architecture by Vaswani *et al.* [15], most of the Sequence to Sequence (Seq2Seq) models in the field of NLP were based on recurrent architectures, such as Recurrent Neural Networks (RNNs), Gated Recurrent Units (GRUs), and Long Short-Term Memory (LSTM) networks. These recurrent architectures process each input in a given sequence one step at a time, i.e., each token is processed sequentially. This means that these architectures have an inherent notion of the order within the sequence [19, 20].

In contrast to recurrent architectures, the Transformer encoder receives a sequence of embedded tokens in parallel, not sequentially. This means that in the context of NLP, there was a need to inject the model with information regarding the position of the tokens within the sequence [15].

As a result of the direct adaptation of concepts from NLP to CV, the ViT model uses the same positional encoding method as the Bert model [17]. This positional encoding technique involves the addition of a learnable positional embedding matrix, denoted as $\mathbf{E_{pos}} \in \mathbb{R}^{(N+1)\times D}$, to the sequence of flattened patch embeddings represented in (2.2). Consequently, we can define the input for the initial ViT encoder block as follows:

$$\mathbf{z_0} = \begin{bmatrix} \text{——} & \mathbf{x_{class}^\top} & \text{——} \\ \text{——} & \mathbf{x_1^\top E} & \text{——} \\ & \vdots & \\ \text{——} & \mathbf{x_N^\top E} & \text{——} \end{bmatrix} + \mathbf{E_{pos}}, \quad \text{where: } \mathbf{z_0} \in \mathbb{R}^{(N+1)\times D}. \tag{2.3}$$

2.1.2 The Vision Transformer Encoder Block

The ViT encoder is composed of two main components: an MSA block and a MLP block. Prior to each block, Layer Normalization (LN) [21] is applied, and residual connections are added after each block. The ViT model has L encoder blocks, also referred to as ViT layers. A visual representation of the ViT encoder is shown in Figure 2.1.

The MSA mechanism originally introduced by Vaswani *et al.* [15] plays a pivotal role in the Transformer encoder process. This mechanism involves the simultaneous execution of multiple Self-Attention (SA) processes in parallel. To develop a comprehensive understanding of both the SA and MSA mechanisms, we will begin by explaining a single SA process. To do so, we will follow the scheme shown in Figure 2.3, which provides deeper insights into the functionalities of both SA and MSA.

As previously stated, the ViT model comprises L encoder blocks. The input for the l-th encoder block is denoted as $\mathbf{z}_l \in \mathbb{R}^{(N+1)\times D}$ (note that the expression for $\mathbf{z_0}$ is presented in (2.3)). Each encoder block

l ($\forall_{l=1,\dots,L}$) is equipped with three learnable weight matrices: $\mathbf{U_Q}$, $\mathbf{U_K}$, and $\mathbf{U_V} \in \mathbb{R}^{D' \times D}$ (represented as linear layers in Figure 2.3). These weight matrices are initialized using a uniform distribution [22].

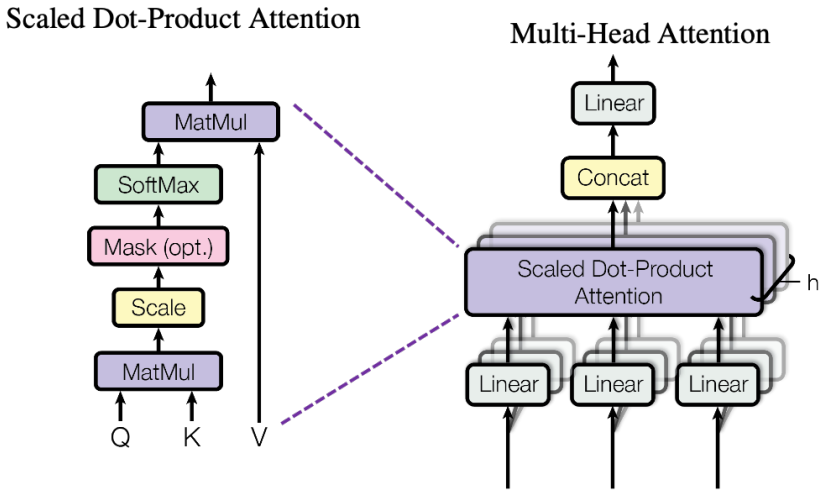

Figure 2.3: MSA pipeline overview. MSA consists in multiple SA mechanisms running in parallel [15].

The matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are computed by taking the dot product between $\mathbf{z}_l$ and the linear layers $\mathbf{U_Q}$, $\mathbf{U_K}$, and $\mathbf{U_V}$, as shown in the following equations:

$$\mathbf{Q} = \mathbf{z}_l \cdot \mathbf{U}_\mathbf{Q}^\top, \quad \mathbf{K} = \mathbf{z}_l \cdot \mathbf{U}_\mathbf{K}^\top, \quad \mathbf{V} = \mathbf{z}_l \cdot \mathbf{U}_\mathbf{V}^\top, \quad \text{where: } \mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{(N+1) \times D'}. \tag{2.4}$$

These matrices, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, serve as projections of the input into three respective subspaces. Each row of these matrices contains information related to the patch embedding located at the corresponding row of the encoder's input, $\mathbf{z}_l$.

From the scaled dot-product attention block, shown in figure 2.3, will result what is called a SA Head:

$$\text{SA}(\mathbf{z}_l) = \mathbf{AV} \in \mathbb{R}^{(N+1) \times D'}, \qquad \text{where: } \mathbf{A} = softmax\left(\frac{\mathbf{QK}^\top}{\sqrt{D'}}\right) \in \mathbb{R}^{(N+1) \times (N+1)}. \tag{2.5}$$

Note that the *softmax* function applied to the attention map $\mathbf{A}$ in (2.5) is a row-wise operation. The elements of the i-th row of the attention map $\mathbf{A}$ represent the attention weights from the i-th token to all other tokens [14]. Essentially, the attention map $\mathbf{A}$ contains information about the pairwise similarity between every pair of patches in the image [23, 24]. For instance, the attention weight A_{ij} (the entry in row i and column j of the attention map $\mathbf{A}$) provides information into the pairwise similarity between patches i and j. Thus, elements in the i-th row with higher attention weights correspond to patches that are most similar to the i-th patch [1].

As it is possible to see from (2.5), the SA Head is equal to the dot product between the attention map $\mathbf{A}$ and the value matrix $\mathbf{V}$. Each element of the $\text{SA}(\mathbf{z}_l)$, can be computed by the following linear

combination:

$$\text{SA}_{ij} = \sum_{n=0}^{N} A_{in} \cdot V_{nj}, \qquad \forall_{i=0,...,N \text{ and } j=0,...,(D'-1)}. \tag{2.6}$$

Each i-th row of the SA Head provides a representation of the i-th patch. This representation is constructed as a linear combination between the value representations of all the n patches, V_{nj}, with the respective attention weights in the i-th row of the attention map $\mathbf{A}$, denoted as A_{in} ($\forall_{n=0,...,N}$). These attention weights, A_{in}, serve as weights that determine the relevance (or "attention") that the n-th patch will have in constructing the new features of the i-th patch. Thus, it is reasonable to conclude that the SA mechanism generates new patch features based on the pairwise correlations between each patch and all other patches in the image [14].

In MSA, multiple SA mechanisms operate in parallel. Given that the learnable weight matrices $\mathbf{U_Q}$, $\mathbf{U_K}$, and $\mathbf{U_V}$ are uniformly initialized, having several SA mechanisms enables the ViT to have multiple distinct attention maps. This allows the model to capture different types of relationships between the patches in the image [15]. The MSA mechanism combines these individual SA Heads by concatenating them and applying a linear layer (represented by $\mathbf{U_{msa}} \in \mathbb{R}^{D \times (H \cdot D')}$) to map the patch embedding sequence to the D-dimensional space. For H SA processes, the output of the MSA mechanism is defined as follow:

$$\text{MSA} = \begin{bmatrix} \text{SA}_1(\mathbf{z}_l) & \text{SA}_2(\mathbf{z}_l) & ... & \text{SA}_H(\mathbf{z}_l) \end{bmatrix} \cdot \mathbf{U}_{\mathbf{msa}}^{\top}, \qquad \text{where: } \text{MSA}(\mathbf{z}_l) \in \mathbb{R}^{(N+1) \times D}. \tag{2.7}$$

In short, the MSA mechanism produces highly complex features that are heavily based on the similarity between the patches that comprise the image. Considering the Transformer encoder architecture presented in figure 2.1, the output of the encoder is defined as follows:

$$\begin{aligned} \mathbf{z'}_l &= \text{MSA}(\text{LN}(\mathbf{z}_{l-1})) + \mathbf{z}_{l-1}, \qquad l = 1, ..., L \\ \mathbf{z}_l &= \text{MLP}(\text{LN}(\mathbf{z'}_l)) + \mathbf{z'}_l, \qquad l = 1, ..., L. \end{aligned} \tag{2.8}$$

2.1.3 The Vision Transformer Output And Interpretability

As previously stated, the state of the CLS token at the output of the last encoder block, denoted as $\mathbf{z}_L^{(0)}$, serves as an image representation that is fed into a Fully Connected Layer (FC Layer) to obtain the predicted image label.

Figure 2.4 presents a visual representation of what the ViT model perceives. In this particular example, the goal of the model is to classify the dog in the image. When considering the case where $l = 8$, it becomes evident that the model highlights the patches containing the dog. This is reflected in higher attention weights assigned to these patches, effectively "filtering out" the background. Although the ViT model isn't inherently interpretable, Figure 2.4 shows that it is possible to understand which patches the

model considers more relevant.

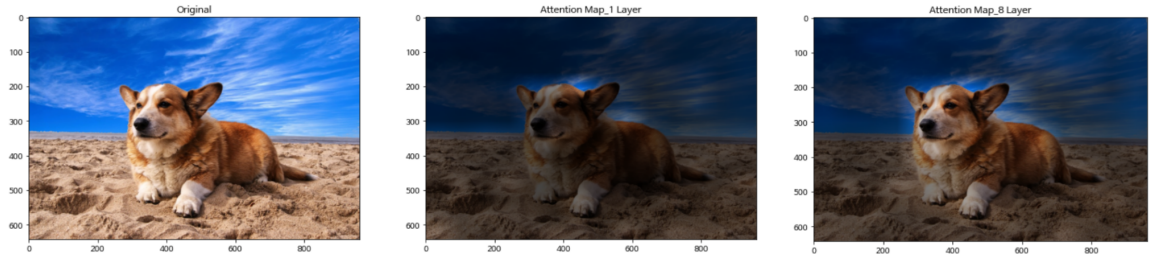

Figure 2.4: Attention map visualization. On the left, it is displayed the original input image. In the middle, it is possible to see the effects of the attention map in the first Transformer encoder layer ($l = 1$). On the right, a visualization of the attention map for the eighth encoder block ($l = 8$) is presented [25].

2.2 The Expediting Vision Transformer (EViT)

Thanks to the MSA mechanism, the ViT model can identify the most relevant patches to some extent. However, the EViT model introduced by Liang *et al.* [14] is a variant of the ViT that takes a more direct approach by distinguishing "attentive" from "inattentive" patches. In this section, we will take a closer look at the EViT architecture and explore its inner workings. In Figure 2.5 it is shown the architecture of the EViT encoder. The EViT is very similar to the ViT. The only difference is that it uses the attention weights of the first row of the attention map $\mathbf{A}$ to determine the attentive and inattentive patches.

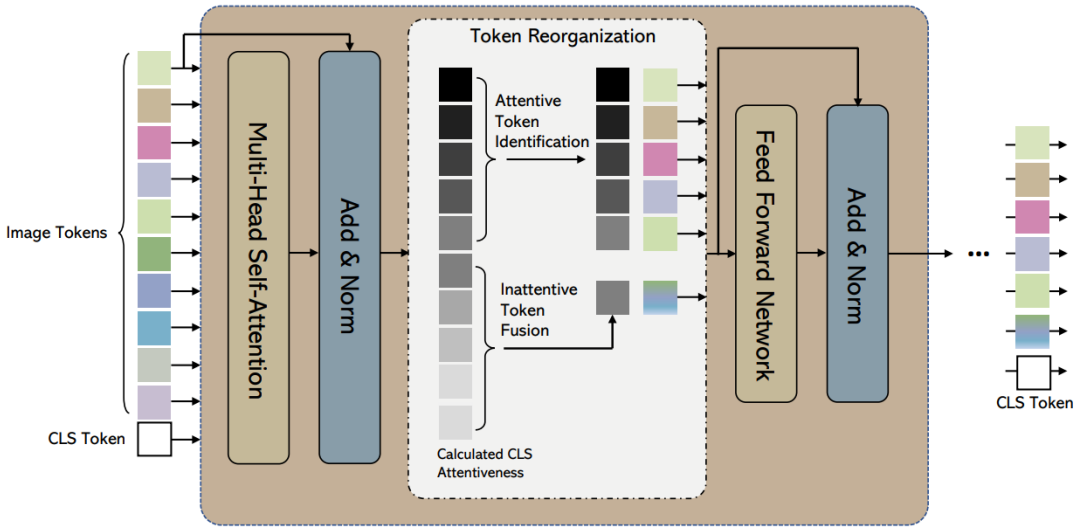

Figure 2.5: Illustration of the EViT encoder block. The EViT encoder is identical to the ViT encoder. The main difference lies in EViT's integration of the token reorganization technique before the MLP block. By calculating CLS attentiveness, EViT distinguishes between attentive and inattentive tokens. Attentive tokens are propagated forward in the network, while inattentive tokens are merged into an embedded representation [14].

The EViT model uses information from the first row of the attention map, $\mathbf{A}$, to determine the importance score of each patch's contribution to the model's output. Based on these scores, patches are categorized as either "attentive" or "inattentive" tokens. The k most relevant patches, i.e., the patches that translate greater importance to the model's output, are labeled attentive. All the other patches are labeled inattentive and are subsequently merged into a single embedding.

Figure 2.6 provides visual examples of the process of discarding inattentive patches. Inspection of the image shows that the majority of the removed patches correspond to background elements. This demonstrates how EViT efficiently discards less relevant information.

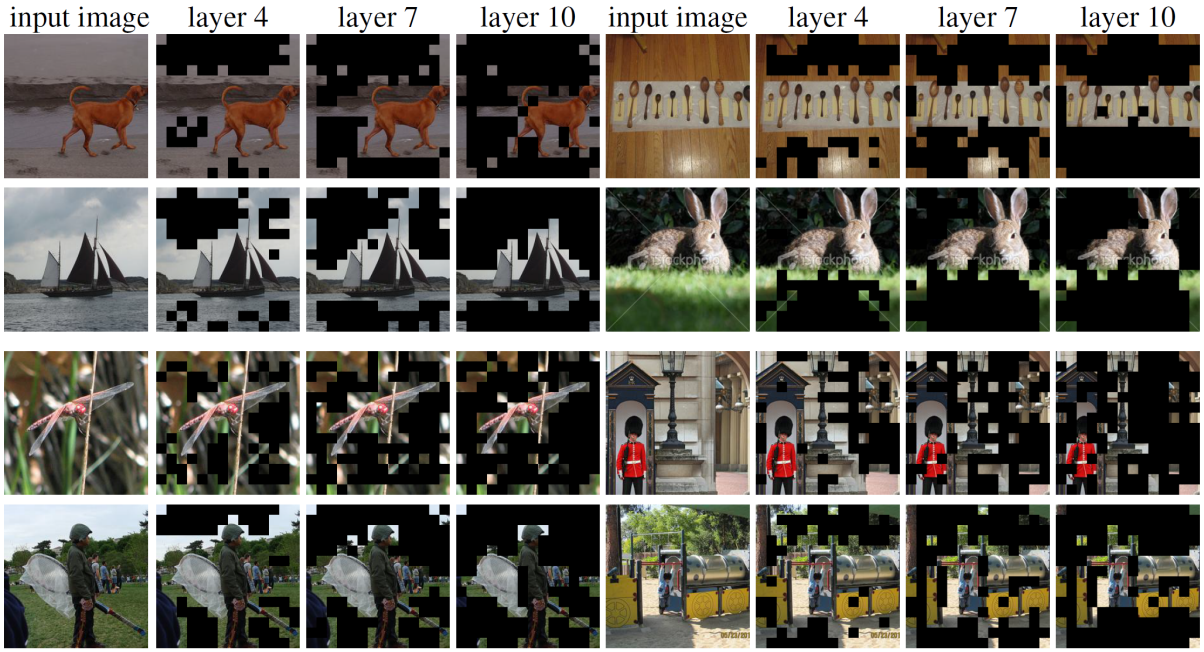

Figure 2.6: Some examples of the process of discarding inattentive patches. Inspection of the image shows that the majority of the removed patches correspond to background elements, demonstrating that no crucial information is lost during the token reorganization process [14].

To better understand EViT's selection process, let us denote $\boldsymbol{a}$ as the first row of the attention map $\mathbf{A}$. With this notation, we can define the state of the CLS token at the end of the SA process as $\mathbf{x_{class}} = \boldsymbol{a}\mathbf{V}$, where $\boldsymbol{a} \in \mathbb{R}^{1\times(N+1)}$ and $\mathbf{V} \in \mathbb{R}^{(N+1)\times D'}$. Since the state of this token at the output of the last encoder is used by the model to represent the entire image, and since $\mathbf{V}$ contains information about each patch embedding, we can reasonably conclude that the i-th element of $\boldsymbol{a}$, a_i, quantifies the importance of the corresponding i-th patch in the model's output.

In the context of MSA, where we have multiple SA heads, we get multiple $\boldsymbol{a}$ vectors, one from each attention head. Consequently, we can calculate the average attentiveness across all heads ($\overline{\boldsymbol{a}} = \sum_{h=1}^{H} \boldsymbol{a}^{(h)}/H$) and use it to determine the k most significant patches, with k as a hyperparameter. The remaining tokens are then merged into a single token, denoted as $\boldsymbol{x}_{fused} = \sum_{i\in\mathcal{N}} \overline{a_i}\boldsymbol{x}_i$, where $\mathcal{N}$

represents the set of inattentive tokens.

The fusion of inattentive tokens in the EViT significantly improves efficiency by reducing the number of tokens throughout the network. This reduction in tokens not only improves computational efficiency, but also reduces the computational complexity of the MSA mechanism in deeper layers. Consequently, this token reorganization technique significantly increases EViT's efficiency compared to the original ViT, all while maintaining its accuracy levels.

Both ViT and EViT have demonstrated competitive performance, often outperforming existing state-of-the-art models, on various image classification datasets. However, this thesis primarily focuses on dermoscopy image classification. To address this specific context, we will review two studies that apply ViT-based architectures to skin cancer classification tasks.

2.3 Vision Transformer Architectures In Dermoscopy Image Processing

Since the introduction of ViT, there have been many works published that proposed various ViT variants to solve CV problems. Especially in the field of classification of dermoscopy images, compelling new articles show that ViT has surpassed state-of-the-art methods [2, 26].

Aladhadh *et al.* [2] presented the Medical Vision Transformer (MVT), an effective approach for skin cancer classification in dermoscopy images using a two-tier framework. In the first stage of this framework, various data augmentation techniques, including brightness adjustment, contrast enhancement, geometric transformations, and more, are applied. This augmentation significantly increases the number of image samples available for training, which proves advantageous for ViT-based architectures, which are known for their strong generalization when trained on extensive data [27]. The second stage of the framework uses a ViT model. This ViT model divides the input image into 7×7 patches, followed by the standard ViT patch processing pipeline, as explained in section 2.1. Figure 2.7 illustrates the two-tier framework proposed by the authors.

Another significant contribution of this work was the establishment of an experimental setup for training a ViT model for skin cancer classification. The researchers used the HAM10000 dataset [3] for both training and evaluation. The evaluation metrics included accuracy, recall, precision, and F1 score. The results showed that the ViT outperformed the state-of-the-art CNN-based frameworks [11, 12]. In particular, the authors emphasized that the performance of the ViT significantly decreased without data augmentation, especially in the classification of melanoma. This underlines the need of the ViT model for extensive training data to achieve optimal performance.

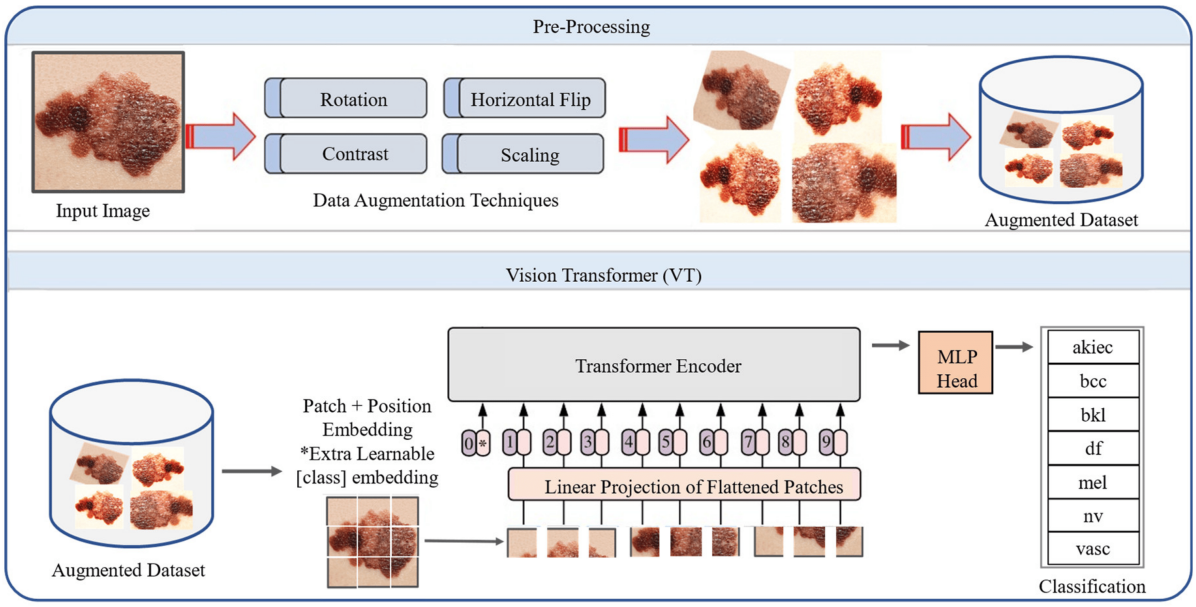

Figure 2.7: Illustration of the two-tier MVT framework [2]. The first stage involves a data pre-processing block, where various data augmentation techniques are applied to create an augmented dataset. In the second stage, the ViT model, originally introduced by Dosovitskiy *et al.* [1], is trained using the augmented dataset.

Figure 2.8 shows heatmaps of some dermoscopy images generated by a Grad-CAM [28] method. These heatmaps effectively highlight the location of the lesions, demonstrating the model's ability to provide some information about the ROIs in the image. Such insights can be particularly helpful in the clinical diagnosis of skin cancer lesions.

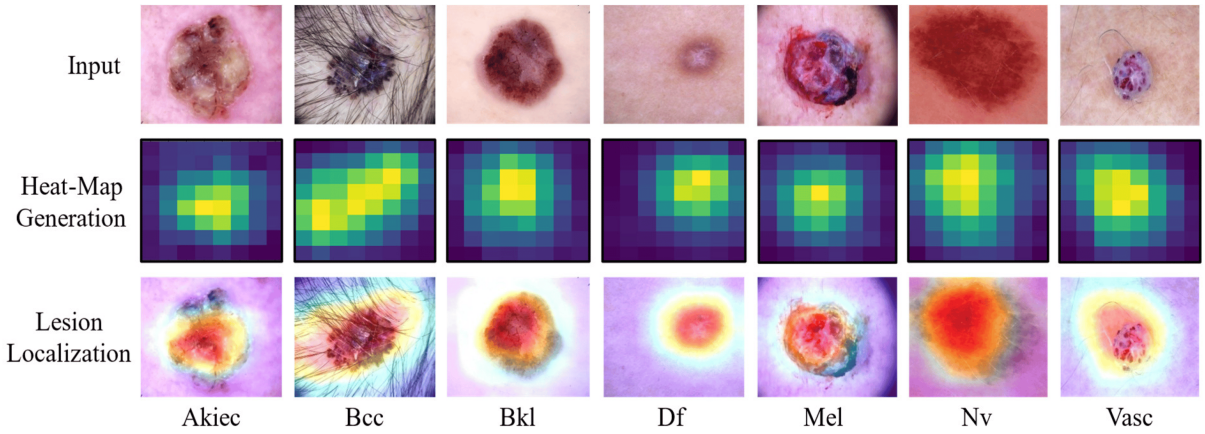

Figure 2.8: Visualization of the heatmaps for dermoscopic images from the HAM10000 dataset [3]. Each column corresponds to a type of skin lesion. Akiec for Actinic keratosis, Bcc for Basal cell carcinoma, Bkl for Benign keratosis, Df for Dermatofibroma, Mel stands for Melanoma, Nv for Nevus, and Vasc for Vascular [2].

The work presented by Aladhadh *et al.* [2] makes notable contributions, highlighting the competi-

tiveness of ViT models over CNN-based state-of-the-art counterparts. However, it is important to highlight certain areas where the article falls short. First, the study's exclusive reliance on the HAM10000 dataset [3] for evaluation raises concerns regarding the model's generalization capability. The absence of validation on diverse dermoscopy datasets, including the publicly available ISIC datasets [4], limits our understanding of the model's adaptability to different clinical contexts. Second, there is a lack of an in-depth exploration of the MSA mechanism within the ViT architecture.

In the same year, Xin *et al.* [26] introduced the SkinTrans model, a ViT-based architecture that uses a multi-scale sliding window approach to divide an image into different types of patches. The SkinTrans model mainly consists of a three-step procedure.

First, it uses multi-scale and overlapping windows to serialize the image, which facilitates the generation of multi-scale patch embeddings. Second, it implements the ViT network as originally proposed by Dosovitskiy *et al.* [1]. Finally, the authors incorporate a contrastive learning loss function. This allows the model to capture specific features that distinguish one class from another, as well as more complex features that are shared by instances of the same class [29].

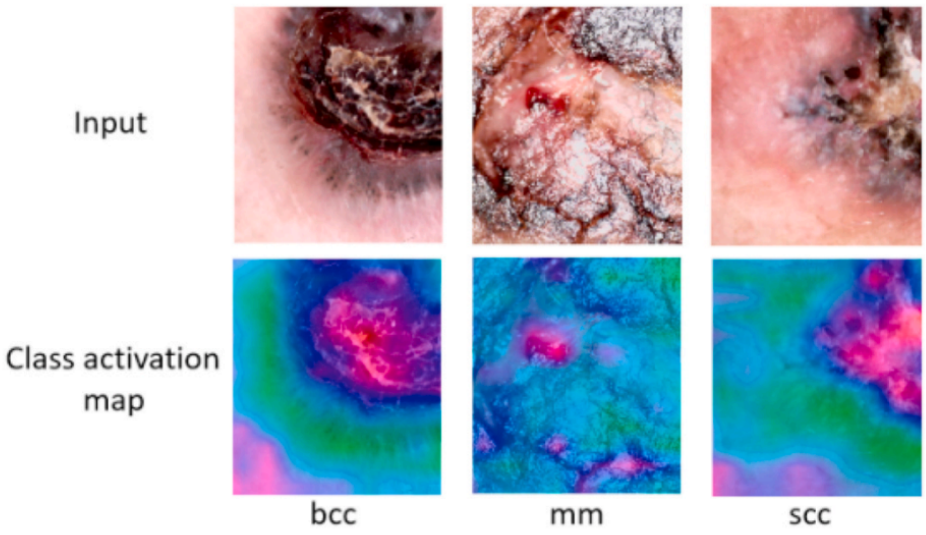

Figure 2.9: Illustration of the class activation maps created using the Grad-CAM method. In the image, 'bbc' represents Basal cell carcinoma, 'mm' corresponds to Malignant melanoma, and 'scc' stands for Squamous cell carcinoma, as defined in [26].

Similar to the MVT [2], the SkinTrans framework incorporates data augmentation and data normalization techniques during the initial pre-processing stage. The authors of the SkinTrans model also used the HAM10000 dataset [3] for both training and evaluation, and extended their evaluation to a smaller private dataset to measure model generalization. Xin *et al.* [26] demonstrated that the SkinTrans model competes favorably with established CNN models.

Figure 2.9 shows the class activation maps for each class on the authors' private dataset. The presence of red areas in these maps highlights the importance of certain regions in skin cancer diagnosis, demonstrating that the SkinTrans model can identify ROIs in the dermoscopy image, to some extent.

Xin *et al.* [26], demonstrated that a ViT-based network can successfully generalize when evaluated in other dermoscopy datasets. However, their study did not explore the potential of the MSA mechanism to detect whether dermoscopy images exhibit a distinction between attentive and inattentive patches.

2.3.1 Conclusions

In clinical practice, dermatologists rely on established criteria such as the 7-point checklist criterion [9] to diagnose skin cancer lesions. Essentially, the 7-point checklist diagnoses a skin lesion as melanoma based on the presence or absence of specific dermoscopic criteria within the lesion. Consequently, the different ROIs learned by deep models have the potential to significantly benefit clinical practice, as they may contain information related to these dermoscopy criteria. The 7-point criterion implies that only a few ROIs within a dermoscopy image are relevant for the diagnosis of the respective skin lesion. This leads to the question of whether deep models, such as ViT-based architectures, can accurately identify these ROIs instead of treating the entire lesion as a single ROI.

It is worth noting that the 7-point criterion [9] implies that the position of the dermoscopy criteria in the lesion does not matter for the diagnostic process, their presence or absence is what counts. However, all ViT-based models discussed so far remain faithful to the original Transformer architecture introduced by Vaswani *et al.* [15]. Consequently, they use positional encoding techniques to provide the model with a spatial understanding of the patch positions within the image. As shown by Dosovitskiy *et al.* [1], the use of positional encoding slightly improves model performance. This improvement is particularly relevant for most datasets, where adjacent patches are likely to contain similar content. In contrast, dermoscopy images are inherently complex, making it difficult to distinguish specific concepts within the image.

In summary, the ViT architecture shows great promise, not only in terms of performance, but also in using the SA mechanism to identify patches critical to the model's output. However, these ViT architectures differ from some of the methods used by dermatologists in clinical practice. For example, the 7-point criterion bases its diagnosis solely on the presence of specific dermoscopic concepts in the image, without considering their spatial arrangement. On the other hand, ViT architectures construct complex features that rely heavily on patch correlations, taking into account their spatial location. To be more closely aligned with the diagnostic practices of dermatologists, we will explore a weakly supervised learning technique known as MIL. MIL can be considered as a suitable approximation of the 7-point checklist criterion, as it allows independent scoring of each patch, with the image classification depending on these individual scores. In MIL, there is no dependency or ordering between patches, similar to the way the 7-point criterion analyzes dermoscopy images.

In the following section, we will delve into the MIL framework and outline a general formulation for creating a MIL classifier.

2.4 Multiple Instance Learning

MIL is a type of weakly supervised learning, where the training instances are arranged in groups called "bags", and the model is given only a bag label during training [30, 31]. In the standard MIL assumption, a bag is classified as positive if and only if it contains at least one positive instance, whereas a bag consisting solely of negative instances is assigned a negative label [32].

MIL is often used when dealing with weakly annotated data, a situation frequently encountered in the field of CV [33]. Weakly annotated data in CV refers to scenarios where the dataset's images are associated only with class labels and lack annotations depicting specific ROIs [34]. This scenario is notably applicable when dealing with dermoscopy image datasets. In this context, a dermoscopy image is conceptualized as a "bag", and the constituent patches within the image serve as the independent "instances" that collectively form the bag. In the problem of diagnosing dermoscopy lesions as either melanomas or nevi, the melanoma class can be designated as positive and the nevus class as negative.

2.4.1 The Standard Multiple Instance Learning Assumption

The MIL assumption aligns with the conventional binary classification problem, where there exist only two distinct classes: positive ($y = 1$) and negative ($y = 0$). In the MIL approach, instances are grouped into bags of size N, $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$, where N can vary from bag to bag. As mentioned previously, a bag is classified as negative if it solely contains negative instances. Therefore, we establish the bag label, denoted as Y, using the following formulation:

$$Y = \begin{cases} 0, & \text{iff } \sum_n y_n = 0 \\ 1, & \text{otherwise} \end{cases}, \tag{2.9}$$

or even in a more compact form:

$$Y = \max_n \{y_n\}, \tag{2.10}$$

where $y_1, y_2, \ldots, y_N$ are the respective instance labels.

According to this MIL formulation, the instances within a bag should not have any dependency or ordering among themselves. This implies that a MIL model must be permutation-invariant [34].

2.4.2 A Multiple Instance Learning Classifier Framework

Ilse *et al.* [34] proposed an alternative training methodology for a MIL model, opting for the optimization of the log-likelihood function. In this revised approach, the bag label is represented as a Bernoulli distribution parameterized by $\theta(X) \in [0, 1]$. Essentially, $\theta(X)$ denotes the probability of the bag label being $Y = 1$, which corresponds to the likelihood of the MIL model classifying the dermoscopy image as a melanoma.

Any deep MIL classifier takes as input a bag of embedded instances, denoted by:

$$X = \begin{bmatrix} — & \mathbf{x_1^\top} & — \\ — & \mathbf{x_2^\top} & — \\ & \vdots & \\ — & \mathbf{x_N^\top} & — \end{bmatrix} \in \mathbb{R}^{N \times D}, \tag{2.11}$$

where D represents the embedding dimension, and N denotes the number of instances within a given bag. The instances within such a bag are defined as column vectors, $\mathbf{x_n} \in \mathbb{R}^D \; \forall_{n=1,\dots,N}$. As previously mentioned, within the context of skin cancer diagnosis, a bag corresponds to a dermoscopy image, represented as $X \in \mathbb{R}^{H \times W \times C}$ (with H denoting the image's height, W the width, and C the number of channels). Therefore, any deep MIL classifier framework requires the use of a patch extractor, denoted as E, responsible for converting a dermoscopy image into a bag of embedded patches. The function of this patch extractor, $E : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{N \times D}$, is to perform transformations on each patch representation independently. That is, the creation of each patch embedding should depend only on the patch itself and not on the other patches in the dermoscopy image. For a visual representation of this transformation process, see Figures 2.10 and 2.11, which provide a high-level overview of the operations conducted by the patch extractor E.

Ilse *et al.* [34] proposed that any MIL-based bag classification framework can be generalized by employing a three-step procedure, formulated as follows:

i. Each instance is independently transformed using a function f.

ii. The transformed instances (by f) are combined by a permutation-invariant function ϕ, called MIL pooling.

iii. The resulting representation of the combined instances is transformed by a function g.

The choice of functions f, ϕ, and g determines the specific type of MIL approach used for bag classification. There are primarily two prominent deep MIL approaches for bag classification: the *instance-level* and the *embedding-level* approaches. In the following sections, we will illustrate how to formulate both *instance-level* and *embedding-level* approaches using the three-step method proposed by Ilse *et al.* [34].

2.4.2.A *Instance-level* Approach

The three-step method for the *instance-level* approach, along with the transformation executed by the patch extractor E, is shown in Figure 2.10.

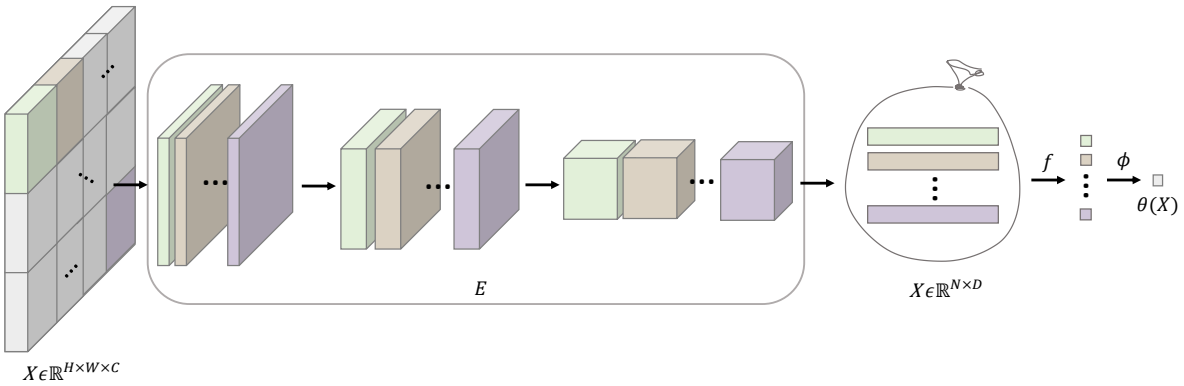

Figure 2.10: *Instance-level* MIL model pipeline for image classification. For a given input image $X \in \mathbb{R}^{H \times W \times C}$, the patch extractor, E, transforms such an image into a bag of embedded patches, $X \in \mathbb{R}^{N \times D}$. These bags of patch embeddings correspond to the input for the *instance-level* MIL classifier. The function f transforms each patch embedding into a respective patch probability of the melanoma class, $f : \mathbb{R}^D \rightarrow \mathcal{H}$, where $\mathcal{H} \in [0, 1]$. The MIL pooling function, $\phi : \mathcal{H}^N \rightarrow \mathcal{H}$, combines all the melanoma patch probabilities into a melanoma bag probability $\theta(X) \in \mathcal{H}$. The function g corresponds to the identity function.

To formulate an *instance-level* classifier, the three-step method should be defined as follows:

i. The function f is designed as an *instance-level* classifier. Given that a patch embedding is denoted as $\mathbf{x_n} \in \mathbb{R}^D$ ($\forall_{n=1,\ldots,N}$), the function $f(\mathbf{x_n})$ computes the probability of the positive class (i.e., the melanoma class) for each patch, $f : \mathbb{R}^D \rightarrow [0, 1]$.

ii. The MIL pooling function, ϕ, combines all the individual patch probabilities, $f(\mathbf{x_n})$ $\forall_{n=1,\ldots,N}$, into a bag probability of the melanoma class, represented as $\phi(X) = \theta(X) \in [0, 1]$. To formalize this operation, we introduce the concept of a space $\mathcal{H} \in [0, 1]$, and accordingly, $\phi : \mathcal{H}^N \rightarrow \mathcal{H}$. The MIL pooling function must be permutation-invariant, e.g., the maximum operator:

$$\phi(X) = \max_{n=1,\ldots,N} \{f(\mathbf{x_n})\}. \tag{2.12}$$

or the mean operator:

$$\phi(X) = \frac{1}{N} \sum_{n=1}^{N} f(\mathbf{x_n}). \tag{2.13}$$

iii. Function g corresponds to identity since we already have the bag probability: $\theta(X) = \phi(X)$.

2.4.2.B *Embedding-level* Approach

The three-step method for the *embedding-level* approach, accompanied by the transformation carried out by the patch extractor E, is visually depicted in Figure 2.11.

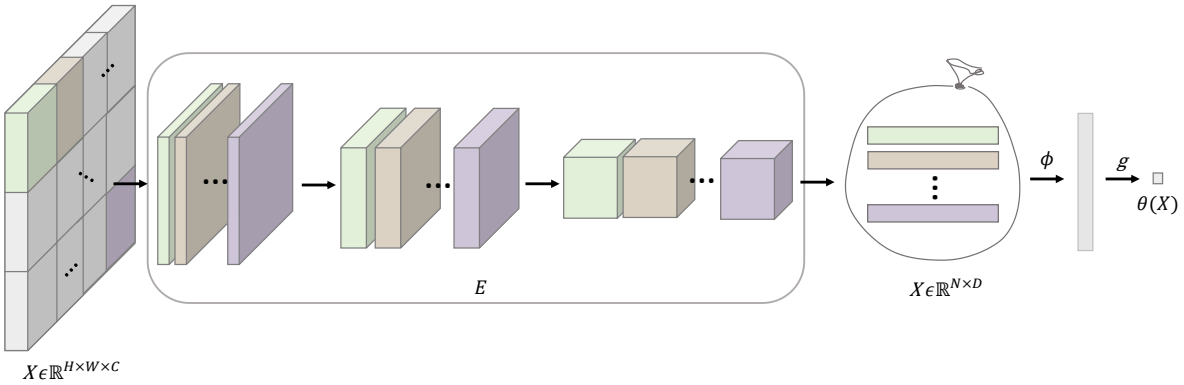

Figure 2.11: *Embedding-level* MIL model pipeline for image classification. For a given input image $X \in \mathbb{R}^{H \times W \times C}$, the patch extractor, E, transforms such an image into a bag of embedded patches: $X \in \mathbb{R}^{N \times D}$. This bag of patch embeddings corresponds to the input of the *embedding-level* MIL classifier. In this context, the function f operates as an extension of the patch extractor E, effectively being defined as $f : \mathbb{R}^{D} \rightarrow \mathbb{R}^{D'}$. The MIL pooling function, ϕ, combines all the individual patch embeddings into a bag embedding representation, $\phi : \mathbb{R}^{N \times D'} \rightarrow \mathbb{R}^{D'}$. Finally, the function $g : \mathbb{R}^{D'} \rightarrow [0, 1]$ corresponds to a bag classifier, i.e., it returns the melanoma bag probability $\theta(X) \in [0, 1]$.

For the *embedding-level* MIL classifier, the three-step method is defined as follows:

i. The function f performs a transformation on each patch embedding, $f : \mathbb{R}^{D} \rightarrow \mathbb{R}^{D'}$. Given that a patch embedding is denoted as $\mathbf{x_n} \in \mathbb{R}^{D}$ ($\forall_{n=1,\ldots,N}$), the function $f(\mathbf{x_n})$ returns a new patch representation in the D'-dimensional space. It is important to note that in this case, the function f extends the feature extraction process initiated by the patch extractor E.

ii. The MIL pooling function, $\phi : \mathbb{R}^{N \times D'} \rightarrow \mathbb{R}^{D'}$, combines all the individual patch embeddings, $f(\mathbf{x_n})$ $\forall_{n=1,\ldots,N}$, into a bag representation $\phi(X) \in \mathbb{R}^{D'}$. The MIL pooling function, ϕ, must be permutation-invariant, e.g., the column-wise global max pooling operator:

$$\forall_{j=1,\ldots,D} : \phi(X)_j = \max_{n=1,\ldots,N} \{f(\mathbf{x_n})_j\}, \tag{2.14}$$

or the column-wise global average pooling operator:

$$\forall_{j=1,\ldots,D} : \phi(X)_j = \frac{1}{N} \sum_{n=1}^{N} f(\mathbf{x_n})_j. \tag{2.15}$$

iii. The function g corresponds to a *bag-level* classifier. The input of this *bag-level* classifier corresponds to the bag's embedding representation obtained from the MIL pooling function, denoted

as $\phi(X) \in \mathbb{R}^{D'}$. In the context of the *embedding-level* approach, the function $g(\phi(X))$ computes the probability of classifying the dermoscopy image as melanoma, denoted as $\theta(X) \in [0, 1]$, with $g : \mathbb{R}^{D'} \rightarrow [0, 1]$.

2.4.2.C Comparison between *Instance-level* and *Embedding-level* Approaches

Both the *instance-level* and *embedding-level* paradigms produce a bag classification, yet through different methods. In the *instance-level* approach, an instance classifier is learned and bag classification is achieved by aggregating independent instance probabilities. Conversely, the *embedding-level* approach aggregates multiple instance embeddings into a compact bag representation and then applies a classifier to estimate the bag probability of the positive class [35, 36].

The *embedding-level* approach generates a low-dimensional bag representation without distinguishing between the constituent instances in the bag. Consequently, it does not allow for the identification of *key instances* within the bag, i.e., the specific instances responsible for the bag classification [34, 36]. Although *embedding-level* approaches typically demonstrate superior performance compared to *instance-level* approaches, they do not provide substantial clinical insight to healthcare professionals when diagnosing skin cancer lesions. Instead, *embedding-level* MIL approaches primarily serve as a "bridge" between MIL frameworks and traditional CNN architectures.

In contrast, the *instance-level* approach provides the ability to identify the specific instances that trigger the bag's classification [34]. This information can provide valuable clinical insight to physicians during the lesion diagnosis process. Since it is possible to map these *key instances* into ROIs within the dermoscopy image. The identification of these ROIs is of vital importance in skin cancer diagnosis. As stated previously, dermatologists often rely on the detection of such regions to apply diagnostic criteria, including the 7-point checklist [9] and the ABCD rule [37], among others, to make accurate diagnoses.

In the following section, we will provide a general overview of some of the works related to MIL in the field of medical image processing.

2.5 Multiple Instance Learning in Medical Image Processing

MIL is particularly well suited to the domain of medical image analysis, where an image can be analogized to a bag, and each instance can be associated with a specific location within the image [38]. In recent years, many applications of MIL in the field of medical image analysis have emerged.

Ilse *et al.* [34], proposed a new attention-based MIL approach. In this new approach, the authors proposed an attention-based MIL pooling operator. This operator consists of a weighted average of the low-dimensional embedding representation of the instances, where the weights are determined by a Neural Network (NN). Consider $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$ the bag of N instance embeddings. The proposed

MIL pooling operator is defined as follows:

$$\phi = \sum_{n=1}^{N} a_n \mathbf{x}_n, \qquad \text{where:} \quad a_n = \frac{\exp\left\{\mathbf{w}^\top \tanh\left(\mathbf{V}\mathbf{x}_n\right)\right\}}{\sum_{j=1}^{N} \exp\left\{\mathbf{w}^\top \tanh\left(\mathbf{V}\mathbf{x}_j\right)\right\}}. \tag{2.16}$$

In this context $\mathbf{x}_n \in \mathbb{R}^D$ $(\forall_{n=1,\ldots,N})$ represents an embedded instance. The vector $\mathbf{w} \in \mathbb{R}^{D'}$ and the matrix $\mathbf{V} \in \mathbb{R}^{D' \times D}$ are learnable parameters.

The MIL pooling operator proposed by Ilse *et al.* [34] can be associated with a version of the attention mechanism [39,40]. However, it differs from the SA mechanism used in the ViT model. Nevertheless, the proposed operator is capable of measuring similarities between instances. To evaluate the performance of this method and to determine whether the attention-based deep MIL pooling operator can provide interpretable results that highlight *key instances* or ROIs, the authors conducted experiments on two real histopathology datasets, specifically related to breast cancer and colon cancer. Figure 2.12 illustrates a heatmap of a histopathology image, which was generated by multiplying each patch by its corresponding attention weight.

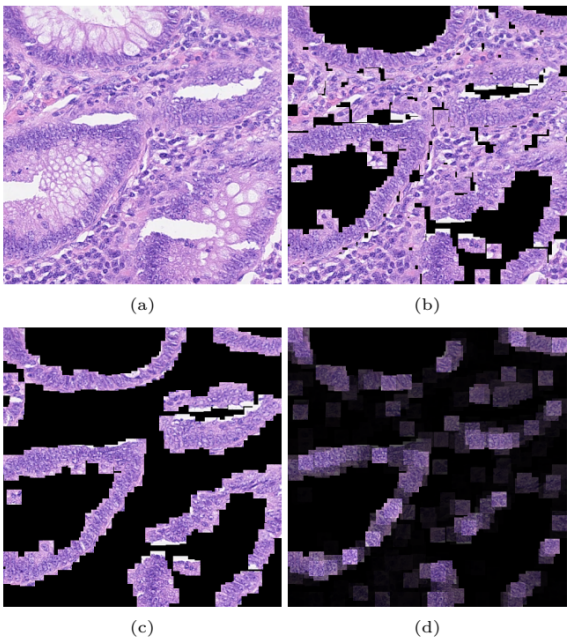

Figure 2.12: (a) H&E stained histology image. (b) 27×27 patches centered around all marked nuclei. (c) Ground truth: Patches that belong to the class epithelial. (d) Heatmap: Every patch from (b) multiplied by its corresponding attention weight [34].

Ilse *et al.* [34] established a compelling connection between MIL and attention mechanisms, demonstrating its competitiveness with established MIL models. However, this study did not introduce a multi-class paradigm for a MIL framework. In addition, the experiments were limited to histopathology datasets focusing on breast and colon cancer.

There is still much to be explored in the field of medical image analysis for skin cancer, especially in the context of dermoscopy image processing. It is worth noting that many of the existing MIL models in skin cancer analysis have been predominantly applied to Whole Slide Images (WSIs) of skin biopsies, rather than dermoscopy images [41, 42]. This highlights the need for further research and development of a MIL framework in the specific area of dermoscopy image analysis for skin cancer diagnosis.

In the context of dermoscopy image classification, Astorino *et al.* [43] introduced a notable MIL framework based on a mixed-integer nonlinear optimization problem. Their method is based on the Support Vector Machine (SVM)-type model for MIL frameworks originally proposed by Andrews *et al.* [44]. The authors conducted experiments using the PH^2 dataset [45]. However, the authors only considered negative examples of common nevi and positive examples of melanoma lesions. Therefore, the experimental dataset contained only 40 images of melanoma and 40 images of common nevi, which is a limited sample size. Although the proposed method showed promising results, the small dataset size limits the analysis of the generalization ability of the proposed method. In the field of dermoscopy image analysis, there remains a need for a more comprehensive study into various deep MIL approaches, including different MIL pooling operators.

The limited study of the MIL framework in dermoscopy image analysis can be attributed to several factors. First, dermoscopy images are relatively small compared to WSIs, which poses significant challenges in extracting high-resolution patches. In addition, while there have been some attempts to adapt MIL to multi-class problems [46–48], in general, there isn't a clear-cut MIL approach that solves the multi-class problem in the context of dermoscopy image analysis. A thorough investigation of different MIL approaches in the field of dermoscopy image analysis may prove to be of significant importance in clinical practice. The ability of the MIL framework to identify ROIs has the potential to provide valuable clinical insights to dermatologists. Therefore, further research in this direction may prove to be highly relevant for improving the diagnosis of skin lesions in dermoscopy images.

2.5.1 Conclusions

MIL is a weakly-supervised framework in which we consider a dermoscopy image as a bag, with each patch within the image treated as an instance. The fundamental principle of MIL is that the classification of the entire image depends solely on the presence or absence of *key instances*. In the context of binary melanoma vs. nevus classification, a *key instance* is defined as a patch belonging to the melanoma class. If all patches within the bag belong to the nevus class, indicating the absence of *key instances*, the image is classified as a nevus. Conversely, if there is at least one *key instance* within the bag, signifying the presence of melanoma characteristics in at least one patch, the entire dermoscopy image is classified as melanoma. This approach is strikingly similar to the 7-point checklist procedure. Both methods base their classification on the presence or absence of specific regions within the image, regardless of

the spatial location of these regions. Therefore, it is reasonable to consider the MIL framework as a robust approximation of the 7-point criterion in the DL world.

2.6 The Connection Between MIL, ViTs, and the 7-Point Criterion

In our current landscape, two apparently opposing paradigms have emerged. On one side, we have the ViT architecture, which exploits the correlated information across different patches while incorporating spatial information into its predictions. On the other side, we have the MIL framework, a concept resembling the 7-point criterion [9], where specific patches are selected for classification regardless of their spatial relationships.

At first glance, the MIL framework and the ViT architecture may seem to be on opposite sides of the spectrum. However, a closer inspection of the ViT architecture reveals some interesting parallels with MIL. As mentioned earlier, the inclusion of positional encoding in ViT comes as a result of the direct adaptation of concepts from NLP to CV. This raises questions about the necessity of positional encoding in ViT architectures designed for vision tasks. To shed further light on the connection between ViTs and the MIL framework, we will show that by **removing positional encoding** from the ViT architecture, ViT models can be considered a deep MIL framework.

Let us consider the ViT encoder (Figure 2.1) with input $z \in \mathbb{R}^{N \times D}$, which represents the sequence of patch embeddings **without** the addition of the positional embedding matrix, $\mathbf{E_{pos}}$:

$$z = \begin{bmatrix} — & \mathbf{z_0^\top} & — \\ — & \mathbf{z_1^\top} & — \\ — & \mathbf{z_1^\top} & — \\ & \vdots & \\ — & \mathbf{z_N^\top} & — \end{bmatrix}, \qquad \text{where: } \mathbf{z_n} \in \mathbb{R}^D \quad \forall_{n=0,\dots,N}. \tag{2.17}$$

In this context, $\mathbf{z_0}$ represents the CLS token. As explained in section 2.1.2, the first step of the SA pipeline involves the computation of matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$. These matrices are obtained through the dot product of the Transformer encoder input, denoted as z, with corresponding learnable linear matrices. For instance, $\mathbf{V}$ is calculated as the product of z and $\mathbf{U_V^\top}$, where $\mathbf{U_V} \in \mathbf{R}^{D' \times D}$, more precisely:

$$\mathbf{V} = z \cdot \mathbf{U_V^\top} = \begin{bmatrix} — & \mathbf{z_0^\top} & — \\ — & \mathbf{z_1^\top} & — \\ — & \mathbf{z_1^\top} & — \\ & \vdots & \\ — & \mathbf{z_N^\top} & — \end{bmatrix} \cdot \begin{bmatrix} | & | & & | \\ \mathbf{u_{V_1}} & \mathbf{u_{V_2}} & \dots & \mathbf{u_{V_N}} \\ | & | & & | \end{bmatrix} = \begin{bmatrix} — & \mathbf{z_0^\top U_V^\top} & — \\ — & \mathbf{z_1^\top U_V^\top} & — \\ — & \mathbf{z_1^\top U_V^\top} & — \\ & \vdots & \\ — & \mathbf{z_N^\top U_V^\top} & — \end{bmatrix}. \tag{2.18}$$

The procedure for computing $\mathbf{Q}$ and $\mathbf{K}$ is identical to that described in (2.4). Moving on in the SA process, our next task is to compute the attention map, $\mathbf{A}$, as described in (2.5). To simplify our sub-

sequent computations, we will exclude the *softmax* and *scale* operators because they do not affect the row-wise permutation invariance. Consequently, we can represent the attention map, denoted as $\mathbf{A} \in \mathbb{R}^{(N+1)\times(N+1)}$, as follows:

$$\begin{aligned}\mathbf{QK}^\top &= \begin{bmatrix} — & \mathbf{z_0}^\top\mathbf{U_Q}^\top & — \\ — & \mathbf{z_1}^\top\mathbf{U_Q}^\top & — \\ — & \mathbf{z_1}^\top\mathbf{U_Q}^\top & — \\ & \vdots & \\ — & \mathbf{z_N}^\top\mathbf{U_Q}^\top & — \end{bmatrix} \cdot \begin{bmatrix} | & | & | & & | \\ \mathbf{U_K z_0} & \mathbf{U_K z_1} & \mathbf{U_K z_2} & \dots & \mathbf{U_K z_N} \\ | & | & | & & | \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \end{bmatrix}. \end{aligned} \tag{2.19}$$

Each element in the resulting matrix, denoted as $\mathbf{QK}^\top$, is a real number. After applying the *scale* factor and performing the *softmax* operation on each row, we derive the attention map $\mathbf{A}$ (2.5), as illustrated in Figure 2.3.

The next step in the Transformer encoder pipeline involves computing the SA Head, which results from the dot product between the attention map $\mathbf{A}$ and the value matrix $\mathbf{V} \in \mathbb{R}^{N\times D'}$. For simplicity, we will continue to consider the attention map $\mathbf{A}$ as simply the dot product between the query matrix $\mathbf{Q}$ and the transpose of the key matrix $\mathbf{K}^\top$. Thus, we can define $\mathbf{A}\cdot\mathbf{V}$ as follows:

$$\begin{aligned}\mathbf{QK}^\top \cdot \mathbf{V} &= \begin{bmatrix} \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_0} & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_1} & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_2} & \dots & \mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_N} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{V_{01}} & \dots & \mathbf{V_{0D'}} \\ \mathbf{V_{11}} & \dots & \mathbf{V_{1D'}} \\ \mathbf{V_{21}} & \dots & \mathbf{V_{2D'}} \\ \vdots & \ddots & \vdots \\ \mathbf{V_{N1}} & \dots & \mathbf{V_{ND'}} \end{bmatrix} \\ &= \begin{bmatrix} \sum_{n=0}^{N}(\mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} & \dots & \sum_{n=0}^{N}(\mathbf{z_0}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{nD'}} \\ \sum_{n=0}^{N}(\mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} & \dots & \sum_{n=0}^{N}(\mathbf{z_1}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} \\ \sum_{n=0}^{N}(\mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} & \dots & \sum_{n=0}^{N}(\mathbf{z_2}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} \\ \vdots & \ddots & \vdots \\ \sum_{n=0}^{N}(\mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} & \dots & \sum_{n=0}^{N}(\mathbf{z_N}^\top\mathbf{U_Q}^\top\mathbf{U_K z_n})\mathbf{V_{n1}} \end{bmatrix} \in \mathbb{R}^{N\times D'}. \end{aligned} \tag{2.20}$$

As discussed in section 2.1.2, the MSA process involves running multiple SA processes in parallel. Thus, when the patches are initially rearranged at the beginning of the ViT model, all SA processes are subject to the same permutations.

The linear matrix $\mathbf{U_{msa}} \in \mathbb{R}^{D\times(H\cdot D')}$ (as seen in (2.7)) performs a linear mapping in the column dimension, specifically in the embedding dimension. Since each column in every SA process involves linear combinations, this transformation remains unaffected by row-wise permutations (i.e., patch per-

mutations). This linear layer can be interpreted as processing each patch representation independently and identically [24]. Consequently, we can confidently affirm that both SA and MSA processes maintain permutation-equivariance in the absence of positional encoding [49].

The LN [21] does not affect this demonstration, and the residual connections also maintain permutation equivariance. Thus, the Transformer encoder remains permutation-equivariant with no positional encoding. The MLP within the Transformer Encoder block can be thought of as a row-based Feedfoward Neural Network (FNN), meaning that each patch embedding is processed independently and identically [24]. Consequently, this MLP preserves the permutation-equivariance of patches within the input sequence.

Consequently, we can deduce that the Transformer encoder block shown in Figure 2.1 exhibits permutation-equivariance in the absence of positional encoding. This insight leads to another revelation: each row of the patch embedding sequence is inherently permutation-invariant. In other words, in the absence of positional encoding, the Transformer encoder maintains permutation-equivariance with respect to the patch permutations within the input sequence. This also means that any patch embedding created by the Transformer encoder is unaffected by permutations of patches within the input sequence.

As we will show in chapter 3, a MIL bag classifier only requires two essential components: a permutation-invariant aggregation function and a classifier. In the absence of positional encoding, the Transformer encoder can be interpreted as functioning similarly to a MIL pooling operator, with the MLP head serving as the classifier. In particular, previous studies have demonstrated the aggregation properties of the Transformer encoder, especially the final encoder block [18]. This conclusion is plausibly consistent with our observations because, as noted above, each patch representation generated by the Transformer encoder block remains immune to permutations between patches within the input sequence. Consequently, the state of the CLS token at the end of the last Transformer encoder block remains unaffected by permutations between the patches in the input image.

Thus, the MIL and ViT frameworks reveal a compelling connection. We can view the MIL and ViT architectures as two seemingly opposing paradigms. On the one hand, the MIL framework closely approximates the 7-point checklist criterion [9] in the context of DL. Both methods score each patch independently, regardless of their spatial positions. They then aggregate these scores to classify the entire image. The ViT architecture, on the other hand, exploits correlations between patches and considers their spatial positions within the image for classification. However, by **removing positional encoding** from the ViT architecture, we can view the ViT model as a highly complex MIL pooling framework, with the Transformer encoder acting as the MIL pooling function.

This intricate relationship between the MIL framework and the ViT architecture serves as the primary motivation for this thesis. To gain a better understanding of the connection between these two different paradigms, it is essential to address two key questions regarding dermoscopy image analysis:

1. Do all patches within a dermoscopy image have the same relevance, or can we distinguish ROIs?

2. Is the spatial positioning of patches in a dermoscopy image crucial for accurate classification?

To address these fundamental questions, we introduce a two-branch framework. One branch consists of an EViT architecture [14], while the other branch contains a MIL model. Our work focuses on tackling two different challenges: a binary classification task distinguishing melanoma from nevus, and a multi-class problem involving eight different skin disease types. In the binary classification scenario, we propose a modified MIL formulation that is more appropriate for dermoscopy images. Due to their smaller size compared to WSIs, extracting high-resolution patches from dermoscopy images is not straightforward. In addition, we present a specialized MIL framework to address the multi-class problem in dermoscopy image classification.

Our work involves a comprehensive exploration of the number of patches used for model inference. Specifically, we experiment with three types of MIL pooling operators: *maximum*, *average*, and *top-k average* operators. To visualize the different ROIs identified by both the MIL and EViT branches, we employ a gradient-based visualization technique inspired by Chefer *et al.* [50]. In addition, we delve into the evaluation of positional encoding within ViT architectures to assess its significance within our framework.

3

Proposed Approach

Contents

In this chapter, we introduce our proposed method, offering a detailed exploration of the model's architecture and delving into the functions of each constituent block. Our model addresses two problems: the binary classification problem of melanoma vs. nevus, and the multi-class problem. With the proposed architecture, we aim to address two key questions in the analysis of dermoscopy images:

1. Are all patches within a dermoscopy image equally relevant, or can we filter the image into specific ROIs?

2. Does the spatial positioning of patches within the image play a critical role in the diagnosis of skin cancer?

The architecture of the proposed approach is depicted in Figure 3.1.

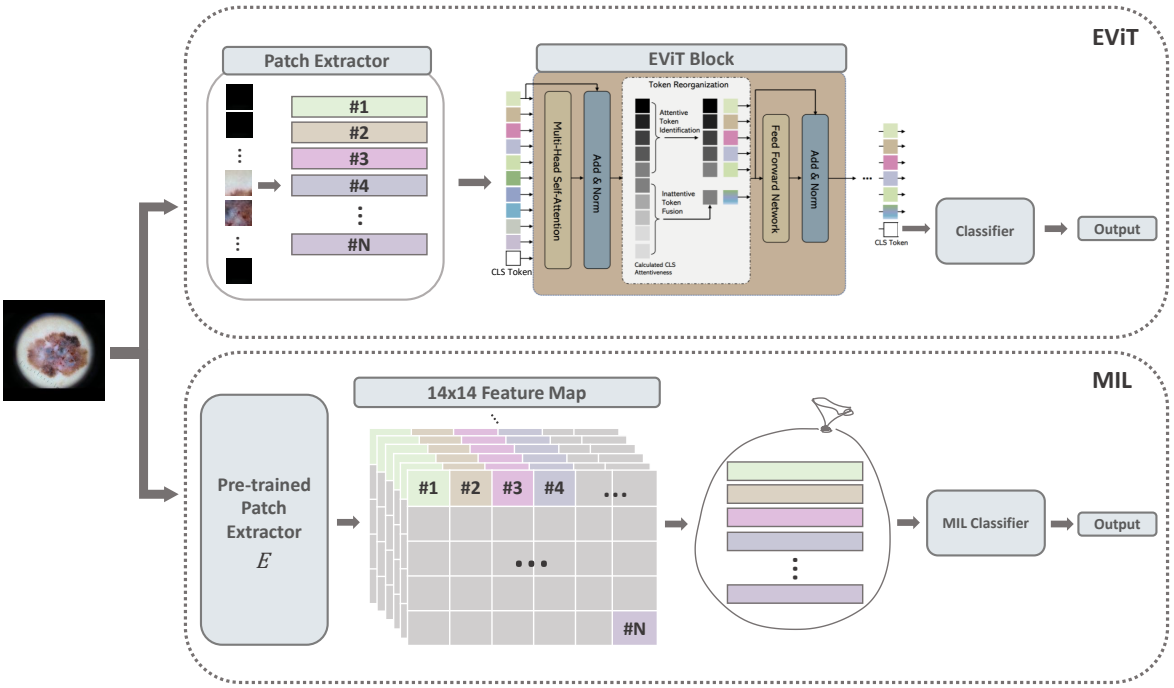

Figure 3.1: Architecture of the proposed approach. The proposed model consists of two primary branches: the EViT branch [14] and the MIL branch. The model takes as input a dermoscopy image represented as $X \in \mathbb{R}^{H \times W \times C}$, where $H = W = 224$ and $C = 3$. The EViT branch is composed of three blocks: the patch extractor block, the EViT encoder block, and the classifier head. Similarly, the MIL branch consists of two main blocks: a pre-trained patch extractor block, E, responsible for extracting a 14×14 feature map, and a MIL classifier. Each branch works independently, producing its own output.

Our proposed approach comprises two primary branches: the EViT branch [14] and a MIL branch. Both branches take a dermoscopy image as input, represented as $X \in \mathbb{R}^{H \times W \times C}$, with dimensions set to $H = W = 224$ and $C = 3$. These branches are trained separately, and this separation continues during inference, allowing an unbiased comparison of their outputs. This approach also helps to evaluate whether the more relevant patches in the EViT's output correspond to the *key patches* learned by the

proposed MIL framework.

The EViT branch closely follows the architecture proposed by Liang *et al.* [14]. It consists of three primary blocks. First, a patch extractor block processes the input image using a procedure similar to that described in section 2.1.1. Next, the EViT encoder is applied to the sequence of patch embeddings. Finally, the state of the CLS token at the last encoder block is fed into a FC Layer to perform the classification task.

The MIL branch comprises two key blocks. First, the dermoscopy image is passed to a pre-trained feature extractor, denoted E. This feature extractor is responsible for generating a 14×14 feature map, where each entry represents a patch embedding. The collection of patch embeddings forms a bag, which is then processed by a MIL classifier framework.

In the following sections, we will delve deeper into the inner workings of each branch, with a particular focus on presenting a novel formulation for the MIL framework that is applicable to both binary and multi-class problems.

3.1 The EViT Branch

As mentioned above, the EViT branch comprises three main blocks. In this section, we will delve into each of these blocks in more detail. In Figure 3.1 a high-level overview of the EViT branch is presented.

The first block corresponds to the *Patch Extractor* block, which is responsible for converting the input dermoscopy image into a sequence of patch embeddings, more specifically: $X \in \mathbb{R}^{H \times W \times C} \rightarrow X \in \mathbb{R}^{N \times D}$. In our implementation, the patch size, P, is equal to 16. Since the input images have a resolution of $(224, 224)$, this translates to a total of $N = 196$ patches. This block uses a convolutional layer with a $(P \times P)$ kernel and a corresponding P-stride to separate the different patches in the image. The convolutional layer is defined with C input channels, aligning with the image's Red,Green and Blue (RGB) composition. The output channels of the convolutional layer are defined by the embedding dimension, denoted by D. This results in a 14×14 feature map, with each entry corresponding to a distinct patch within the image. A flattening operation shapes the feature map into the sequence of patch embeddings, denoted by $X \in \mathbb{R}^{N \times D}$. To this sequence of patch embeddings, a CLS token is appended, and positional embeddings are incorporated, following the description provided in section 2.1.1. This block serves as a necessary step to effectively segment the dermoscopy image into a sequence of patch embeddings, which corresponds to the input of the second block, the EViT encoder.

The EViT encoder block closely follows the structure proposed by Dosovitskiy *et al.* [1], with the addition of a token reorganization block preceding the MLP block. This token reorganization block is crucial in distinguishing between attentive and inattentive patches. Attentive patches are propagated through the network, while inattentive patches are merged into a concise representation. Similar to the

ViT model, the state of the CLS token at the end of the last EViT encoder block serves as input to a classifier that produces the classification for the entire image. For a more complete understanding of the EViT architecture, refer to sections 2.1 and 2.2.

Finally, a FC Layer is used to derive a classification for the entire image. The state of the CLS token at the end of the last EViT encoder block can be represented as $\mathbf{x_{class}} \in \mathbb{R}^D$. In this context, the FC Layer is defined as $h(\mathbf{x_{class}}) = \sigma(\mathbf{w}^\top \mathbf{x_{class}} + \mathbf{b})$, where $\mathbf{w} \in \mathbb{R}^{D \times k}$, $\mathbf{b} \in \mathbb{R}^k$, and σ denotes the *softmax* function. Here, k represents the number of classes. This operation gives us the probability distribution over the k classes.

3.2 The MIL Branch

As mentioned above, the MIL branch comprises two fundamental components: a deep patch extractor and a MIL classifier. In the following sections, we will explore the constituent blocks of the MIL branch in more detail. Our approach introduces a two-step method for generalizing binary MIL classifiers and extends it to a three-step method for multi-class classification. For clarity, we discuss binary and multi-class scenarios separately, and provide insights into the two deep MIL approaches: the *instance-level* and *embedding-level* methods. We also provide clear descriptions of the MIL pooling operators used in these approaches.

3.2.1 Deep Patch Extractor

The first component of the MIL branch is a deep patch extractor denoted as E, responsible for generating a 14×14 feature map. We employ this deep patch extractor to address the challenges posed by dermoscopy images' relatively small size, especially when compared to the WSIs typically used in many MIL-based studies. Given the inherent limitations of dermoscopy images, extracting high-resolution patches individually proves to be a very challenging task. The choice of a 14×14 configuration solves this problem, as each element within this map directly corresponds to a specific region in the input image, a property attributed to the receptive field of CNN-based architectures [51]. The 14×14 feature map is strategically chosen to serve as a "bridge" between the Transformer branch and the MIL branch. This choice derives from EViT's method of dividing the input image into $(16, 16)$ patches, a process adapted to images with dimensions of $(224, 224)$. This results in a total of $N = 196$ patches, which can be efficiently transformed into a 14×14 feature map.

Figure 3.2 provides an overview of the functions performed by the patch extraction block. In this scheme, the dermoscopy image, represented as $X \in \mathbb{R}^{H \times W \times C}$, is processed by a pre-trained patch extractor, denoted as E. The extractor produces a 14×14 feature map, which can be expressed as $E : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{14 \times 14 \times D}$. The 14×14 output is transformed into a collection of patch embeddings,

denoted as $X \in \mathbb{R}^{N \times D}$, where, in our particular scenario, $N = 196$ and D denotes the embedding dimension.

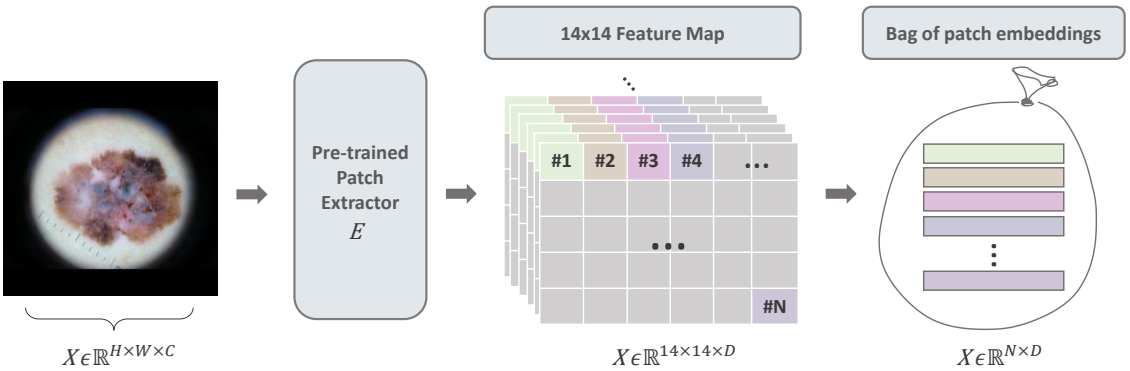

Figure 3.2: Transformation of a dermoscopy image into a bag of patch embeddings by a pre-trained feature extractor, denoted E. The pre-trained patch extractor E processes the input dermoscopy image, resulting in a 14×14 feature map, more precisely: $E : \mathbb{R}^{H \times W \times C} \rightarrow \mathbb{R}^{14 \times 14 \times D}$. Next, this 14×14 feature map is reshaped, resulting in a bag of patch embeddings denoted as $X \in \mathbb{R}^{N \times D}$. In this particular case, $N = 196$ patches are generated.

The deep patch extractor block, plays a pivotal role within the MIL branch pipeline, as it transforms the input image into a bag of patch embeddings. This bag of patch embeddings serves as the input to both the *instance-level* and *embedding-level* MIL approaches, in both binary and multi-class scenarios.

3.2.2 MIL Classifier: Binary Formulation

The second component of the MIL branch is the MIL classifier. In section 2.4.2, we provide a detailed description of the three-step method proposed by Ilse *et al.* [34], which forms the basis of many MIL classifiers. However, this three-step approach can sometimes create ambiguity in defining the boundary between the patch extractor, E, and the function f. To address this ambiguity, we propose a two-step framework. Our two-step approach is validated by two different works. These works confirm the feasibility of decomposing any permutation-invariant function, a property shared by MIL classifiers, into two essential parts [24, 52]. This deconstruction is closely related to our own approach.

The proposed framework operates on a bag of embedded instances, denoted as $X \in \mathbb{R}^{N \times D}$, where N signifies the total number of instances within the bag, and D denotes the embedding dimension. Our two-step approach is characterized by two key functions: $\{h, \phi\}$, where:

- h represents a non-linear classifier.
- ϕ represents a permutation-invariant aggregation function.

The order in which these functions are applied, $\{h, \phi\}$, defines the specific deep MIL approach, i.e.,

the *instance-level* or the *embedding-level* approach. In the following sections, we will elaborate on the formulation of both *instance-level* and *embedding-level* approaches using this two-step method.

3.2.2.A *Instance-level* Approach

The *instance-level* MIL classifier is defined by the proposed two-step method as follows:

i. The function h is designed as an *instance-level* classifier. We define $h(\mathbf{x_n})$ as a FC Layer with a *sigmoid* activation function: $h(\mathbf{x_n}) = \sigma(\mathbf{w}^\top \mathbf{x_n} + b)$, where $\mathbf{w} \in \mathbb{R}^D$, $b \in \mathbb{R}$, and $\sigma(\cdot)$ represents the *sigmoid* function. In this context, $\mathbf{x_n} \in \mathbb{R}^D$ ($\forall_{n=1,\dots,N}$) represents a patch embedding. The function $h(\mathbf{x_n})$ provides the probability of the positive class (i.e., the melanoma class) for each patch, expressed as $h : \mathbb{R}^D \to [0, 1]$.

ii. The function ϕ combines all the individual patch melanoma probabilities, $h(\mathbf{x_n})$ $\forall_{n=1,\dots,N}$, into a bag probability for the melanoma class, denoted as $\theta(X) = \phi(X) \in [0, 1]$. To formalize this operation, we introduce the concept of a space $\mathcal{H} \in [0, 1]$, and thus, $\phi : \mathcal{H}^N \to \mathcal{H}$. The MIL pooling function must exhibit permutation invariance, such as the maximum operator (2.12), the average operator (2.13), or the top-k average operator.

Figure 3.3 illustrates the workflow of the *instance-level* MIL classifier framework using the proposed two-step approach.

3.2.2.B *Embedding-level* Approach

The *embedding-level* MIL classifier is defined by the proposed two-step method as follows:

i. The MIL pooling function, denoted as $\phi : \mathbb{R}^{N \times D} \to \mathbb{R}^D$, aggregates individual patch embeddings, $\mathbf{x_n}$ $\forall_{n=1,\dots,N}$, into a bag representation, $\phi(X) \in \mathbb{R}^D$. It is crucial that ϕ is permutation-invariant, and options include the column-wise global max pooling operator (2.14), the column-wise global average pooling operator (2.15), or the column-wise global top-k average pooling operator.

ii. The function h serves as a *bag-level* classifier. We define $h(\phi(X))$ as an FC Layer with a *sigmoid* activation function: $h(\mathbf{x_n}) = \sigma(\mathbf{w}^\top \phi(X) + b)$, where $\mathbf{w} \in \mathbb{R}^D$, $b \in \mathbb{R}$, and $\sigma(\cdot)$ represents the *sigmoid* function. In this context, $\phi(X) \in \mathbb{R}^D$ represents the output of the MIL pooling function, which serves as the bag's embedding representation. $h(\phi(X))$ computes the probability of classifying the dermoscopy image as melanoma, denoted as $\theta(X) \in [0, 1]$, with $h : \mathbb{R}^D \to \mathcal{H}$, where $\mathcal{H} \in [0, 1]$.

Figure 3.3 illustrates the workflow of the *embedding-level* MIL classifier framework using the proposed two-step method.

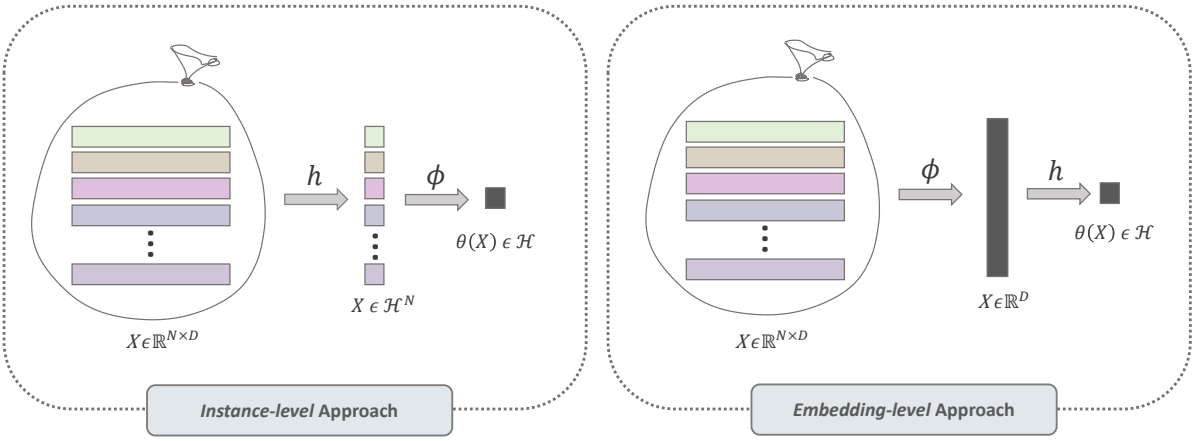

Figure 3.3: MIL classifier approaches. On the left, the figure presents the *instance-level* MIL classifier framework, while on the right, it illustrates the *embedding-level* MIL classifier framework. It is important to note that the order of the functions $\{h, \phi\}$ defines the specific deep MIL approach used.

3.2.3 MIL Classifier: Multi-class Formulation

As previously mentioned, there is a notable gap in the literature when it comes to formulating multi-class MIL solutions. This limitation was the driving force behind our quest to address the multi-class challenge within the MIL framework.

Similar to the two-step approach, the proposed three-step method operates on a bag of embedded instances denoted as $X \in \mathbb{R}^{N \times D}$, where N is the total number of instances in the bag and D is the embedding dimension. The three-step method has three key functions: $\{z, \phi, \sigma\}$, each with a different role:

- z is a linear layer that maps the input from the embedding dimension D to the k-th dimension, where k represents the number of classes, more specifically $z : \mathbb{R}^D \rightarrow \mathbb{R}^k$.
- ϕ is a permutation-invariant aggregation function.
- σ is the *softmax* function.

The order of these three functions: $\{z, \phi, \sigma\}$, determines the specific deep MIL approach used: *instance-level* or *embedding-level*.

It should be emphasized that the binary formulation represents a specific case within the broader multi-class approach. In binary scenarios, the functions z and σ operate sequentially, forming the h function of the two-step method. In particular, the σ function is represented by a *sigmoid* activation function rather than the *softmax* used in multi-class classification. Consequently, the proposed two-step method can be seen as a specific case within the more comprehensive three-step method used

in multi-class problems. In this three-step method, the separation of the affine function z from the non-linear function σ opens the way for the creation of two different *instance-level* methods in the multi-class scenario. These approaches will be elaborated in the following sections. These sections will describe in detail both the *instance-level* and *embedding-level* approaches using the proposed three-step method.

3.2.3.A *Instance-level* Approach

As previously mentioned, the *instance-level* approach can be implemented in two different modes using the proposed three-step method. In this section, we will explain both implementations, providing insights into the underlying reasoning for each.

With the proposed three-step method for the multi-class MIL framework, we derive the **first** *instance-level* implementation as follows:

i. The function z is a linear layer, expressed as $z(\mathbf{x_n}) = \mathbf{w}^\top \mathbf{x_n} + \mathbf{b}$. Here, $\mathbf{w} \in \mathbb{R}^{D \times k}$ and $\mathbf{b} \in \mathbb{R}^k$. In this context, k represents the number of classes, and $\mathbf{x_n} \in \mathbb{R}^D$ ($\forall_{n=1,\dots,N}$) denotes a patch embedding. The function $z(\phi(X))$ independently maps each patch embedding linearly onto a k-dimensional subspace, namely $z : \mathbb{R}^D \to \mathbb{R}^k$.

ii. The *softmax* function, denoted as σ, performs a non-linear transformation on each $z(\mathbf{x_n})$ ($\forall_{n=1,\dots,N}$), converting real-numbered vectors into probability distributions. Specifically, $\sigma : \mathbb{R}^k \to \mathcal{H}^k$, where $\mathcal{H} \in [0, 1]$. Thus, instead of patch embeddings, we obtain a probability distribution for each patch.

iii. The MIL pooling function, $\phi : \mathcal{H}^{N \times k} \to \mathcal{H}^k$, aggregates the individual patch probability distributions into a bag-level *softmax* probability distribution, denoted as $\theta(X) = \phi(X) \in \mathcal{H}^k$. The MIL pooling function ϕ must be permutation-invariant. We can use operators such as the maximum operator, which selects the patch with the highest probability across all classes; the column-wise global average operator; or a top-k average operator, which selects the patches with the highest probabilities across all classes and then performs the average operation per class.

In this first implementation, we compute a probability distribution for each patch. Through the MIL pooling function, we then derive a bag-level *softmax* probability distribution. This method is inspired by the original MIL formulation, where a bag is considered positive if it contains at least one positive instance. For example, when using the maximum pooling operator, we select the patch with the highest probability across all classes to represent the entire image. Consequently, the image's label corresponds to the label of that particular patch.

When defining the MIL pooling operators for the multi-class scenario, it is essential to guarantee that the resulting bag probability distribution sums to 1. This requirement is satisfied by the maximum pooling operator, which selects the patch with the highest probability distribution across all classes. Likewise, the column-wise global average pooling operator also satisfies this constraint. For the mathematical

proof that this average operator also satisfies this constraint, see section A.1 in appendix A. The top-k operator, which is a "filtered" version of the average operator, maintains the probability distribution constraint by using only certain patch representations for the averaging operation.

As previously mentioned, using the proposed three-step method for a multi-class MIL framework, we can derive a **second** *instance-level* implementation:

i. The function z is designed as a linear layer: $z(\mathbf{x_n}) = \mathbf{w}^\top \mathbf{x_n} + \mathbf{b}$, where $\mathbf{w} \in \mathbb{R}^{D \times k}$, and $\mathbf{b} \in \mathbb{R}^k$. Here, k represents the number of classes, and $\mathbf{x_n} \in \mathbb{R}^D$ ($\forall_{n=1,\dots,N}$) represents a patch embedding. In this case, the function $z(\phi(X))$ linearly maps each patch embedding independently to a k-dimensional subspace, denoted as $z : \mathbb{R}^D \to \mathbb{R}^k$.

ii. The MIL pooling function, $\phi : \mathbb{R}^{N \times k} \to \mathbb{R}^k$, aggregates individual patch representations into a bag-level representation, $\phi(X) \in \mathbb{R}^k$. The MIL pooling function, ϕ, must be permutation-invariant, e.g., the column-wise global max pooling operator, the column-wise global average pooling operator, or the column-wise global top-k average pooling operator.

iii. The *softmax* function, denoted σ, performs a non-linear transformation on the bag-level representation $\phi(X)$. This function transforms a vector of real numbers into a probability distribution, represented as $\sigma : \mathbb{R}^k \to \mathcal{H}^k$, with $\mathcal{H} \in [0, 1]$.

In this second approach, the MIL pooling function is applied before the *softmax* step, ensuring that the resulting bag probability distribution sums to 1. For example, in the case of the maximum operator, the resulting bag probability distribution corresponds to the *softmax* of the highest logits for each respective class. The other MIL operations follow a similar logic, with their respective operators.

3.2.3.B *Embedding-level* Approach

The *embedding-level* MIL multi-class classifier is defined by the proposed three-step method as follows:

i. The MIL pooling function, denoted as $\phi : \mathbb{R}^{N \times D} \to \mathbb{R}^D$, aggregates all individual patch embeddings, represented as $\mathbf{x_n}$ $\forall_{n=1,\dots,N}$, into a bag-level representation, $\phi(X) \in \mathbb{R}^D$. The MIL pooling function ϕ must be permutation-invariant, and common choices include the column-wise global max pooling operator, the column-wise global average pooling operator, or the column-wise global top-k average pooling operator.

ii. The function z is designed as a linear layer, specifically: $z(\mathbf{x_n}) = \mathbf{w}^\top \phi(X) + \mathbf{b}$, where $\mathbf{w} \in \mathbb{R}^{D \times k}$, $\mathbf{b} \in \mathbb{R}^k$. In this context, k denotes the number of classes, and $\phi(X) \in \mathbb{R}^D$ represents the output of the MIL pooling function, which serves as the embedded representation of the bag. Here, the function $z(\phi(X))$ linearly maps the embedded bag representation onto a k-dimensional subspace, notably: $z : \mathbb{R}^D \to \mathbb{R}^k$.

iii. The *softmax* function, denoted as σ, operates as a non-linear transformation on $z(\phi(X))$, transforming the vector of real numbers into a probability distribution. More specifically, $\sigma : \mathbb{R}^k \rightarrow \mathcal{H}^k$, where $\mathcal{H} \in [0, 1]$.

In the next chapter, we will delve into the experimental setup, where we will present the datasets used for both training and testing, along with a detailed overview of the model configurations we used, among other important experimental details.

4

Experimental Set-Up

Contents

4.1 The Datasets

To conduct this thesis research, we used several publicly available dermoscopy image datasets. The datasets used in this thesis are the ISIC 2019 [3–5], the PH^2 [45], and the Derm7pt [53]. The ISIC 2019 dataset was used for training and validation, while the other two were used exclusively for testing. This approach allowed us to assess the ability of the model to generalize across different hospital domains and distribution patterns, addressing a gap present in the works discussed in section 2.3.

4.1.1 Training and Validation Datasets

The original ISIC 2019 training dataset [3–5] consists of $25,331$ dermoscopic images categorized into 8 diagnostic classes: Actinic keratosis (AK), Basal cell carcinoma (BCC), Benign keratosis (BKL), Dermatofibroma (DF), Melanoma (MEL), Melanocytic nevus (NV), Squamous cell carcinoma (SCC), and Vascular lesion (VASC). The corresponding test set contains $8,238$ images. However, it lacks publicly available labels and was therefore not used in this thesis. A summary of the original ISIC 2019 dataset [3–5] is shown in table 4.1.

Table 4.1: Number of training and test samples of the original ISIC 2019 dataset [3–5]. This dataset comprises a total of 8 classes: AK, BCC, BKL, DF, MEL, NV, SCC, and VASC.

ISIC 2019 Dataset	AK	BCC	BKL	DF	MEL	NV	SCC	VASC	**Total**
Train	867	3323	2624	2624	4522	12875	628	253	25331
Test	—	—	—	—	—	—	—	—	8238

To prepare the original ISIC 2019 training dataset for our dissertation experiments, we performed initial data processing steps. First, we ensured data cleanliness by removing duplicate images, resulting in a reduction from $25,331$ to $25,294$ images. Additionally, we split the training set into separate training and validation sets, resulting in $20,228$ images for training and $5,066$ images for validation. This split is essential for training and validating our models.

Our research includes two classification problems: binary and multi-class classification. The binary task focuses on classifying samples as either MEL or NV, while the multi-class task includes all eight classes. Both the training and validation datasets generated are unbalanced for both the binary and multi-class scenarios. Table 4.2 provides a detailed analysis of the class distributions within the training and validation sets.

4.1.2 Test Datasets

Two datasets were used for testing in this study. First, we used the PH^2 dataset [45], which consists of a modest 200 dermoscopy images. Among these, 40 images represent melanoma cases (MEL), while

the remaining 160 represent nevus cases (NV). Additionally, we included the Derm7pt dataset [53] for testing purposes. These two test sets are only used in the binary classification problem, which focuses only on the MEL and NV classes for testing.

Table 4.2 provides a comprehensive view of the class distributions of the training and validation datasets. It also provides an overview of the number of samples in the test sets, which come from both the PH^2 [45] and Derm7pt [53] datasets.

Table 4.2: Summary of the overall distribution of the training, validation, and testing datasets. This summary includes the number of samples in the training and validation sets from the original ISIC 2019 dataset [3–5], along with the number of samples for the test sets from the PH^2 [45] and Derm7pt [53] datasets.

Classes	**ISIC 2019**		$\mathbf{PH^2}$	**Derm7pt**
	Train	Validation	Test	Test
AK	687	173	—	—
BCC	2653	664	—	—
BKL	2089	525	—	—
DF	191	48	—	—
MEL	3611	904	40	252
NV	10293	2575	160	575
SCC	502	126	—	—
VASC	202	51	—	—
Total	20228	5066	200	827

4.2 Dermoscopic Image and Mask Pre-processing

In this section, we address the pre-processing of dermoscopy images. To facilitate the effective use of pre-trained models in both training and inference, dermatoscopy images require dimensional standardization. To achieve this, we resized all input images to a common size of $224 \times 224 \times 3$. During resizing, we preserved the original aspect ratio by first converting it into a square format, which was achieved via padding. Only then was the image reduced to the desired dimensions. For both the EViT and MIL branches, we divided the images into patches with a resolution of 16×16, resulting in a total of $N = 196$ patches.

In specific experiments, we integrated binary segmentation masks into the MIL framework. This was done to introduce some domain knowledge into the framework and to narrow the focus of the identification of *key patches* to the lesion area only. This prevented the framework from identifying key patches in undesirable regions of the image, such as corners, edges, air bubbles, hair, or rulers. These masks use binary values, where “1” represents relevant pixels and “0” represents non-relevant pixels.

In the process of creating binary masks, we encountered two scenarios: one where we used segmentation algorithms and another where manual binary masks were publicly available. For the ISIC 2019 dataset, some images already had segmentation masks from the ISIC 2018 dataset [3, 4]. For the

rest, we used an automatic segmentation model [54, 55]. It is important to note that this segmentation did not always produce perfect "0" and "1" masks. Sometimes pixel values fell between "0" and "1", especially near lesion borders. To address this, we designated values below 0.5 as "0" and those above 0.5 as "1". This method resulted in binary masks for all training and validation images. For binary classification, approximately 14% of the masks came from the ISIC 2018 dataset, while the remaining 86% were generated by the segmentation algorithm. In the case of the test sets, the PH^2 test dataset [45] had publicly available manual segmentation masks. For the Derm7pt test set [53], we had to generate binary masks by applying a segmentation algorithm [54, 55]. We did not use binary masks in the context of the multi-class problem, as explained in section 4.4.2.

Figure 4.1 visually summarizes the process of obtaining the $224 \times 224 \times 3$ input images and binary masks from the original dermoscopy images.

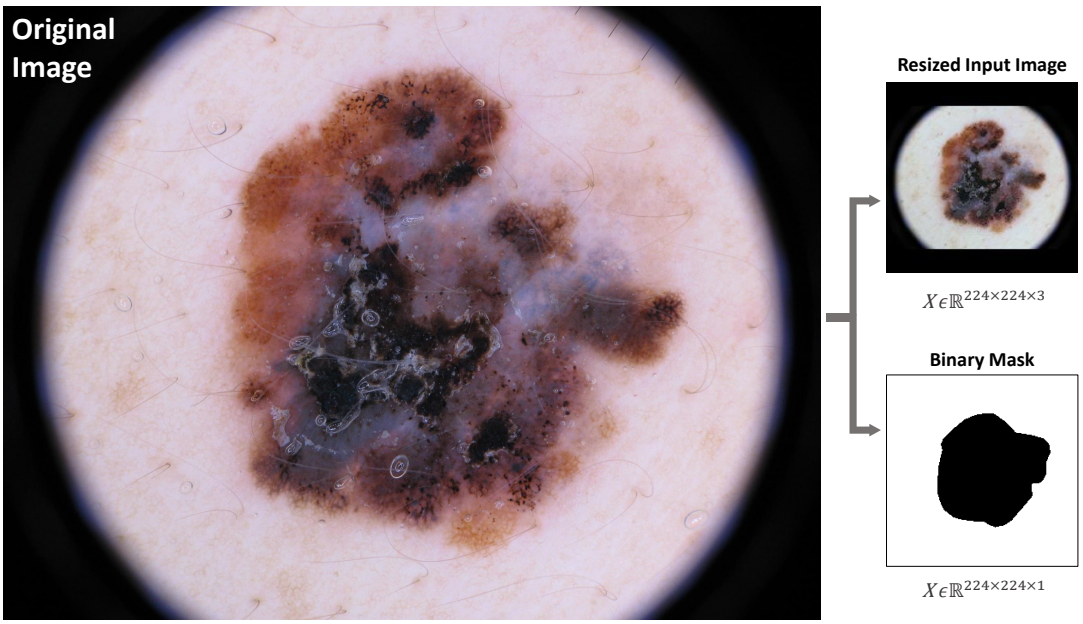

Figure 4.1: An illustration of a training image from the ISIC 2019 dataset [3–5] assigned to the class MEL. On the left, the dermoscopy image is shown in its original size. The resized model input is shown on the upper right, while the corresponding binary mask is shown on the lower right. The resized input and binary mask retain the original proportions and are set to dimensions of $224 \times 224 \times 3$.

4.3 Evaluation Metrics

In this section, we discuss the evaluation metrics used to assess the performance of different approaches on different architectures. These metrics, including Balanced Accuracy (BA) and recall, are derived from the confusion matrix and play a crucial role in measuring the effectiveness and accuracy of our methods.

4.3.1 Confusion Matrix

The confusion matrix helps us evaluate the performance of a model. When dealing with k classes, the confusion matrix takes the form of a square $k \times k$ matrix. Each cell at position (i, j) tells us how many samples were classified as class j when they really belonged to class i. More specifically, the rows in a confusion matrix represent the true classes, and the columns represent the predicted classes. By analyzing the confusion matrix for class i, we can compute metrics such as True Positives (TP), False Positives (FP), True Negative (TN), and False Negatives (FN). Let us consider the case where class i is considered the positive class, while the other classes are considered negative. In this case, TP_i stands for correctly identified samples that belong to class i. FP_i includes samples that do not belong to class i but are mislabeled as class i. TN_i includes samples that don't belong to class i but are correctly labeled as negative. Finally, FN_i represents instances that really belong to class i but are misclassified as negative.

4.3.2 Balanced Accuracy and Recall

As previously mentioned, the datasets used in the experiments in this thesis exhibit imbalances in both the binary and multi-class contexts. Therefore, evaluating the proposed models using the Accuracy (Acc) metric may lead to misleading results and conclusions. To address this issue, we chose to evaluate all the models tested using the recall metric for each class i $(\forall_{i=0,\ldots,(k-1)})$. Here, k denotes the total number of classes, with $k = 2$ for the binary problem and $k = 8$ for the multi-class problem. For each class i, the recall R_i is computed as follows:

$$R_i = \frac{\mathsf{TP}_i}{\mathsf{TP}_i + \mathsf{FN}_i}. \tag{4.1}$$

The Balanced Accuracy (BA) corresponds to the average of the recall values across all k classes and is defined as follows:

$$\mathsf{BA} = \frac{1}{k} \sum_{i=0}^{k-1} R_i. \tag{4.2}$$

4.4 Model Configurations and General Set-up

In this section, we present the model configurations used for each branch of the framework proposed in chapter 3. In addition, we discuss the configurations of the baseline models used for comparison with our approach. The experimental setup was intentionally consistent across all models to ensure fair comparisons. We will begin by discussing the general methods applied to all models in the EViT branch, the MIL branch, and the baseline models. Specific model details will follow.

Since Transformer-based architectures involve a large number of learnable parameters, training from scratch would be time-consuming and potentially lead to sub-optimal generalization. Hence, to maintain

fairness in model comparisons, we initialized all architectures with weights pre-trained on the ImageNet-1k dataset [56].

Given the need for large amounts of data for DL models to effectively generalize [57], we employed data augmentation techniques. Inspired by the Data-efficient Image Transformer (DEiT) model [57], we incorporated random augmentation techniques [58], repeated augmentation [59], and random erasing [60]. Additionally, we implemented an early stopping technique to prevent overfitting on the training data. Training is stopped if the validation loss does not decrease over a period of 15 epochs.

Given the unbalanced nature of the datasets used in the experiments of this thesis, we employed a class-weighting technique for all models. These models were trained using the categorical Cross Entropy (CE) loss function. In the multi-class scenario, the CE loss function for one training example is defined as follows:

$$\mathcal{L}_{CE} = \sum_{i=0}^{k} -w_i y_i \log\left(p_i\right), \tag{4.3}$$

Where k represents the total number of classes, p_i is the probability of the i-th class defined by the *softmax* function, and w_i is the class weight, computed as:

$$w_i = \frac{\text{Total number of samples}}{k \times (\text{number of samples of the } i\text{-th class})}, \quad \forall_{i=0,\ldots,k}. \tag{4.4}$$

In the binary classification problem of melanoma (MEL) versus nevus (NV), we use the binary CE loss function and only consider the class weights of the positive class, i.e., the melanoma (MEL) class. Table B.1, located in Section B.1.1 in Appendix B, provides an overview of the different configurations used for each of the architectures used in this thesis.

All the experiments in this thesis work were performed using a computer equipped with the high-performance NVIDIA GeForce RTX 3090 graphics card paired with 24 GB of GDDR6X RAM memory. All models were implemented using the Python programming language, specifically using deep learning Python libraries such as Pytorch [61], timm [62], and Scikit-learn [63].

It is crucial to note that the configurations for each branch were determined based on performance in the binary melanoma (MEL) versus nevus (NV) problem on the validation set of the ISIC 2019 dataset [3–5].

4.4.1 EViT Branch Configuration

The architecture of the EViT model is governed by four critical hyper-parameters that require pre-configuration. The EViT architecture includes several models, such as the EViT-S with approximately 22 million parameters and the EViT-B with nearly 86 million learnable parameters. Therefore, it is crucial to specify the particular EViT model that will serve as the basis for our experiments. In addition,

we must decide how to handle inattentive tokens within EViT, whether to aggregate them into a fused representation or to discard them altogether. Implementation-wise, the distinction between attentive and inattentive tokens depends on two crucial settings: Keep rate (Kr), which determines the token survival rate (i.e., how many attentive tokens are left), and the explicit specification of layers, known as EViT encoder blocks, where token reorganization takes place. For all the experiments performed, we decided to apply the token reorganization block in three layers: the 3rd, the 6th, and the 9th layers. We decided to keep this configuration because it matches the default settings of the original EViT model [14]. In this section, we will provide a clear overview of the EViT configurations used in the experiments conducted in chapter 5.

The EViT models tested include the EViT-S and EViT-B variants. Table 4.3 shows a comparison between the configuration of these two models. In essence, the EViT-B has nearly four times the number of parameters compared to the EViT-S. However, preliminary experiments indicate that both models show similar performance on the validation set of the ISIC 2019 dataset [3–5]. Our experiments included both models, with Kr $= 0.6$ and inattentive tokens discarded at the 3rd, 6th, and 9th layers, without introducing fused tokens. The EViT-S obtained a balanced accuracy (BA) of 91.4%, while the EViT-B obtained 91.5%. Consequently, for the subsequent experiments conducted using the EViT architecture, we chose to proceed with the EViT-S due to its comparable performance and significantly fewer parameters.

Table 4.3: Comparison between EViT-S and EViT-B model configurations.

EViT Models	#Layers	#SA Heads	Embedding Dim. (D)	#Params.
EViT-S	12	6	384	22.1M
EViT-B	12	12	768	85.1M

In the original EViT architecture, inattentive tokens are combined into a fused representation. This fused token is then forwarded through the network. The rationale behind this approach is that even though inattentive tokens may be less significant than attentive ones, they still hold valuable information that could contribute to classification. Figure 4.2 presents a bar plot that allows for a comparison of different EViT-S configurations with and without the fused token. For this experiment, we considered six different Kr configurations: Kr $= \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The token reorganization block was applied in the 3rd, 6th, and 9th layers.

Upon thorough examination of the bar plot depicted in Figure 4.2, it becomes evident that the performance of various EViT-S configurations shows minimal variance. In fact, across different Kr values, the BA on the ISIC 2019 validation set [3–5] only varies between 90% and 92%. Overall, while the use of a fused token can improve the performance of the EViT-S model, the difference is relatively small. Furthermore, since the MIL framework does not accommodate the concept of "fused embedding", we have chosen to use the EViT-S configuration **without** the fused token in our subsequent experiments.

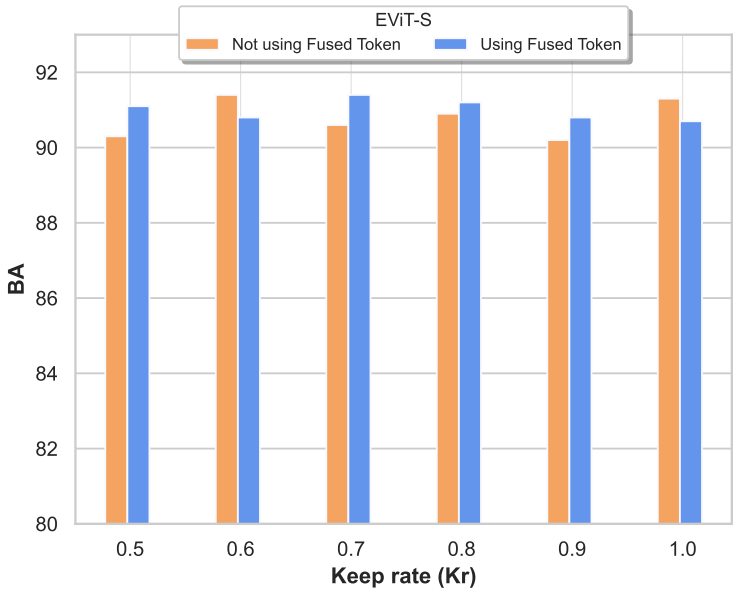

Figure 4.2: Comparison between different EViT-S configurations with and without the use of the fused token. These results were obtained through an evaluation on the ISIC 2019 validation set [3–5], specifically addressing the binary classification task of melanoma (MEL) versus nevus (NV). The x-axis represents the different Krs considered, and the y-axis represents the BA. For this experiment, we considered six different Kr configurations: Kr $= \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The layers where inattentive tokens are discarded correspond to the 3rd, 6th, and 9th layers.

The combination of layers in which token reorganization takes place and the choice of Kr are the most critical hyper-parameters in the EViT architecture. Since we decided to fix the token reorganization block in the 3rd, 6th, and 9th layers, an extensive search for the best Kr configurations in the EViT-S model was required. For a better understanding of the importance of the Kr hyper-parameter in the EViT model, please refer to Figure 4.3 and table 4.4. Figure 4.3, illustrates the process of eliminating inattentive tokens across the three layers (3rd, 6th, and 9th) using Kr $= 0.7$. The table 4.4 shows the number of preserved attentive patches that were not discarded by the EViT model over the different Krs.

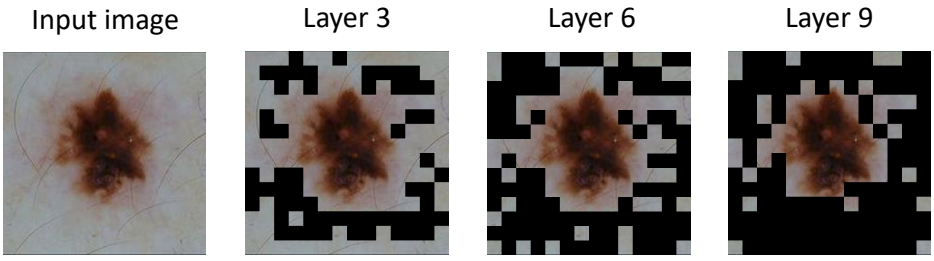

Figure 4.3: Visual representation of the removal of inattentive patches in the 3rd, 6th, and 9th layers with a Kr $= 0.7$. In the final layer, only 68 attentive patches remain. Note that the inattentive patches that were discarded correspond to healthy parts of the skin.

Table 4.4: EViT's number of attentive tokens left for each respective Kr, accordingly with the configuration of three removal layers: 3rd, 6th, and 9th layers.

Kr	0.5	0.6	0.7	0.8	0.9	1.0 (DEiT)
#Attentive patches	25	43	68	101	144	196 (N)

Since the Kr hyper-parameter plays a crucial role within the EViT architecture, we conducted a series of experiments to examine its impact on the EViT-S configuration. Figure 4.4 shows the BA results across different Kr values for the ISIC 2019 [3–5] validation set and both test datasets: PH^2 [45] and Derm7pt [53]. These results indicate that a higher Kr does not necessarily lead to better performance, especially on the test datasets. It is clear that Kr values of 0.6, 0.7, and 0.8 outperform the rest. Consequently, in the experiments in chapter 5 we will use the Kr $= 0.6$ and Kr $= 0.7$ EViT-S configurations.

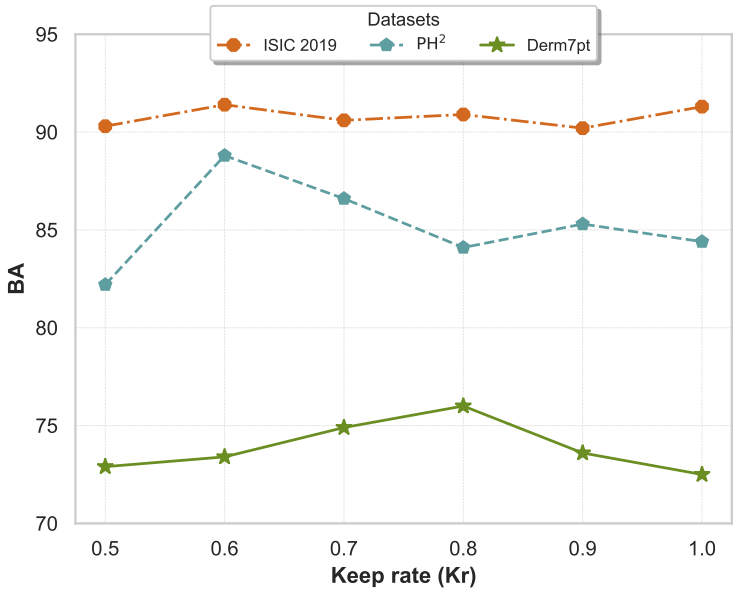

Figure 4.4: Performance comparison between EViT-S models with different Krs. The experiments were conducted on the ISIC 2019 validation dataset [3–5], as well as on the two test datasets: PH^2 [45] and Derm7pt [53], for the binary classification task of melanoma (MEL) versus nevus (NV). The x-axis represents the different Kr values, while the y-axis represents the corresponding BA results.

4.4.2 MIL Branch Configuration

In this section, we will define the configurations for the MIL branch. As with the EViT architecture, determining the optimal configuration for the MIL branch is critical to our experiments.

First, let us address some implementation details that play a crucial role during the training process of the MIL branch. During the first 5 epochs of training, only the classifier is updated, i.e., the backpropagation algorithm only targets the classification head. After this initial phase, we start updating the weights

of the patch extractor, E. This approach can be seen as a "warm-up" strategy for the MIL classifier.

For the pre-trained patch extractors, we experimented with five CNN-based architectures and with four Transformer-based architectures. The CNN-based pre-trained patch extractors used in this thesis consist of the ResNet-18 (RN-18) [64], the ResNet-50 (RN-50) [64], the VGG-16 [65], the DenseNet-169 (DN-169) [66] and the EfficientNet-B3 (EN-B3) [67]. The Transformer-based architectures include the DEiT-S (with and without the CLS token) and the EViT-S (without the CLS token). The EViT-S models are configured with Kr $= 0.7$ and with the default layer distribution for patch removal (i.e., the 3rd, 6th, and 9th layers). One version of EViT-S has a fused token (denoted as EViT-fused), while the other does not (denoted as EViT-S). Note that using Transformer-based architectures as patch extractors in the MIL framework changes the conventional MIL formulation presented in section 2.4. Nevertheless, this exploration may allow us to understand whether ViT models really need a CLS token for classification, or whether using other types of classifiers, in this case the MIL classifier, can achieve competitive results. This exploration may also provide insight into whether it is beneficial for the MIL framework to use instance embeddings that contain correlated and spatial information of all patches in the image.

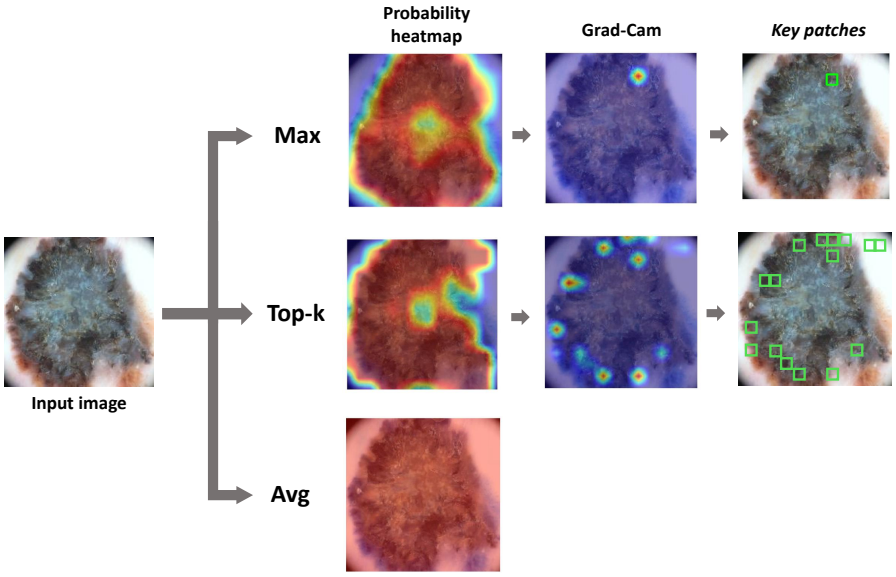

Figure 4.5: MIL visualization pipeline for identifying *key patches* using the three main MIL pooling operators: *maximum* operator, *average* operator, and *top-k average* operator. The pipeline begins by visualizing heatmaps representing the probabilities of each patch belonging to the melanoma (MEL) class. Subsequently, it applies a Grad-Cam visualization technique [50] to highlight the patches that shape the model's output. This process identifies the *key patches* responsible for the classification of the image. In the case of the *top-k average* operator, the k value is set at approximately 12%, which corresponds to 16 patches for this configuration ($N = 196$). It is important to note that the *average pooling* operator assumes that all patches belong to the melanoma class, making it difficult to identify specific *key patches* since they all contribute to the model's output. These visualizations are based on the MIL framework using the RN-18 backbone.

To visualize and better comprehend the “key patches” identified in the melanoma (MEL) versus nevus (NV) problem, we have developed a visualization pipeline. The pipeline involves visualizing the melanoma probability heatmap for each patch in the image and subsequently using a Grad-Cam technique to identify the patches that triggered the bag classification. Essentially, for an image classified as melanoma by the MIL framework, we aim to determine which patches exhibit the highest melanoma probability. Figure 4.5 provides an overview of this process for the three MIL pooling operators studied in the *instance-level* approach: the *maximum* operator, the *average* operator, and the *top-k average* operator. When visualizing the gradients, we adopt an approach similar to that of Chefer *et al.* [50], which differs from the original Grad-Cam implementation [28]. In our approach, we propagate gradients in their raw form, applying only min-max normalization to visually capture the “key patches”.

This pipeline only applies to the *instance-level* approach, as the *embedding-level* approach does not allow for the identification of “key patches”. The *embedding-level* approach primarily serves as a bridge between the MIL framework and traditional CNN architectures. However, we use the original Grad-Cam technique [28] to visualize the most significant regions identified by the *embedding-level* models.

As depicted in Figure 4.5, it is evident that the *average* MIL pooling operator falls short in providing insightful details about the “key patches” in a dermoscopy image. To improve the performance of this operator, we introduced domain knowledge by incorporating binary masks of the images into the MIL framework. Consequently, we devised four additional MIL pooling operators. For the *instance-level* approach, we experimented with a *masked maximum* operator and a *masked average* operator. In the *embedding-level* approach, we explored a *masked column-wise global max* pooling operator and a *masked column-wise global average* pooling operator. Figure 4.6 visually illustrates the operations of the *masked average* operator.

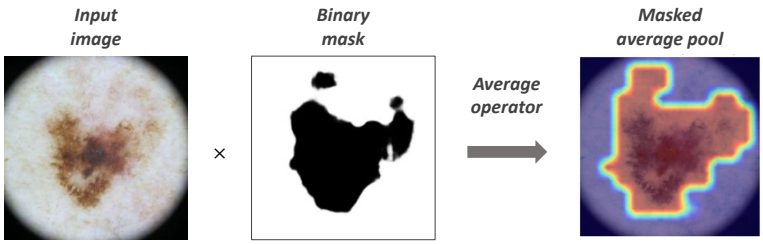

Figure 4.6: Visualization of the process performed by the *instance-level* MIL model using the *masked average pooling* operator on an image from the ISIC 2019 dataset validation set [3–5]. The *masked average pooling* operator multiplies a binary mask with the input image and then applies the *average* operator during both training and inference. Similar reasoning applies to the *masked max* operator and the corresponding masked pooling operators used in the *embedding-level* approach.

To assess the impact of applying binary segmentation masks on the performance of the MIL framework, we performed a comparative analysis of different MIL pooling operators. Specifically, we compared the masked operators with the *max* and *average operators* using the validation set of the ISIC

2019 dataset [3–5]. The experiments were performed with two different backbones, namely RN-50 and EN-B3, within our MIL framework. The results of this performance comparison are summarized in table 4.5.

Table 4.5: Comparison between standard MIL pooling operators and the MIL pooling operators incorporating binary masks. In this experiment, we used the RN-50 and EN-B3 architectures as backbones for the MIL framework. The experiment was conducted on the ISIC 2019 validation dataset [3–5], with respect to the binary melanoma (MEL) versus nevus (NV) task. The results suggest that constraining deep models to specific image regions may not consistently improve performance.

Models			**ISIC 2019**		
Backbone	Approach	Pool.	BA	R-MEL	R-NV
RN-50	Instance	Max	88.1	85.6	90.6
		Avg	89.0	85.3	**92.6**
		Masked max	87.5	**86.2**	88.9
		Masked avg	**89.9**	85.3	92.5
	Embedding	Max	**89.9**	84.7	**95.1**
		Avg	88.9	85.6	92.2
		Masked max	89.3	85.7	92.8
		Masked avg	89.2	**88.1**	90.3
EN-B3	Instance	Max	88.5	86.7	90.2
		Avg	**89.1**	**86.7**	**91.8**
		Masked max	88.3	86.4	90.1
		Masked avg	87.4	85.1	89.8
	Embedding	Max	86.0	85.7	86.4
		Avg	**89.1**	84.4	**93.8**
		Masked max	87.1	83.9	90.3
		Masked avg	89.0	**85.8**	92.2

The results shown in table 4.5 suggest that constraining the model with domain-specific knowledge does not consistently lead to superior performance. Consequently, we decided not to use domain-specific binary masks in the experiments conducted in chapter 5. Instead, we focused on investigating the effect of the k hyper-parameter associated with the *top-k average* pooling operator.

We conducted experiments with three different k values in the *instance-level* approach: approximately 12.5%, 25%, and 50%. In the context of our experiments, which involved input images of $(224, 224)$ resolution and patches of $(16, 16)$ resolution, we dealt with a total of $N = 196$ patches. This implies that for the $k \approx 12.5\%$ configuration, we considered 25 patches; for $k = 25\%$, we worked with 49 patches, and for $k = 50\%$, we used 98 patches. To determine the optimal k value for the *top-k average* operator in the *instance-level* approach, we conducted experiments using various backbones on the validation set of the ISIC 2019 dataset [3–5]. A summary of these experiments is shown in Figure 4.7.

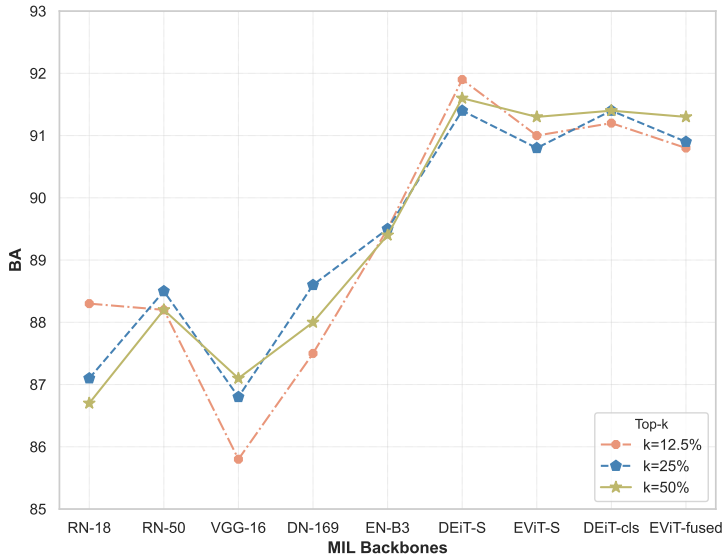

Figure 4.7: Search for the optimal k hyper-parameter in the *instance-level* top-k average MIL pooling operator. We explored three values for the hyper-parameter: $k \approx 12.5\%$, $k = 25\%$, and $k = 50\%$. Our experiments were conducted and evaluated on the validation set of the ISIC 2019 dataset [3–5], employing various MIL backbones. The backbones included RN-18, RN-50, VGG16, DN-169, EN-B3, DEiT-S, DEiT-cls, EViT-S, and EViT-fused. Notably, with EViT backbones, $k \approx 12.5\%$ resulted in only 9 patches, $k = 25\%$ retained 17 patches, and $k = 50\%$ maintained 34 patches. These results indicate that using more patches in the bag evaluation does not necessarily lead to better performance.

The plot in Figure 4.7 shows that the choice of k for the *top-k average* operator in the *instance-level* approach does not significantly impact the performance of the different MIL models. This observation suggests that not all patches within a dermoscopy image contribute equally to the classification task. Interestingly, the $k \approx 12.5\%$ and $k = 25\%$ scenarios consistently yield the highest BA results across different MIL backbones. Based on these results, we selected $k = 25\%$ as the default configuration for the *top-k average* pooling operator. To ensure a fair comparison between the *embedding-level* and *instance-level* approaches, we have also adopted $k = 25\%$ as the preferred setting for the *column-wise global top-k average* operator in the *embedding-level* approach.

Among the diverse backbones used in our MIL framework, we selected the top-performing configurations from both CNN-based and Transformer-based backbones. To make this choice, we conducted comprehensive experiments using all MIL backbones, approaches, and associated pooling operators on the validation set of ISIC 2019 [3–5]. Figure 4.8 presents a bar plot summarizing the results of these experiments.

Our results indicate that, in general, Transformer-based backbones tend to deliver superior performance. It is important to note that using a Transformer-based model as a patch extractor within the MIL framework, as proposed in chapter 3, introduces a paradigm shift in traditional MIL approaches. There-

fore, it would be naive to conclude that Transformer architectures universally outperform CNN models.

Based on the findings in Figure 4.8, we have identified the best performing CNN backbones as RN-50 and EN-B3, and in the Transformer-based category, we have chosen the DEiT-S and EViT-S architectures. In chapter 5, we will refer to the MIL models using these respective patch extractors as MIL-RN-50, MIL-EN-B3, MIL-DEiT-S, and MIL-EViT-S. In particular, the EN-B3 and DEiT-S backbones have demonstrated superior performance, and, as a result, our focus will be primarily on the MIL-EN-B3 and MIL-DEiT-S models.

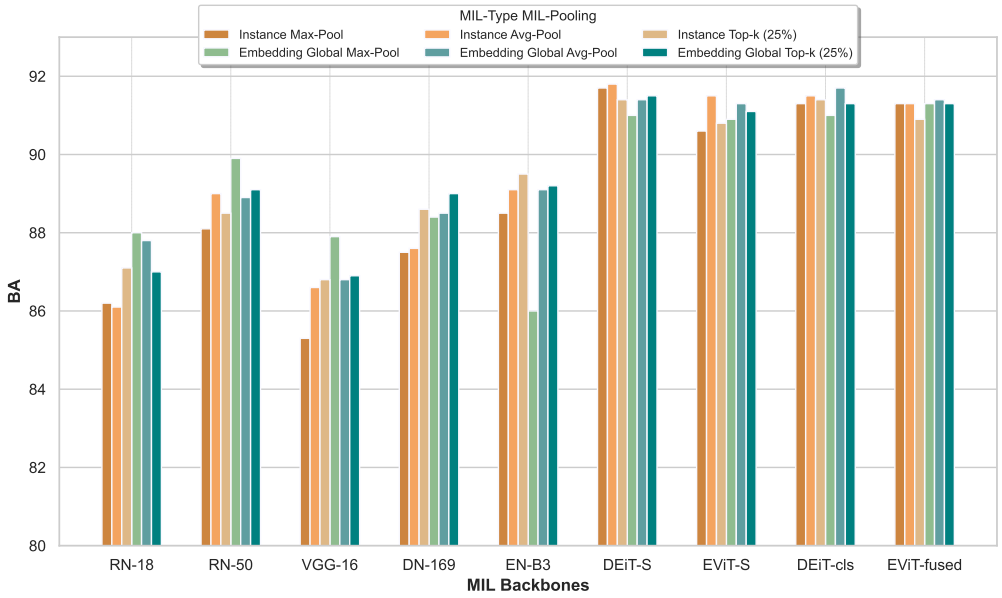

Figure 4.8: Bar plot illustrating the performance of the MIL framework for different patch extractors, referred to as backbones in this plot. Our evaluation involved extensive experimentation with several CNN-based and Transformer-based backbones, as well as different MIL approaches and pooling operators. The backbones experimented with consisted of the RN-18, the RN-50, the VGG-16, the DN-169, the EN-B3, the EViT-S, the EViT-fused, the DEiT-S, and the DEiT-cls. The evaluation was conducted on the ISIC 2019 validation dataset [3–5]. While Transformer-based backbones generally showed superior performance, it is important to acknowledge that the inclusion of Transformer models as patch extractors in the MIL framework represents a significant deviation from the conventional MIL formulation.

4.4.3 Baseline Models

To ensure a fair and comprehensive evaluation of our models, we conducted comparative evaluations involving both the EViT and MIL branches in conjunction with a set of baseline models covering both CNN-based and ViT-based architectures. The CNN-based baseline models include RN-18 [64], RN-50 [64], VGG-16 [65], DN-169 [66], and EN-B3 [67]. In the Transformer-based architectures category, we considered ViT-S and ViT-B [1], and DEiT-S and DEiT-B [57].

To facilitate the experiments described in chapter 5, we carefully selected two representative mod-

els from each of the CNN and Transformer baselines. This selection process involved an extensive evaluation of each model on the validation set of the ISIC 2019 dataset [3–5], focusing on the binary classification problem of melanoma (MEL) versus nevus (NV). The results of this evaluation are summarized in table 4.6.

Table 4.6: Evaluation results of a set of baseline models on the validation set of the ISIC 2019 dataset [3–5]. The baseline models include different architectures, including RN-18, RN-50, VGG-16, DN-169, EN-B3 from the CNN-based category, and ViT-S, ViT-B, DEiT-S, DEiT-B from the Transformer-based category. The evaluation is performed for the binary classification problem of melanoma (MEL) versus nevus (NV).

Baseline models		ISIC 2019		
		BA	R-MEL	R-NV
CNNs	RN-18	88.6	83.8	93.4
	RN-50	88.9	82.6	95.1
	VGG-16	87.7	83.6	91.8
	DN-169	89.1	83.2	95.0
	EN-B3	**90.7**	**85.5**	**95.8**
ViTs	ViT-S	91.3	86.8	95.8
	ViT-B	90.6	85.3	95.8
	DEiT-S	**91.7**	86.7	**96.7**
	DEiT-B	91.7	**87.2**	96.2

Based on the results present in table 4.6, in conjunction with the decisions made in section 4.4.2, we have decided to proceed with RN-50, EN-B3, DEiT-S, and ViT-S as our selected baseline models. Of these models, the EN-B3 and DEiT-S baseline models will receive special attention due to their superior performance. Additional results related to the multi-class problem can be found in section B.1.1.A of appendix B.

5

Experimental Results and Discussion

Contents

In this chapter, we present the results related to the framework proposed in chapter 3. These results are crucial in addressing the two key questions that motivate this thesis:

1. Are all patches within a dermoscopy image equally relevant, or do deep models rely primarily on specific ROIs to perform classification?
2. Does the spatial positioning of patches within the image play a critical role in an accurate diagnosis of skin cancer?

5.1 Binary Problem: Melanoma vs. Nevus

5.1.1 Performance and Generalization Results

Table 5.1 provides a comparative analysis of the results for different model configurations following the settings detailed in Section 4.4. The models evaluated include the best-performing baseline models (EN-B3 and DEiT-S), the two best EViT configurations (Kr $= 0.6$ and Kr $= 0.7$), and the best-performing MIL backbones (MIL-EN-B3 and MIL-DEiT-S). For a more comprehensive overview, please refer to Table B.3 in Section B.2.1 of Appendix B, which contains additional results for the RN-50 and ViT-S baseline models, as well as the MIL-RN-50 and MIL-EViT-S models.

Table 5.1: Results of the best-performing models within the EViT branch, the MIL branch and the baseline models. These models include the best-performing baseline models (EN-B3 and DEiT-S), the two best EViT configurations (Kr $= 0.6$ and Kr $= 0.7$), and the most effective MIL backbones (MIL-EN-B3 and MIL-DEiT-S). Validation was performed on the ISIC2019 validation set [3–5], together with assessments on the two test datasets: PH^2 [45] and Derm7pt [53].

	Models		ISIC 2019			PH^2			Derm7pt		
			BA	R-MEL	R-NV	BA	R-MEL	R-NV	BA	R-MEL	R-NV
Base	EN-B3		90.7	85.5	95.8	88.8	82.5	95.0	76.2	57.9	94.4
	DEiT-S		91.7	86.7	**96.7**	86.6	75.0	98.1	74.0	53.2	94.8
EViT	Kr=0.6		91.4	86.6	96.3	88.8	80.0	97.5	73.4	52.8	94.1
	Kr=0.7		90.7	85.4	95.7	86.6	75.0	98.1	74.9	56.7	93.0
MIL-EN-B3	Instance	Max	88.5	86.7	90.2	84.4	75.0	93.8	**78.7**	**67.1**	90.4
		Topk	89.5	85.6	93.3	**89.7**	**82.5**	96.9	77.0	60.7	93.2
		Avg	89.1	86.7	91.6	85.0	77.5	92.5	76.5	64.3	88.7
	Embedding	Max	86.0	85.7	86.4	85.3	77.5	93.1	76.1	66.3	85.9
		Topk	89.2	85.7	92.7	88.1	82.5	93.8	76.7	59.5	93.9
		Avg	89.1	84.4	93.8	83.4	70.0	96.9	75.3	56.3	94.3
MIL-DEiT-S	Instance	Max	91.7	87.1	96.3	87.2	77.5	96.9	74.4	54.8	94.1
		Topk	91.4	86.6	96.2	84.1	72.5	95.6	71.8	49.6	94.1
		Avg	**91.8**	**87.5**	96.1	87.8	80.0	95.6	74.8	56.0	93.7
	Embedding	Max	91.0	87.4	94.5	85.3	75.0	95.6	74.7	58.3	91.1
		Topk	91.5	86.9	96.1	**89.7**	80.0	**99.4**	75.1	55.2	**95.1**
		Avg	91.4	87.4	95.4	85.0	75.0	95.0	76.0	61.1	91.0

Results on the validation set: A detailed analysis of the results in table 5.1 shows that Transformer-based architectures perform slightly better in terms of BA compared to CNN-based architectures. For example, both MIL-DEiT-S and DEiT-S achieve approximately 2% higher BA compared to the MIL-EN-B3 models. However, we cannot definitively conclude that ViT-based models generally outperform CNN-based models, as the performance difference does not appear to be substantial. In addition, changing the classification head for the DEiT-S model to a MIL classifier did not significantly affect its performance. This observation suggests that to obtain accurate results with ViTs, it may not be necessary to introduce an additional CLS token for classification purposes. This finding opens the possibility for further exploration of Transformer-based feature extractors with different classifiers, including an assessment of the individual contributions of the composing patches to the output of the model.

In terms of recall of the melanoma (MEL) class, R-MEL, the MIL-EN-B3 models effectively compete with and sometimes outperform the Transformer-based models. The MIL-EN-B3 and DEiT-S baseline models both achieved an R-MEL of 86.7%. This observation is consistent with the classic MIL formulation, suggesting that a single melanoma patch may be sufficient to classify the entire image as melanoma. However, in terms of recall for the nevus class, R-NV, Transformer-based architectures stand out, justifying the differences seen in the BA results. It is important to note that MIL models are implicitly designed to maximize R-MEL, possibly leading to lower recalls for the nevus class. Surprisingly, the MIL-DEiT-S model did not significantly decrease R-NV, which may indicate that creating complex features that combine information between different patches in the image while also considering their spatial location may be relevant to correctly identify nevi lesions.

Results on the $\mathbf{PH^2}$ test set: Contrary to the results observed on the validation set, the MIL-EN-B3 competes effectively with Transformer-based models on the PH^2 test set. In particular, the *instance-level* MIL-EN-B3 model using the *top-k average* pooling operator achieves approximately 3% higher BA compared to the DEiT-S model and 1% higher compared to the best EViT configuration. This suggests that Transformer-based architectures do not necessarily generalize better than CNN-based architectures. Furthermore, the performance of the *instance-level* MIL-EN-B3 with *top-k average* pooling implies that specific ROIs may be sufficient for the diagnosis of melanoma (MEL) lesions. In contrast, Transformer-based architectures may perform better in identifying nevus lesions, highlighting the importance of correlations and spatial location of patches in the image to correctly identify nevus lesions. R-MEL results between the DEiT-S model and the EViT configurations suggest that more patches do not necessarily lead to better performance, reinforcing the idea that specific ROIs are crucial for correct melanoma classification. However, this is different for R-NV, where the EViT configuration with $\mathrm{Kr} = 0.6$ performs slightly worse than the $\mathrm{Kr} = 0.7$ configuration and the DEiT-S baseline model.

Results on the Derm7pt test set: Similar to the PH^2 test set results, Transformer-based models do not necessarily generalize better than CNN-based architectures. The best BA and R-MEL results

on this test set come from the *instance-level* MIL-EN-B3 with *max* and *top-k average* pooling. This strengthens the argument that specific ROIs may be sufficient for the diagnosis of melanoma (MEL) lesions. These observations suggest that the number of patches used for classification does not always correlate with better BA performance. In the same manner, the DEiT-S models also achieve slightly worse BA compared to the EViT-S model with Kr $= 0.7$. Similar to the previous two datasets, the Transformer-based models performed better in identifying nevus lesions.

In summary, although ViTs exhibited slightly better performance on the validation set compared to CNNs and various MIL models, this superior performance did not consistently carry over to the generalization tests. In fact, on both generalization tests, the *instance-level* MIL-EN-B3 achieve the best BA and R-MEL results. As a result, it remains uncertain whether ViTs have definitively surpassed CNNs in CV tasks, particularly in classification.

The MIL-DEiT-S model showed the highest BA and R-MEL results on the validation set. This result suggests that further exploration of ViT-based feature extractors in conjunction with alternative classifiers, such as the MIL classifier, has the potential to uncover interesting findings.

In particular, we observed an interesting trend for the MIL branch across all tested datasets. Contrary to the conventional view suggested by the literature [34], the *embedding-level* approach did not consistently outperform the *instance-level* models. In fact, the *instance-level* models often outperformed their *embedding-level* counterparts. These results underscore the ability of the *instance-level* approach to provide valuable clinical insight to physicians by identifying *key patches* that influence bag classification results, while achieving very good performance. In addition, the *instance-level* models also demonstrated competitive performance against both the CNN and Transformer architectures, reinforcing the previous statement.

5.1.2 Are All Patches Equally Important in Dermoscopy Image Analysis?

Throughout our evaluation of different datasets, an interesting trend emerged. When considering the recall of melanoma (MEL), denoted as R-MEL, we observed that models using fewer patches for classification often competed effectively and even outperformed those using all patches. For example, both the *instance-level* MIL-EN-B3 models using the *max* and *top-k average* pooling operators achieved the best results on the Derm7pt [53] and PH^2 [45] test sets, respectively. This suggests that the accurate diagnosis of melanoma (MEL) lesions may rely primarily on the identification of specific regions within dermoscopy images, aligning with the 7-point checklist criterion [9]. However, the same trend did not hold for R-NV, where the MIL-EN-B3 models generally achieved lower recalls compared to Transformer-based architectures. This suggests that correlations between patches and their spatial positions may be important in nevus identification.

To further investigate the effect of the number of patches used for classification, we performed a

special experiment. We evaluated the BA for several *instance-level* MIL pooling operators, including the *maximum* operator, the *average* operator, and three different *top-k average* operators ($k = 12.5\%$, $K = 25\%$, and $k = 50\%$). These approaches were tested on different MIL backbones, as shown in Figure 5.1.

As Figure 5.1 shows, there is no conclusive evidence regarding the optimal number of patches for the classification process in the MIL branch. For CNN-based backbones, the *top-25% average* operator yielded the highest BA results for three of the five backbone architectures, and ranked second for the other two. However, for MIL models with Transformer-based backbones, the *average* pooling operator, which uses all patches, appeared to perform better. Nonetheless, the difference in performance between BA and other MIL pooling operators was less than 1%, indicating its limited significance. Thus, it appears that not all patches in a dermoscopy image are equally relevant for accurate classification of a dermoscopy image.

The results from the various EViT-S configurations and the DEiT-S model support this assessment. In both generalization tests, the EViT-S configurations outperformed their DEiT counterparts. Specifically, the EViT-S configuration with Kr $= 0.6$ showed better BA and R-MEL in the PH^2 test set, while for Derm7pt the EViT-S configuration with Kr $= 0.7$ achieved higher BA and R-MEL than DEiT-S.

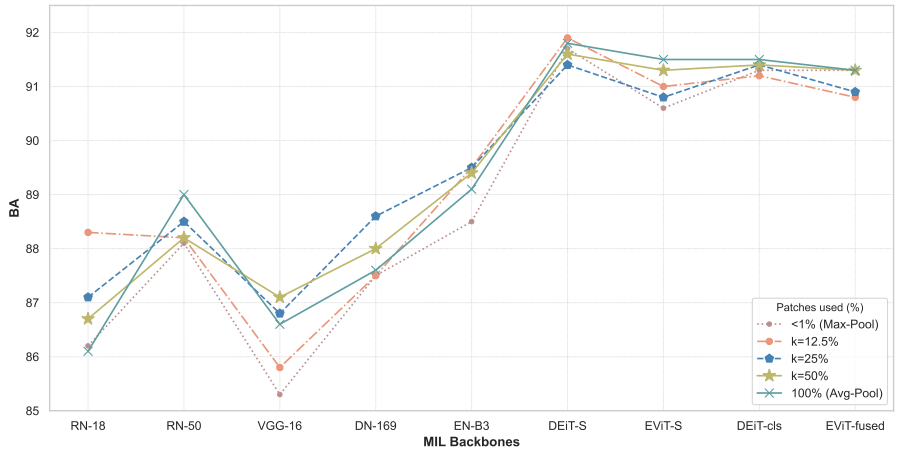

Figure 5.1: Performance comparison of various *instance-level* MIL frameworks using different MIL pooling operators, including the *maximum* operator, the *average* operator, and the *top-k average* operator with three different configurations: $k \approx 12.5\%$, $k = 25\%$, and $k = 50\%$. Our evaluation included extensive experiments with multiple CNN-based and Transformer-based backbones, such as RN-18, RN-50, VGG-16, DN-169, EN-B3, EViT-S, EViT-fused, DEiT-S, and DEiT-cls. We conducted these evaluations on the ISIC 2019 validation dataset [3–5].

While it is a stretch to assume that fewer patches used for classification are always better, it is evident that certain patches are more significant in dermoscopy image analysis, especially in melanoma (MEL) lesion identification. These “key patches” deserve special attention and may provide invaluable insights

for clinical skin cancer diagnosis.

5.1.3 Is Spatial Information Relevant in Dermoscopy Image Analysis?

In our exploration of dermoscopy image analysis, we also set out to understand the importance of spatial composition in a dermoscopy image. To investigate this aspect, we performed a special experiment. In this experiment, we removed the positional embedding matrix, $\mathbf{E}_{\mathbf{pos}}$, from the ViT pipeline, i.e., it was not added to the sequence of patch embeddings in (2.3). We applied this experiment only to Transformer-based architectures. The results obtained for the validation set and both test sets are summarized in table 5.2.

Table 5.2: Results of the best-performing models in the EViT branch, the MIL branch with Transformer-based backbones, and the DEiT-S baseline model, **without** positional encoding. These models include the two best EViT configurations (Kr $= 0.6$ and Kr $= 0.7$) and the best-performing MIL framework with a ViT-based backbone (MIL-DEiT-S). All models were validated using the ISIC2019 validation set [3–5] and the two test datasets: PH^2 [45] and Derm7pt [53].

	Models		ISIC 2019			$\mathbf{PH^2}$			Derm7pt		
			BA	R-MEL	R-NV	BA	R-MEL	R-NV	BA	R-MEL	R-NV
	DEiT-S		**90.3**	85.3	**95.4**	80.3	65.0	95.6	72.0	53.2	90.8
	EViT (Kr=0.6)		88.6	83.2	94.0	**82.5**	**72.5**	92.5	72.9	52.0	93.7
	EViT (Kr=0.7)		88.9	83.4	94.4	80.3	62.5	98.1	72.4	50.8	**94.1**
MIL-DEiT-S	Instance	Max	88.8	83.5	94.2	73.1	50.0	96.3	**72.8**	53.2	92.3
		Topk	90.1	**85.8**	94.3	80.0	65.0	95.0	72.4	**56.3**	88.5
		Avg	89.2	83.5	94.9	79.1	60.0	98.1	71.4	52.8	90.1
	Embedding	Max	88.8	83.5	94.2	75.3	52.5	98.1	71.3	50.0	92.5
		Topk	90.0	85.5	94.6	80.3	65.0	95.6	70.8	51.2	90.4
		Avg	89.2	83.5	94.9	81.6	65.0	**98.1**	70.9	51.2	90.6

Results on the validation set: Excluding positional encoding resulted in a slight decrease in BA performance (about 1% to 2%) compared to models **with** positional encoding (as shown in table 5.1). This decrease is also reflected in the R-MEL and R-NV values. Although there was a slight decrease in results, this decrease is not sufficiently significant to conclude that spatial information in dermoscopy images is essential to obtain accurate results. Comparing the results in Table 5.2 with those in Table 5.1, it is interesting to note that Transformer-based architectures without positional encoding perform remarkably close to the MIL models in Table 5.1. This suggests that the ViT and MIL frameworks may in fact be closer than originally thought.

Results on the $\mathbf{PH^2}$ test-set: The BA results **without** position encoding on this test set decrease significantly compared to the results in table 5.1. The R-NV remains relatively stable without positional encoding, suggesting that spatial patch positions may be less critical in identifying nevus lesions. However, for R-MEL there is a significant drop in performance, suggesting that spatial composition may be

critical for identifying melanoma (MEL) lesions. It is important to note that the PH^2 test set consists of only 40 melanoma samples, so a few misclassifications can have a significant impact on R-MEL.

Results on the Derm7pt test set: On the Derm7pt test set, BA performance decreases slightly (about 1% to 2%), similar to the validation set. However, R-MEL without positional encoding does not show a significant decrease compared to the results of **with** positional encoding (Table 5.1). The performance of R-NV seems to have decreased, suggesting that spatial information may be relevant for the identification of nevus lesions in dermoscopy images.

In summary, the inclusion of positional information in ViTs appears to slightly improve model performance. Due to the specific characteristics of the PH^2 test set, our conclusions are based only on the results of the validation set and the Derm7pt test set. For accurate identification of nevi lesions, positional encoding appears to improve the overall performance. However, we suspect that there is another specific property of Transformer-based models that is more important for the correct identification of nevus lesions. As can be seen in table 5.1, Transformer-based architectures exceed CNN-based models in nevus lesion identification. This is due to the fact that ViTs have a wider receptive field over the entire image in earlier layers, which allows it to generate features that depend on correlations between patches from opposite sides of the image. This ability is not typical in CNN architectures. We hypothesize that ViTs generate features with global information, which is advantageous for identifying nevi lesions. In the case of melanoma (MEL) lesion identification, the influence of positional encoding is less obvious. We believe that the positional embedding matrix, $\mathbf{E_{pos}}$, represents an additional learnable parameter that provides the model with more flexibility, incorporating spatial information. While this parameter improves the performance of Transformer-based models, it may not be the primary factor that differentiates ViTs from the MIL framework.

5.2 Multi-class Problem

Table 5.3 provides a comparative analysis of the results for the method described in section 3.2.3 for different model configurations accordingly with the settings established in section 4.4.

In the case of the multi-class problem, we can observe that the baseline models and the EViT-S configurations generally outperform the proposed *instance-level* MIL implementations. For example, the *instance-level* MIL-EN-B3 model achieved a BA of 78.8%, which is 4.4% lower than the EN-B3 baseline model. This discrepancy is more significant than what we observed in the results for the binary classification problem (see section 5.1). However, it is important to note that the traditional MIL assumption is inherently aligned with the binary classification problem, which revolves around defining positive and negative classes. Defining positive and negative classes in the context of the multi-class problem is more complex. Therefore, the proposed method represents a relaxation of the classical MIL formulation

adapted to multi-class problems.

Both proposed *instance-level* methods perform competitively with the *embedding-level* approach. Although the *embedding-level* approach does not identify *key instances*, it serves as a "bridge" to traditional CNN-based architectures. A comparison of the BA results of the *embedding-level* MIL-EN-B3 with the EN-B3 baseline model shows that the observed discrepancies are consistent with those seen for the two *instance-level* methods. We hypothesize that the choice of configuration for the path extractor, E, in the MIL branch plays a significant role in these results. As explained in chapter 3, we extract a 14×14 feature map. Whereas, the EN-B3 model produces a 7×7 feature map, with a much deeper embedding dimension. Since the multi-class problem is more complex, the preference for a deeper model that extracts more complex features may explain the observed discrepancies.

Table 5.3: Results from the best-performing models in the EViT branch, MIL branch, and baseline architectures. These models include the best-performing baseline models (EN-B3 and DEiT-S), the two best EViT configurations (Kr $= 0.6$ and Kr $= 0.7$), and the best-performing MIL backbones (MIL-EN-B3 and MIL-DEiT-S). 'Inst. 1^{st}' and 'Inst. 2^{nd}' stand for the first and second *instance-level* methods proposed in section 3.2.3. All models were evaluated using the ISIC2019 validation set [3–5].

Models			ISIC 2019								
			BA	R-AK	R-BCC	R-BKL	R-DF	R-MEL	R-NV	R-SCC	R-VASC
EN-B3			82.2	71.1	87.8	79.4	81.3	76.5	91.6	72.2	98.0
DEiT-S			83.6	72.3	90.5	**82.7**	87.5	**80.3**	92.1	65.1	98.0
EViT (Kr=0.6)			83.6	71.7	89.0	80.4	**93.8**	77.8	91.3	66.7	98.0
EViT (Kr=0.7)			**84.3**	78.6	**90.7**	80.4	87.5	75.6	**93.5**	69.8	98.0
MIL-EN-B3	Inst. 1^{st}	Max	74.1	57.2	81.5	74.3	77.1	74.6	77.8	61.9	88.2
		Topk	78.4	72.8	78.8	75.6	79.2	74.9	87.3	68.3	90.2
		Avg	79.9	71.7	86.4	78.1	83.3	76.8	84.1	62.7	96.1
	Inst. 2^{nd}	Max	76.4	72.3	82.4	75.4	72.9	66.7	80.0	65.1	96.1
		Topk	76.2	75.7	77.1	65.5	79.2	71.0	82.5	66.7	92.2
		Avg	77.5	68.2	86.5	74.1	77.1	73.5	81.2	69.0	90.2
	Embed.	Max	72.3	55.5	77.6	73.7	68.8	66.9	79.0	70.6	86.3
		Topk	78.9	66.5	86.3	79.6	81.3	74.2	84.1	66.7	92.2
		Avg	77.6	68.8	84.2	77.3	77.1	77.4	80.7	63.5	92.2
MIL-DEiT-S	Inst. 1^{st}	Max	82.2	72.8	88.7	81.1	89.6	80.1	82.7	64.3	98.0
		Topk	78.4	78.0	84.0	73.0	66.7	78.4	88.4	68.3	90.2
		Avg	81.6	76.3	85.7	72.0	89.6	75.4	88.6	69.0	96.1
	Inst. 2^{nd}	Max	75.4	70.5	80.9	75.8	72.9	66.6	83.1	61.1	92.2
		Topk	79.0	74.0	84.5	75.6	72.9	76.3	87.2	65.1	96.1
		Avg	82.6	79.2	87.8	76.2	87.5	77.9	91.1	66.7	94.1
	Embed.	Max	82.4	**80.9**	87.8	75.1	83.3	76.4	89.0	**74.6**	92.2
		Topk	82.2	73.4	83.6	74.3	93.8	75.6	92.2	69.1	96.1
		Avg	82.6	70.5	88.7	79.8	89.6	75.1	91.5	65.9	**99.9**

The EViT branch achieves better BA performance compared to the baseline models, which is consistent with the concept that not all patches in a dermoscopy image have the same relevance. In addition,

Transformer-based models perform well, suggesting that constructing features based on patch correlations and introducing some spatial context may be advantageous for diagnosing dermoscopy images.

In summary, the multi-class problem presents challenges due to the conventional MIL assumption, which is closely tied to binary classification. The definition of positive and negative classes in a multi-class context is a complex task. Our proposed MIL formulation addresses this challenge and serves as a bridge between traditional MIL and the multi-class scenario. The configuration of the patch extractor, E, can explain the discrepancies in performance, since the choice of the size of the feature map is different between the MIL branch and the EN-B3 model. Since the multi-class problem is more complex, deeper models that extract more complex features may be preferred. Thus, we believe that our proposed formulation paves the way for multi-class MIL models.

5.3 Assessment of the Regions of Interest (ROIs)

Both the EViT and MIL branches have consistently produced competitive results when compared to the CNN-based and Transformer-based baseline models, for both binary and multi-class problems. To further our understanding of their performance, we will now turn our attention to the different ROIs identified by these two branches in the context of the binary classification problem of distinguishing melanoma (MEL) from nevus (NV). In addition, to assess whether the proposed framework can provide valuable insights to physicians by performing a detailed patch-level analysis of dermoscopy images, we will compare the identified ROIs with the traditional Grad-Cam technique [28].

5.3.1 Identification of ROIs by the EViT Branch

In this section, we will examine the visualization techniques employed by the EViT branch. Figure 5.2 visually demonstrates the inner workings of the EViT branch. The first row shows the input images. The second row illustrates the patch removal process within the EViT architecture. The third row contains a heatmap showing the influence of each patch on the model's output, and the fourth row shows the corresponding gradient information. This Figure shows the different visualization processes used by two EViT-S models: one **with** and one **without** positional encoding.

Visualization of the EViT Mask: Typically, when the dermoscopy images contain a small lesion and the majority of the image shows healthy skin, the EViT architecture tends to discard patches around the lesion. This process typically starts at the corners of the image and extends to the healthier regions of the skin. However, in both scenarios (**with** and **without** positional encoding), if the lesion is relatively large and essentially covers the entire dermoscopic image (which is often the case with melanoma lesions), some patches within the skin lesion will inevitably be removed. Consequently, while the EViT mask can perform something like a patch-level segmentation of the image, the static nature of the Kr

hyper-parameter may result in the exclusion of potentially important patches, particularly those located inside the skin lesion. A more comprehensive exploration of the Kr hyper-parameter may be worth pursuing, exploring the feasibility of designing a non-fixed Kr parameter.

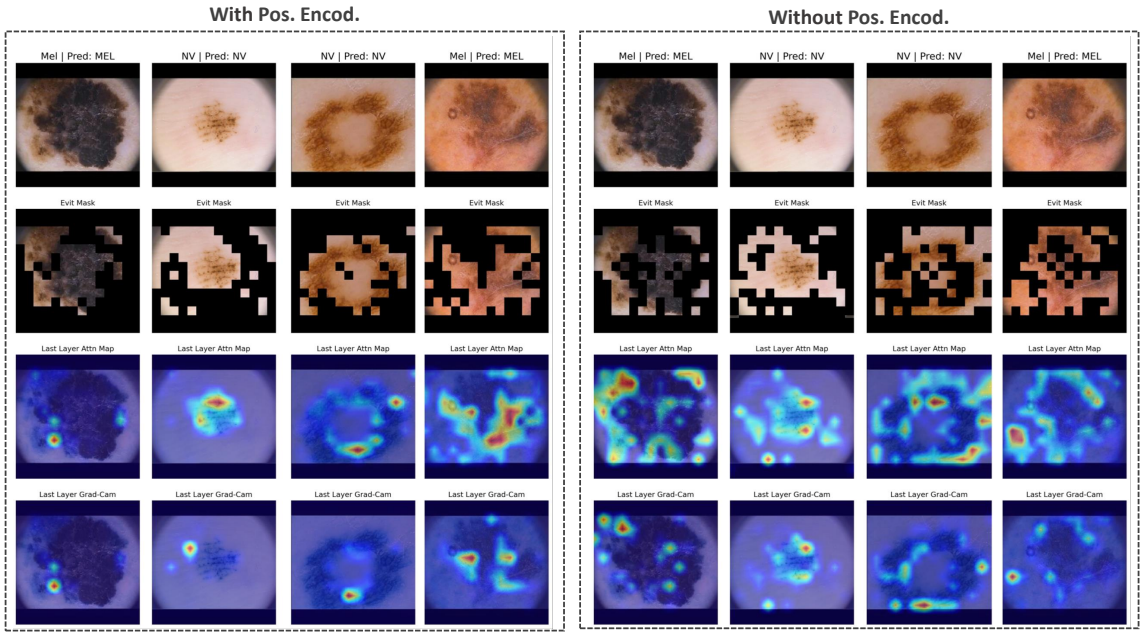

Figure 5.2: Visualization of the EViT architecture process with Kr $= 0.7$ and the default configuration for the placement of token reorganization blocks. On the left we have the visualizations for the EViT-S model **with** positional encoding, while on the right we have the visualizations **without** the inclusion of positional encoding. The first row displays input images from the PH^2 test set [45]. The second row illustrates the inattentive patches removed by the EViT model. The third row exhibits the attentiveness of the CLS token in the final encoder block. This heatmap is created by transforming the first row of the attention map $\mathbf{A}$ (2.5), represented as $\boldsymbol{a} \in \mathbb{R}^{1\times N}$, into a 14×14 heatmap. This heatmap is then transformed into a $224 \times 224 \times 3$ representation using bilinear interpolation. The last row shows the gradient-based visualization of CLS embedding attentiveness, inspired by Chefer *et al.* [50].

Attention Visualization: In Figure 5.2, the third row showcases the attention of the CLS token in the last encoder block. The heatmap is created by transforming the first row of the attention map $\mathbf{A}$ (2.5), represented as $\boldsymbol{a} \in \mathbb{R}^{1\times N}$, into a 14×14 heatmap. This heatmap is then interpolated into a $224 \times 224 \times 3$ representation.

These visualizations show significant differences between the EViT model **with** positional encoding and the model **without** it. In particular, the addition of positional encoding seems to confine the most attentive patches to those located within the skin lesion. In contrast, the model without positional encoding assigns importance to more patches, which are mainly located around the borders of the lesion.

In summary, our visualizations comparing the configurations **with** and **without** positional encoding show that the addition of positional encoding tends to focus attention on patches within the skin lesion. Conversely, the removal of positional encoding encourages the EViT model to pay more attention to

the border regions of the lesion. However, it remains inconclusive which set of visualizations might offer more practical benefits in clinical contexts, as one prioritizes patches within the lesion while the other emphasizes the borders. For additional visualization examples illustrating the impact of positional encoding on the EViT branch, please refer to section B.3.1 in appendix B.

5.3.2 Identification of ROIs by the MIL Branch

As previously mentioned, the existing literature often emphasizes the superior performance of *embedding-level* approaches within the MIL framework as opposed to *instance-level* methods. However, our results challenge this convention by showing that *instance-level* approaches can produce competitive results, often surpassing those of *embedding-level* models. This suggests that *instance-level* methods effectively identify the *key patches* that influence the model's predictions, while maintaining strong performance against baseline methods. To determine whether the ROIs identified by *instance-level* MIL models actually contain more specific and meaningful information compared to CNN-based architectures, we will examine the ROIs identified by two different *instance-level* models and compare them to those identified by an *embedding-level* approach. In this context, the *embedding-level* model serves as a bridge between the MIL framework and conventional CNN architectures, making it a solid representative of CNN-based architectures.

5.3.2.A *Instance-level* Approach

Figure 5.3 provides a visual representation of the visualization pipeline outlined in section 4.4.2 for the *instance-level* MIL-EN-B3 model, using both the *maximum* and *top-k average* pooling operators. It is important to emphasize that in order to simplify the visualization process, we have chosen the *top-k average* operator with a value of approximately 12.5% ($k \approx 12.5\%$), which corresponds to selecting 25 patches to represent the entire image. The first three rows of this visualization pipeline remain the same for both models. The first row displays the input dermoscopy images. The second row presents the patch probability heatmap for the melanoma (MEL) class, while the third row provides insight into the gradients associated with the same class. The fourth row, however, differs between the two models. For the *instance-level* MIL model, which uses the *maximum* operator, the fourth row reveals the *key patch* identified by the model. In contrast, the fourth row of the MIL model using the *top-k average* pooling operator illustrates the gradients associated with the nevus (NV) class.

Note that the *instance-level* approach with the *average* pooling operator is not presented in this section. As noted in section 4.4.2, this pooling operator considers all patches to represent the bag, making it impossible to isolate the most relevant patches in the image as all of them are considered for classification.

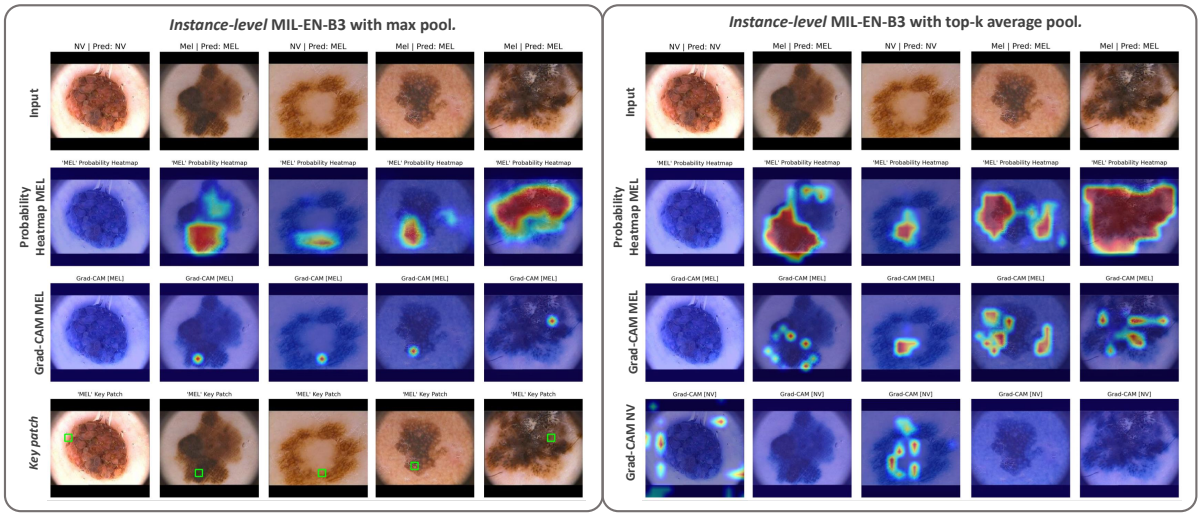

Figure 5.3: Visualization of two different MIL branch processes, specifically the *instance-level* MIL models using the *max* and *top-k average* pooling operators. To simplify the visualization process, we chose the *top-k average* operator with a value of approximately 12.5% ($k \approx 12.5\%$), which is equivalent to selecting 25 patches to represent the entire image. The backbone architecture used for these MIL models is the EN-B3. The images used for visualization are taken from the PH^2 test set [45] and refer to the binary classification task of melanoma versus nevus. On the left is the visualization pipeline for the MIL model using the *max* pooling operator, while on the right is the visualization framework for the model using the *top-k average* pooling operator. The first three rows of this visualization pipeline remain the same for both models. The top row shows the input dermoscopy images. The second row illustrates the patch probability heatmap for the melanoma class, while the third row displays the gradients for the same class. The fourth row diverges between the two models. In the case of the *instance-level* MIL model using the *max* operator, the fourth row shows the *key patch* identified by the model. The fourth row of the MIL model using the *top-k average* pooling operator provides insight into the gradients associated with the nevus class. Note that these images use a gradient-based visualization method inspired by the work of Chefer *et al.* [50].

Visualization of patch probabilities: In the context of the visualization pipeline introduced in section 4.4.2, the first row refers to the heatmap probability for the melanoma class. This heatmap is designed so that reddish regions correspond to probability values closer to 1, while bluish regions represent values closer to 0. Regions with shades of green and yellow denote probability values close to 0.5. In Figure 5.3, the visualizations are specifically focused on the binary classification task of melanoma (MEL) versus nevus (NV). Consequently, it is clear that the probability heatmap for the nevus class is essentially an inverted version of the one shown in the first row. This inversion implies that bluish regions become reddish and vice versa.

Analysis of these probability heatmaps reveals a general trend: for melanoma lesions, patches with high melanoma probabilities tend to be concentrated within the lesion boundaries. Patches outside the lesion boundaries, such as healthy skin areas or image corners, typically have low melanoma probabilities (close to 0). As a result, the probability of nevus is often high in these areas outside the lesion boundaries.

When the model confidently identifies a lesion as a nevus, the melanoma probabilities are significantly low across the entire image. In contrast, the probabilities of nevus patches are consistently high (close to 1) for all image patches. This contrast suggests that for accurate melanoma detection, only specific ROIs are truly relevant, whereas for nevus detection, all information in the image is relevant to the model.

Comparison of the model using *max* pooling with the one using *top-k average* pooling shows that the latter typically assigns high melanoma probabilities to more patches (i.e., reddish regions are often more extensive). This is natural since the model using *top-k average* pooling evaluates a larger number of patches to determine whether the entire lesion is melanoma or not.

Gradient-based visualization: When examining the MIL model using the *maximum* pooling operator, it becomes clear that the gradients provide insight into the *key patches* identified by the model.

For the MIL model with the *top-k average* pooling operator, the gradients reveal the most relevant patches for the model's classification process. Essentially, this model can be considered a more robust version of the MIL model with the *max* pooling operator, as it considers the k patches with the highest melanoma probability to determine the melanoma probability of the entire image. In the traditional MIL framework, represented by the *instance-level* MIL model with *max pooling*, a single positive instance is sufficient to classify the entire bag as positive. This approach is implicitly designed to maximize melanoma recall, since the presence of a single melanoma patch causes the entire image to be classified as a melanoma. However, this can lead to the misclassification of some nevi lesions as melanomas, as shown in the third column of Figure 5.3. In contrast, the MIL model using the *top-k average* operator is usually more robust to this type of misclassification because it does not base its classification on a single patch, but rather on a specific set of patches.

With the *top-k average* pooling operator, we observe that for melanoma cases, the *key patches* identified by the model are mainly within or at the borders of the lesion. In contrast, when visualizing the gradients related to the nevus class, most of the *key patches* identified by the model correspond to healthy skin areas. This reinforces the notion that correct identification of nevus lesions benefits from considering different regions of the dermoscopic image, such as healthy skin areas. However, the same principle is not necessarily true for melanoma lesions, where patches within the lesion borders are critical for accurate classification.

In summary, the *instance-level* visualization pipeline outlined in section 4.4.2 effectively identifies *key patches* within the image. These *key patches* can be viewed as ROIs, which may provide valuable insights into clinical skin cancer diagnosis. We have observed that, in general, for melanoma lesions, the *key patches* tend to be located within the lesion or at its border. Conversely, for nevi lesions, the model identifies *key patches* predominantly at the border or within healthy skin areas.

When comparing the *top-k average* and the traditional *max* pooling operators, it is evident that the

top-k average pooling operator exhibits greater robustness. This can be attributed to the fact that *top-k average* does not rely solely on the melanoma probability of a single patch, but rather considers a specific set of patches. As a result, the *top-k average* operator mitigates the misclassification of nevi lesions when compared to the *max* operator. In future work, it would be worth exploring more sophisticated *top-k average* pooling operators within the MIL framework.

5.3.2.B Comparison between *Instance-level* and *Embedding-level* Approaches

The visualizations shown in Figure 5.3 are a product of the visualization pipeline outlined in section 4.4.2. In this pipeline, gradients are computed during inference and then multiplied by the patch probability heatmap of the corresponding class. However, in the case of the *embedding-level* approach, this visualization pipeline is not applicable because the *embedding-level* approach is not able to compute the independent patch probabilities. To gain insight into the ROIs for *embedding-level* models, we use the standard Grad-Cam technique [28].

This section provides a comparative analysis between the ROIs identified by the *instance-level* MIL model using the *top-k average* pooling operator (as shown in Figure 5.3) and those identified by the corresponding *embedding-level* model using the *column-wise global average* pooling operator, as shown in Figure 5.4.

When we compare the Grad-Cam visualizations shown in Figure 5.4 with the gradient visualizations of the *instance-level* MIL model using *top-k average* pooling (shown in Figure 5.3), significant differences become apparent.

In the Grad-Cam visualization of the *embedding-level* approach, there is a higher occurrence of greenish and yellowish regions compared to the *instance-level* counterpart. This suggests that the *embedding-level* model has greater uncertainty in classifying patches as melanoma or nevus when compared to the *instance-level* model. Consequently, the ROIs generated by the *embedding-level* approach, analogous to CNN architectures, do not provide as much valuable information at the patch level. In fact, the regions generated by the *embedding-level* approach appear more like blurs when compared to the melanoma gradients produced by the *instance-level* model. By comparing these two grad-cam techniques, it is clear that the *instance-level* approach provides a more accurate patch-level identification of the most relevant patches. This, in turn, contributes to the identification of *key patches* on which the model bases its classification. Such patch-level analysis has the potential to be significantly more useful in clinical practice than the vague regions identified by the *embedding-level* and CNN-based architectures.

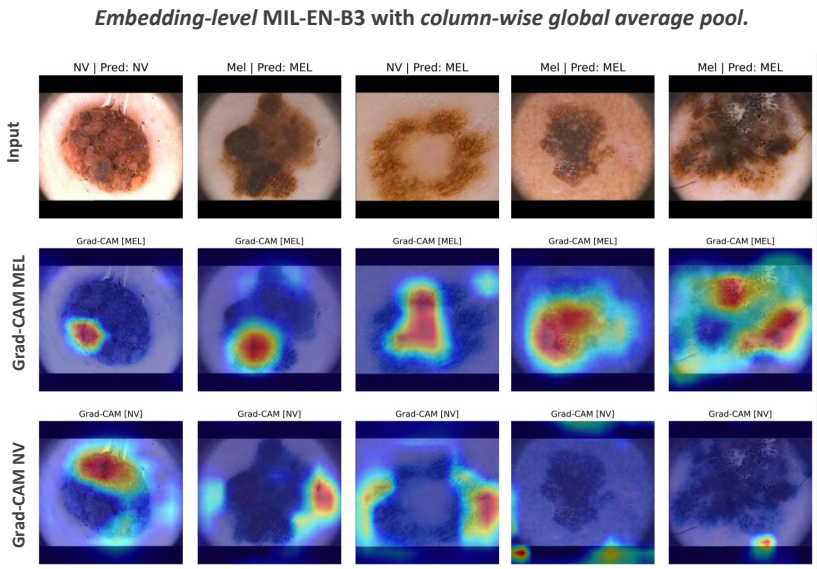

Figure 5.4: Visualization of various results generated by the *embedding-level* MIL model with the *column-wise global average pooling* operator. The backbone used for the MIL model is the EN-B3. The images used for visualization are taken from the PH^2 test set [45] and refer to the binary classification task of melanoma vs. nevus. The first row of the Figure shows the input images, while the second and third rows illustrate Grad-Cam [28] visualizations for the melanoma and nevus classes, respectively.

5.3.3 Comparison between the EViT and MIL branches

Figure 5.5 shows visualizations that highlight the most relevant patches derived from the *instance-level* MIL models using *max* and *top-k average* pooling. Additionally, this illustration includes visualizations from the EViT branch **without** positional encoding.

Upon closer examination of Figure 5.5, we can draw comparisons between the *key patches* identified by the two different MIL models and the attentive patches identified by the EViT architecture. While some relevant patches overlap between the two architectures, such cases are rather rare. Consequently, these models demonstrate the ability to identify different ROIs, but it remains inconclusive whether these regions represent matching patches within the lesion. Although it is difficult to determine which framework provides the most informative ROIs, the *instance-level* MIL framework has the advantage that the identification of *key patches* depends directly on the patch probabilities. This distinction is particularly valuable because it allows for clear segmentation of the patches classified as melanoma or nevus. In contrast, the patches identified by the attention mechanism in the EViT branch do not provide the same level of direct association with either class. This particular advantage reinforces the potential value of the *instance-level* MIL framework in automatically identifying different ROIs within a dermoscopic image.

For a more comprehensive set of visualizations illustrating the *key patches* identified by the EViT and MIL branches, refer to section B.3 within appendix B.

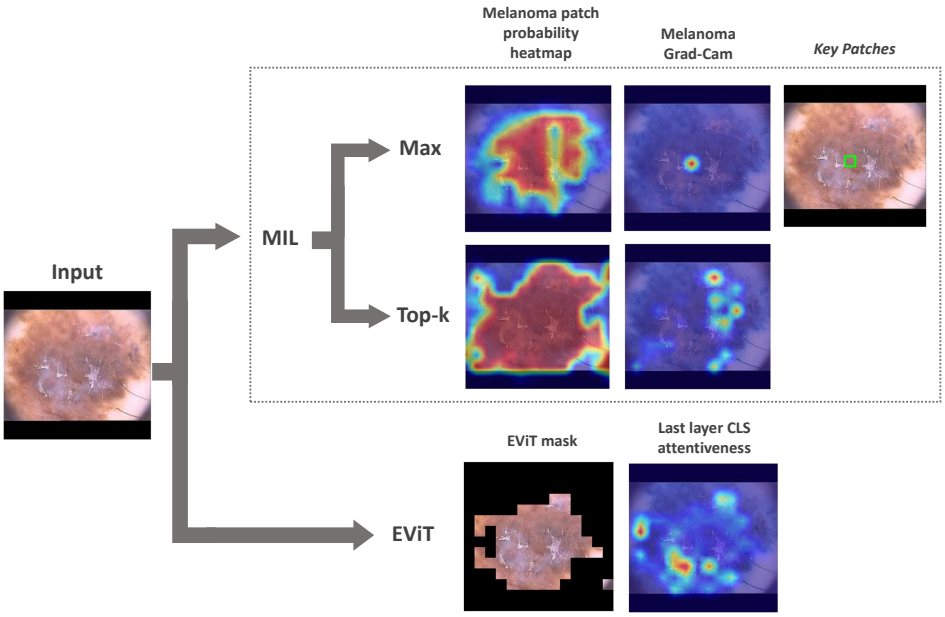

Figure 5.5: Visual exploration of the ROIs identified by the two branches within the proposed framework. It comprises two MIL models, one using an *instance-level* approach with *max* pooling and the other using *top-k average* pooling. The latter model employed a value of approximately 12.5% for k, which corresponds to selecting the top 25 patches with the highest melanoma probability. Both MIL models use the EN-B3 architecture as their backbone. The EViT branch keeps the configuration defined in section $4.4.1$, with a Kr value of 0.7, and is configured **without** positional encoding. In the MIL branch, the sequence starts by displaying the melanoma (MEL) patch probability heatmap. The specific ROIs are then revealed by visualizing gradients associated with the MEL class. In the EViT branch, the process begins with the application of the EViT mask, followed by the display of attention weights associated with the CLS token within the final EViT layer. This image is taken from the PH^2 test set [45].

6

Conclusions and Future Work

Contents

6.1 Conclusions

In this dissertation, we embarked on a quest to answer two key questions in the field of dermoscopy image analysis:

1. Are all patches within a dermoscopy image equally important, or do deep models primarily rely on specific ROIs for classification?
2. Does the spatial arrangement of patches within the dermoscopy image have a significant impact on skin cancer diagnosis?

To address these questions, we introduced a novel framework that combines two branches: one using the EViT and the other using MIL. We applied this framework to study both the binary and multi-class problems.

In the binary scenario, the proposed two-step MIL approach proved to be quite effective. It not only achieved comparable results with baseline models, but often outperformed both ViTs and CNN-based architectures, demonstrating the robustness and generalization ability of the proposed framework across different hospital domains and distribution patterns.

In the multi-class scenario, our three-stage MIL framework demonstrated competitive performance with both the Transformer-based and CNN-baseline models. Although the performance in this case was not as good as in the binary case, we believe that our proposed formulation can open the door to new MIL multi-class framework adaptations.

With respect to the automatic identification of ROIs in dermoscopy images, our results indicate that *instance-level* approaches provide more detailed visual analysis compared to the blurry visualizations produced by *embedding-level* approaches, which closely resemble CNN architectures. *Instance-level* approaches excelled in terms of performance, often outperforming *embedding-level* and CNN-based architectures. In particular, the *instance-level* MIL model with *top-k average* pooling emerged as the framework with the most significant ROIs. This is partly attributed to the robustness of the *top-k average* operator, which considers a set of specific patches rather than relying on a single patch to classify the entire image. When comparing the ROIs identified by the MIL and EViT branches, we concluded that they do not necessarily identify the same ROIs. However, we argue that the ROIs identified by the *instance-level* MIL branch contain more valuable information compared to the regions identified by the EViT branch, due to the fact that the identification of *key patches* by the *instance-level* MIL branch is directly tied to patch probabilities. This ability to distinguish between patches classified as melanoma or nevus may have significant clinical value that is not as apparent in the case of the patches identified by the EViT branch.

In both scenarios, our experiments have implicitly implied that not all patches on a dermoscopy image have the same relevance for the diagnosis of melanoma skin cancer. In the binary problem of distinguish-

ing melanoma (MEL) from nevus (NV), it is evident that the accurate identification of melanoma relies heavily on the detection of distinct, focal regions. This is consistent with the principles of the 7-point checklist criterion [9], suggesting that the identification of specific regions is crucial for the diagnosis of melanoma. On the other hand, when it comes to nevus detection, models that are good at capturing inter-patch relationships tend to outperform the rest. Thus, while the identification of melanoma depends on the detection of precise regions, the same is not true for nevus lesions.

We explored the importance of spatial positioning within the image. Our findings strongly suggest that the spatial information in the image has value in capturing lesion characteristics. In particular, the incorporation of positional encoding proved advantageous for ViT architectures. A notable observation, particularly in the evaluation of the validation set, is that when Transformer-based architectures operated without positional encoding, their results closely matched those of the MIL branch. This observation suggests that the ViT architecture may be more similar to the MIL framework than originally assumed.

In chapter 2, we established the connection between the MIL framework and the 7-point checklist criterion. We first introduced the deep MIL as an appropriate representation of the 7-point criterion in the domain of DL. The two paradigms are quite similar in their approach. Both score each patch independently, regardless of their spatial location within the dermoscopic image. They then use a permutation-invariant function to aggregate these individual scores into an overall classification of the dermoscopic image.

While establishing this connection, we also made sure to point out the differences between the MIL and ViT frameworks. Unlike MIL, the ViT architecture exploits intricate patch correlations, taking into account their positions within the image to produce highly complex features. However, a deeper analysis of the ViT architecture led to a very interesting finding: **without** the inclusion of positional encoding, ViTs could actually be characterized as a variant of the MIL framework. In this context, the Transformer encoder block essentially served as a sophisticated MIL pooling operator. The results obtained are consistent with this observation, suggesting that the ViT architecture and the MIL framework may share more similarities than originally believed. In fact, our experiments show that ViT models, **without** positional encoding, perform similarly to CNN-based MIL architectures. This discovery highlights an interesting connection between the ViT architecture and the MIL framework. While initially perceived as distinct methodologies, their similarities became apparent, especially in the absence of positional encoding, challenging our initial assumptions.

6.2 Future Work

In this thesis, we have provided critical insights into dermoscopy image analysis, highlighting the importance of region emphasis, spatial context, and the intriguing relationship between MIL and SA-based

architectures. Nevertheless, several areas still require further investigation and research.

Within the MIL branch, our findings underscore the potential of the *instance-level* approach coupled with the *top-k average* pooling operator in capturing valuable ROIs in dermoscopy image analysis. Future research on this MIL pooling operator may hold great promise. For example, we could explore an *instance-level* MIL framework that incorporates a *top-k average* operator with a fixed k during its initial training phase. Subsequently, after this initial training phase, the k parameter would become a learned variable, meaning that only gradients associated with patches predicted to be melanoma would be updated. We believe that further experiments with different variants of the *top-k average* pooling operator could yield interesting results.

Within the EViT branch, we observed the behavior of the Kr parameter, which basically performs a learned patch-level segmentation of the lesion. However, given the complex and varied nature of dermoscopic images, a fixed Kr parameter may lead to discrepancies in the results. In cases where the image contains predominantly healthy skin, substantial patch removal may be appropriate. However, in scenarios where the skin lesion occupies most of the image, removing too many patches could result in the loss of critical information. Therefore, it would be valuable to explore the feasibility of a non-fixed Kr parameter that adapts to different image scenarios.

With respect to the identification of ROIs, there is a need for further exploration of evaluation metrics that effectively measure the quality of identified ROIs. One avenue of exploration could be to use publicly available segmentation masks to determine whether the *key patches* identified by the model are predominantly inside or outside the boundaries of the skin lesion. In addition, an intriguing direction of study is to evaluate whether the ROIs identified by the MIL and EViT branches meet established dermoscopy criteria, such as assessing general lesion asymmetry, irregular borders, color variations, and other criteria.

In the context of the multi-class scenario, we have introduced a novel three-stage MIL formulation that shows promise for advancing multi-class MIL frameworks. However, much remains to be explored in this scenario. Extensive testing is needed to evaluate the generalization capability of the proposed three-step approach. In addition, exploring alternative approaches to the multi-class setting, such as transforming it into multiple binary classification tasks where each class is paired against all others, could open new avenues for investigation and development.

Bibliography

[1] 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," *International Conference on Learning Representations*, 2020.

[2] S. Aladhadh, M. Alsanea, M. Aloraini, T. Khan, S. Habib, and M. S. Islam, "An effective skin cancer classification mechanism via medical vision transformer," *Sensors*, 2022.

[3] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset: A large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, 2018.

[4] N. C. F. Codella, D. A. Gutman, E. Celebi, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, N. Mishra, A. Kalloo, A. Kalloo, K. Liopyris, N. K. Mishra, H. Kittler, and A. C. Halpern, "Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (isbi) 2016, hosted by the international skin imaging collaboration (isic)," *arXiv: Computer Vision and Pattern Recognition*, 2016.

[5] M. Combalia, N. C. F. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, O. Reiter, A. C. Halpern, S. Puig, and J. Malvehy, "Bcn20000: Dermoscopic lesions in the wild." *arXiv: Image and Video Processing*, 2019.

[6] P. Kaufman. (2022) Skin cancer signs, symptoms, treatment, and more. Accessed 25-Dec-2022. [Online]. Available: https://www.everydayhealth.com/skin-cancer/guide/

[7] S. C. F. Website. (2022) Skin cancer facts statistics. Accessed 25-Dec-2022. [Online]. Available: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/

[8] M. Goyal, T. Knackstedt, S. Yan, and S. Hassanpour, "Artificial intelligence-based image classification methods for diagnosis of skin cancer: Challenges and opportunities," *Computers in Biology and Medicine*, vol. 127, p. 104065, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482520303966

[9] G. D. Leo, C. Liguori, A. Pietrosanto, G. Fabbrocini, and M. Sclavenzi, “Elm image processing for melanocytic skin lesion diagnosis based on 7-point checklist: a preliminary discussion,” 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:11427159

[10] M. E. Celebi, C. Barata, A. Halpern, P. Tschandl, M. Combalia, and Y. Liu, “Guest editorial skin image analysis in the age of deep learning,” *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 1, pp. 143–144, 2023.

[11] A. Adegun and S. Viriri, “Deep learning techniques for skin lesion analysis and melanoma cancer detection: a survey of state-of-the-art,” *Artificial Intelligence Review*, 2020.

[12] S. Haggenmüller, R. C. Maron, A. Hekler, J. Utikal, C. Barata, C. Barata, R. L. Barnhill, R. L. Barnhill, H. Beltraminelli, C. Berking, C. Berking, B. Betz-Stablein, B. Betz-Stablein, A. Blum, S. A. Braun, S. A. Braun, R. Carr, R. A. Carr, M. Combalia, M. Combalia, M. T. Fernández-Figueras, M. T. Fernández-Figueras, M.-T. Fernandez-Figueras, G. Ferrara, S. Fraitag, S. Fraitag, L. E. French, F. F. Gellrich, K. Ghoreschi, M. Goebeler, R. A. Scolyer, P. Guitera, H. A. Haenssle, S. Haferkamp, S. Haferkamp, L. Heinzerling, L. Heinzerling, M. V. Heppt, F. J. Hilke, S. Hobelsberger, D. Krahl, H. Kutzner, A. Lallas, K. Liopyris, K. Liopyris, M. Llamas-Velasco, M. Llamas-Velasco, J. Malvehy, F. Meier, C. S. Müller, C. S. L. Müller, A. A. Navarini, A. A. Navarini, C. Navarrete-Dechent, C. Navarrete-Dechent, A. Perasole, A. Perasole, G. Poch, S. Podlipnik, L. Requena, V. Rotemberg, V. M. Rotemberg, A. Saggini, A. Saggini, O. P. Sangueza, O. P. Sangueza, C. Santonja, C. Santonja, D. Schadendorf, B. Schilling, M. Schlaak, J. G. Schlager, M. Sergon, M. Sergon, W. Sondermann, H. P. Soyer, H. P. Soyer, H. Starz, H. Starz, W. Stolz, E. Vale, E. Vale, W. Weyers, W. Weyers, A. Zink, A. Zink, E. Krieghoff-Henning, J. N. Kather, C. von Kalle, C. von Kalle, D. B. Lipka, S. Fröhling, A. Hauschild, H. Kittler, and T. J. Brinker, “Skin cancer classification via convolutional neural networks: systematic review of studies involving human experts.” *European Journal of Cancer*, 2021.

[13] M. Goyal, M. H. Yap, and S. Hassanpour, “Deep learning methods and applications for region of interest detection in dermoscopic images,” 2022.

[14] Y. Liang, C. Ge, Z. Tong, Y. Song, J. Wang, and P. Xie, “Not all patches are what you need: Expediting vision transformers via token reorganizations,” *arXiv.org*, 2022.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[16] A. Trockman and J. Z. Kolter, “Patches are all you need?” *Trans. Mach. Learn. Res.*, 2022.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *North American Chapter of the Association for Computational Linguistics*, 2019.

[18] A. Ghiasi, H. Kazemi, E. Borgnia, S. Reich, M. Shu, M. Goldblum, A. Wilson, and T. Goldstein, “What do vision transformers learn? a visual exploration,” *arXiv.org*, 2022.

[19] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. Dauphin, “Convolutional sequence to sequence learning,” *International Conference on Machine Learning*, 2017.

[20] Y.-A. Wang, Y.-A. Wang, Y.-N. Chen, and Y.-N. Chen, “What do position embeddings learn? an empirical study of pre-trained language model positional encoding,” *Conference on Empirical Methods in Natural Language Processing*, 2020.

[21] J. Ba, J. Kiros, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv.org*, 2016.

[22] A. Steiner, A. Kolesnikov, X. Zhai, R. Wightman, J. Uszkoreit, and L. Beyer, “How to train your vit? data, augmentation, and regularization in vision transformers,” *Trans. Mach. Learn. Res.*, 2021.

[23] S. Bhojanapalli, A. Chakrabarti, A. Veit, M. Lukasik, H. Jain, F. Liu, Y.-W. Chang, and S. Kumar, “Leveraging redundancy in attention with reuse transformers,” *arXiv.org*, 2021.

[24] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh, “Set transformer: A framework for attention-based permutation-invariant neural networks,” *International Conference on Machine Learning*, 2019.

[25] jeonsworld. (2020) Vit-pytorch. Accessed 28-Dec-2022. [Online]. Available: https://github.com/jeonsworld/ViT-pytorch

[26] C. Xin, Z. Liu, K. Zhao, L. Miao, Y. Ma, X. Zhu, Q. Zhou, S. Wang, L. Li, F. Yang, S. Xu, and H. Chen, “An improved transformer network for skin cancer classification.” *Computers in Biology and Medicine*, 2022.

[27] N. Park and S. Kim, “How do vision transformers work?” *arXiv preprint arXiv:2202.06709*, 2022.

[28] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626.

[29] P. Khosla, P. Teterwak, C. Wang, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, A. Maschinot, C. Liu, C. Liu, and D. Krishnan, “Supervised contrastive learning,” *Neural Information Processing Systems*, 2020.

[30] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, “Solving the multiple instance problem with axis-parallel rectangles,” *Artificial Intelligence*, 1997.

[31] O. Maron and T. Lozano-Perez, “A framework for multiple-instance learning,” *NIPS*, 1997.

[32] J. Foulds and E. Frank, “A review of multi-instance learning assumptions,” *Knowledge Engineering Review*, 2010.

[33] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, “Multiple instance learning: A survey of problem characteristics and applications,” *Pattern Recognition*, 2017.

[34] M. Ilse, J. M. Tomczak, and M. Welling, “Attention-based deep multiple instance learning,” *International Conference on Machine Learning*, 2018.

[35] X. Wang, Y. Yan, P. Tang, X. Bai, and W. Liu, “Revisiting multiple instance neural networks,” *arXiv: Machine Learning*, 2016.

[36] J. Amores, “Multiple instance classification: Review, taxonomy and comparative study,” *Artificial Intelligence*, 2013.

[37] F. Nachbar, W. Stolz, M. T, A. B. Cognetta, T. Vogt, M. Landthaler, P. Bilek, O. Braun-Falco, and G. Plewig, “The abcd rule of dermatoscopy. high prospective value in the diagnosis of doubtful melanocytic skin lesions,” *Journal of The American Academy of Dermatology*, 1994.

[38] G. Quellec, G. Cazuguel, B. Cochener, and M. Lamard, “Multiple-instance learning for medical image and video analysis,” *IEEE Reviews in Biomedical Engineering*, 2017.

[39] C. Raffel and D. P. W. Ellis, “Feed-forward networks with attention can solve some long-term memory problems,” *arXiv: Learning*, 2015.

[40] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, “A structured self-attentive sentence embedding.” *International Conference on Learning Representations*, 2017.

[41] G. Zhang, X. Shu, Z. Liang, Y. Liang, S. Chen, and J. Yin, “Multi-instance learning for skin biopsy image features recognition,” *IEEE International Conference on Bioinformatics and Biomedicine*, 2012.

[42] G. Zhang, J. Yin, Z. Li, X. Su, X. Su, G.-Z. Li, and H. Zhang, “Automated skin biopsy histopathological image annotation using multi-instance representation and learning,” *BMC Medical Genomics*, 2013.

[43] A. Astorino, A. Fuduli, M. Gaudioso, and E. Vocaturo, “Multiple instance learning algorithm for medical image classification,” *Sistemi Evoluti per Basi di Dati*, 2019.

[44] S. Andrews, I. Tsochantaridis, and T. Hofmann, “Support vector machines for multiple-instance learning,” *NIPS*, 2002.

[45] T. Mendonça, P. M. Ferreira, J. S. Marques, A. R. S. Marçal, and J. Rozeira, “Ph 2 - a dermoscopic image database for research and benchmarking,” *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2013.

[46] X. Xu and B. Li, “Evaluating multi-class multiple-instance learning for image categorization,” in *Computer Vision – ACCV 2007*, Y. Yagi, S. B. Kang, I. S. Kweon, and H. Zha, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 155–165.

[47] D. Pathak, E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional multi-class multiple instance learning,” *International Conference on Learning Representations*, 2014.

[48] X.-L. Li, “A multiclass multiple instance learning method with exact likelihood,” *arXiv: Machine Learning*, 2018.

[49] B. Kim and M. Cho, “Group-equivariant transformers without positional encoding,” 2023. [Online]. Available: https://openreview.net/forum?id=0tiMn18oNd

[50] H. Chefer, S. Gur, and L. Wolf, “Transformer interpretability beyond attention visualization,” *CoRR*, vol. abs/2012.09838, 2020. [Online]. Available: https://arxiv.org/abs/2012.09838

[51] A. Araujo, W. D. Norris, W. Norris, and J. Sim, “Computing receptive fields of convolutional neural networks,” *Distill*, 2019.

[52] M. Zaheer, S. Kottur, S. Ravanbhakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola, “Deep sets,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 3394–3404.

[53] J. Kawahara, S. Daneshvar, G. Argenziano, and G. Hamarneh, “Seven-point checklist and skin lesion classification using multitask multimodal neural nets,” *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 2, pp. 538–546, 2019.

[54] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 833–851.

[55] P. Iakubovskii, “Segmentation models pytorch,” https://github.com/qubvel/segmentation_models.pytorch, 2019.

[56] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[57] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers distillation through attention,” 2021.

[58] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, “Randaugment: Practical automated data augmentation with a reduced search space,” 2019.

[59] E. Hoffer, T. Ben-Nun, I. Hubara, N. Giladi, T. Hoefler, and D. Soudry, “Augment your batch: Improving generalization through instance repetition,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8126–8135.

[60] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, “Random erasing data augmentation,” 2017.

[61] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” 2019.

[62] R. Wightman, “Pytorch image models,” https://github.com/rwightman/pytorch-image-models, 2019.

[63] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[64] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[65] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[66] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely connected convolutional networks,” *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[67] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *CoRR*, vol. abs/1905.11946, 2019. [Online]. Available: http://arxiv.org/abs/1905.11946

A

Supplementary Information on the Proposed Method

A.1 Complementing Information Regarding Section 3.2.3.A

In this section, we mathematically demonstrate that the column-wise global average pooling method used in the first implementation of the *instance-level* approach within the three-stage MIL formulation for multi-class problems results in a bag probability distribution that sums to 1.

Consider the output of the *softmax* step (the second step in this implementation), which is denoted as $\sigma\left(z(X)\right) \in \mathbb{R}^{N\times k}$. Each row in this matrix represents the patch probability distribution across the k classes, more specifically:

$$\sigma\left(z(X)\right) = \begin{bmatrix} \sigma\left(Z_{00}\right) & \sigma\left(Z_{01}\right) & \cdots & \sigma\left(Z_{0(k-1)}\right) \\ \sigma\left(Z_{10}\right) & \sigma\left(Z_{11}\right) & \cdots & \sigma\left(Z_{1(k-1)}\right) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma\left(Z_{(N-1)0}\right) & \sigma\left(Z_{(N-1)1}\right) & \cdots & \sigma\left(Z_{(N-1)(k-1)}\right) \end{bmatrix}, \tag{A.1}$$

where $\sigma\left(Z_{ij}\right) = \frac{e^{Z_{ij}}}{\sum_{j=0}^{k-1} e^{Z_j}}$ and $\sum_{j=0}^{k-1} \sigma\left(Z_{ij}\right) = 1 \; \forall_{i=0,..,(N-1)}$. Applying the *column-wise global average pooling* operator, we obtain:

$$\begin{aligned} &\sum_{i=0}^{N-1}\left(\frac{\sigma\left(Z_{i0}\right)}{N}\right) + \sum_{i=0}^{N-1}\left(\frac{\sigma\left(Z_{i1}\right)}{N}\right) + \ldots + \sum_{i=0}^{N-1}\left(\frac{\sigma\left(Z_{i(k-1)}\right)}{N}\right) = \\ =&\frac{1}{N}\left(\sum_{i=0}^{N-1}\sigma\left(Z_{i0}\right) + \sum_{i=0}^{N-1}\sigma\left(Z_{i1}\right) + \ldots + \sum_{i=0}^{N-1}\sigma\left(Z_{i(k-1)}\right)\right) \\ =&\frac{1}{N}\sum_{i=0}^{N-1}\left(\sigma\left(Z_{i0}\right) + \sigma\left(Z_{i1}\right) + \cdots + \sigma\left(Z_{i(k-1)}\right)\right) \\ =&\underbrace{\frac{1}{N}\sum_{i=0}^{N-1}\underbrace{\left(\sum_{j=0}^{k-1}\sigma\left(Z_{ij}\right)\right)}_{1}}_{N\times 1} = 1. \end{aligned} \tag{A.2}$$

Thus, the column-wise global average pooling operator guarantees that the resulting bag probability distribution adds up to 1.

B

Appendix B: Extra Figures and Tables

B.1 Additional Information Regarding Chapter 4

This section serves as a supplemental resource to the chapter 4, providing additional tables and figures that enhance and expand the information presented in the main document.

B.1.1 Additional Information Regarding Section 4.4

Table B.1 provides an overview of the most important general configurations used in the main architectures experimented with in this thesis.

Table B.1: Models configurations. The symbols ✓ and ✗ indicate that we use and do not use the corresponding method, respectively. Most of these configurations were inspired by the DEiT model [57], specifically the data augmentation techniques, the learning rate values and scheduler, the weight decay, and the optimizer.

Methods	EViT-S	MIL Models	CNN Base.	ViT Base.
Epochs	60	100	100	60
Batch size	128	128	128	128
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate init.	2e−4	2e−4	2e−4	2e−4
Learning rate sched.	Cosine	Cosine	Cosine	Cosine
Min. learning rate	2e−6	2e−6	2e−6	2e−6
Warmup epochs	✗	5	✗	✗
Warmup learning rate	✗	1e−6	✗	✗
Weight decay	1e−6	1e−6	1e−6	1e−6
Dropout	✗	✗	✗	0.1
Data aug.	✓	✓	✓	✓
Class Weighting	✓	✓	✓	✓
Pre-trained dataset	ImageNet-1k	ImageNet-1k	ImageNet-1k	ImageNet-1k
Early stopping	✓	✓	✓	✓

B.1.1.A Additional Information Regarding Section 4.4.3

Table B.2 presents the performance results obtained for each of the baseline models when evaluated in the context of the multi-class classification problem.

Table B.2: Evaluation results of a set of baseline models on the validation set of the ISIC 2019 dataset [3–5]. The baseline models include different architectures, including RN-18, VGG-16, RN-50, DN-169, EN-B3 from the CNN-based category, and ViT-S, ViT-B, DEiT-S, DEiT-B from the Transformer-based category. The evaluation is performed for the multi-class problem.

Baseline models		ISIC 2019								
		BA	R-AK	R-BCC	R-BKL	R-DF	R-MEL	R-NV	R-SCC	R-VASC
CNNs	RN-18	76.3	65.3	81.3	75.8	75.0	73.8	88.0	65.1	86.3
	RN-50	79.8	68.2	87.7	75.2	83.3	72.9	91.5	65.1	94.1
	VGG-16	73.1	63.0	82.7	72.0	62.5	69.9	85.3	59.5	90.2
	DN-169	80.7	75.1	86.4	79.6	79.2	72.5	91.8	70.6	90.2
	EN-B3	82.2	71.1	87.8	79.4	81.3	76.5	91.6	72.2	98.0
ViTs	ViT-S	83.5	76.9	90.5	78.9	93.8	79.4	92.6	65.9	90.2
	ViT-B	84.3	80.9	91.6	74.3	91.7	78.9	89.0	69.8	98.0
	DEiT-S	83.6	72.3	90.5	82.7	87.5	80.3	92.1	65.1	98.0
	DEiT-B	84.5	75.1	92.5	79.6	87.5	79.9	94.6	70.6	96.1

B.2 Additional Information Regarding Chapter 5

In this section, we will provide additional tables that complement the results presented in chapter 5.

B.2.1 Additional Information Regarding Section 5.1

The table B.3 is a more complete version of the table 5.1 originally presented in section 5.1.

Table B.3: This table complements the results presented in table 5.1 by including additional results for the RN-50 and ViT-S baseline models and the EViT-S backbone.

	Models		ISIC 2019			$\mathbf{PH^2}$			Derm7pt		
			BA	R-MEL	R-NV	BA	R-MEL	R-NV	BA	R-MEL	R-NV
CNN	RN-50		88.9	82.6	95.1	81.9	77.5	86.3	74.4	58.7	90.1
	EN-B3		90.7	85.5	95.8	88.8	82.5	95.0	76.2	57.9	94.4
ViTs	ViT-S		91.3	86.8	95.8	88.8	80.0	97.5	71.7	50.8	92.7
	DEiT-S		91.7	86.7	96.7	86.6	75.0	98.1	74.0	53.2	94.8
EViT	Kr=0.6		91.4	86.6	96.3	88.8	80.0	97.5	73.4	52.8	94.1
	Kr=0.7		90.7	85.4	95.7	86.6	75.0	98.1	74.9	56.7	93.0
MIL-RN-50	Instance	Max	88.1	85.6	90.6	75.0	67.5	82.5	74.3	59.9	88.7
		Topk	88.5	85.3	91.8	78.8	67.5	90.0	78.3	71.0	85.6
		Avg	89.0	85.3	92.6	77.2	75.0	79.4	75.0	60.3	89.7
	Embedding	Max	89.9	84.7	95.1	85.9	77.5	94.4	73.8	57.1	90.4
		Topk	89.1	84.7	93.6	85.3	77.5	93.1	75.1	62.3	87.8
		Avg	88.9	85.6	92.2	84.1	77.5	90.6	77.5	67.1	88.0
MIL-EN-B3	Instance	Max	88.5	86.7	90.2	84.4	75.0	93.8	78.7	67.1	90.4
		Topk	89.5	85.6	93.3	89.7	82.5	96.9	77.0	60.7	93.2
		Avg	89.1	86.7	91.6	85.0	77.5	92.5	76.5	64.3	88.7
	Embedding	Max	86.0	85.7	86.4	85.3	77.5	93.1	76.1	66.3	85.9
		Topk	89.2	85.7	92.7	88.1	82.5	93.8	76.7	59.5	93.9
		Avg	89.1	84.4	93.8	83.4	70.0	96.9	75.3	56.3	94.3
MIL-EViT-S	Instance	Max	90.6	86.7	94.4	81.2	72.5	90.0	73.6	54.0	93.2
		Topk	90.8	86.0	95.7	81.3	65.0	97.5	73.0	52.4	93.6
		Avg	91.5	86.9	96.0	84.1	70.0	98.1	73.4	52.4	94.4
	Embedding	Max	90.9	86.0	95.8	82.8	70.0	95.6	74.2	54.4	94.1
		Topk	91.1	86.0	96.3	88.4	80.0	96.9	73.1	50.8	95.5
		Avg	91.3	86.6	96.0	80.9	70.0	91.9	73.4	54.4	92.5
MIL-DEiT-S	Instance	Max	91.7	87.1	96.3	87.2	77.5	96.9	74.4	54.8	94.1
		Topk	91.4	86.6	96.2	84.1	72.5	95.6	71.8	49.6	94.1
		Avg	91.8	87.5	96.1	87.8	80.0	95.6	74.8	56.0	93.7
	Embedding	Max	91.0	87.4	94.5	85.3	75.0	95.6	74.7	58.3	91.1
		Topk	91.5	86.9	96.1	89.7	80.0	99.4	75.1	55.2	95.1
		Avg	91.4	87.4	95.4	85.0	75.0	95.0	76.0	61.1	91.0

B.3 Additional Visualizations Regarding Section 5.3

In this section of the appendix B, we include additional images that visually illustrate the process performed by both branches of the proposed framework.

B.3.1 Additional Visualizations Regarding Section 5.3.1

Figure B.1 provides a visual representation of the operations performed by the EViT architecture. It provides a clear illustration of the patch removal process within the EViT architecture. In addition, this figure presents a heat map showing the "importance" of individual patches in influencing the model's predictions. It also shows the corresponding gradient information that describes this process.

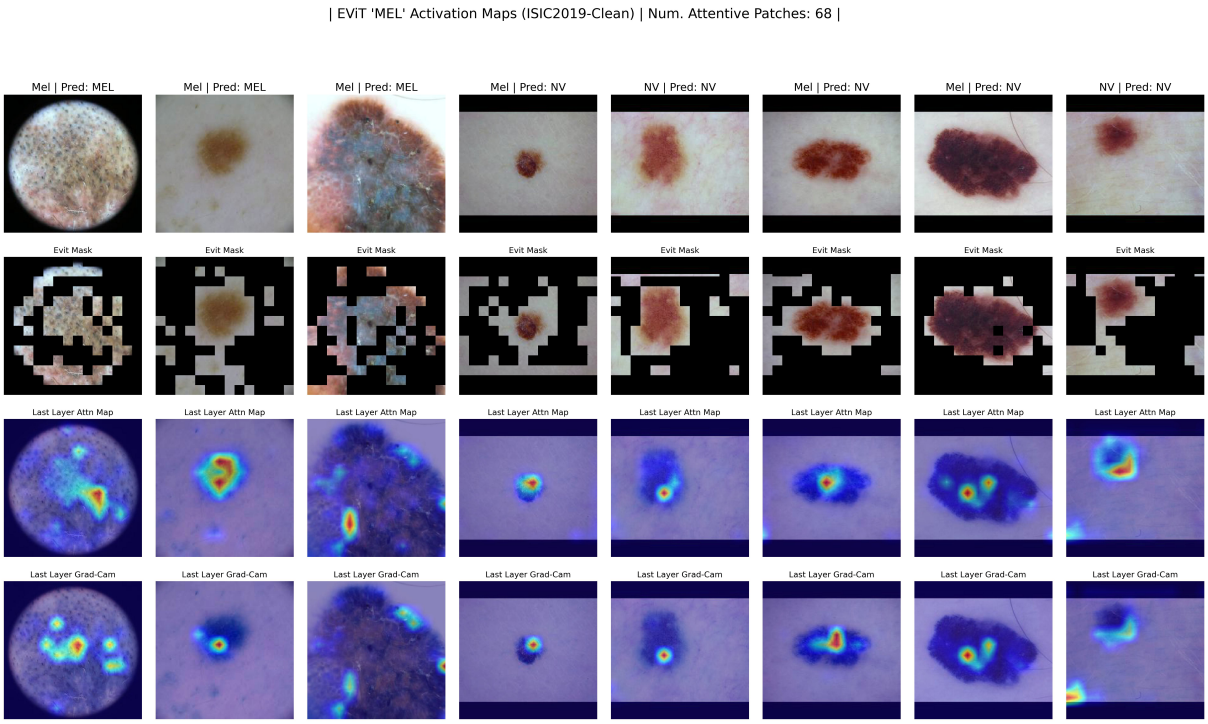

Figure B.1: Visualization of the EViT architecture process with Kr $= 0.7$ and the default configuration for the placement of token reorganization blocks. The first row displays input images from the ISIC 2019 validation dataset [3–5]. The second row illustrates the inattentive patch removed by the EViT model. The third row exhibits the attentiveness of the CLS token in the final encoder block. The heatmap is created by reshaping the first row of the attention map $\mathbf{A}$, represented as $\boldsymbol{a} \in \mathbb{R}^{1\times N}$, into a 14×14 heatmap. This heatmap is then transformed into a $224 \times 224 \times 3$ representation using bilinear interpolation. The last row presents the gradient-based visualization of CLS embedding attentiveness, inspired by Chefer *et al.* [50].

Figures B.2 and B.3 illustrate the visualization processes performed by the EViT branch **with** and **without** positional encoding, respectively.

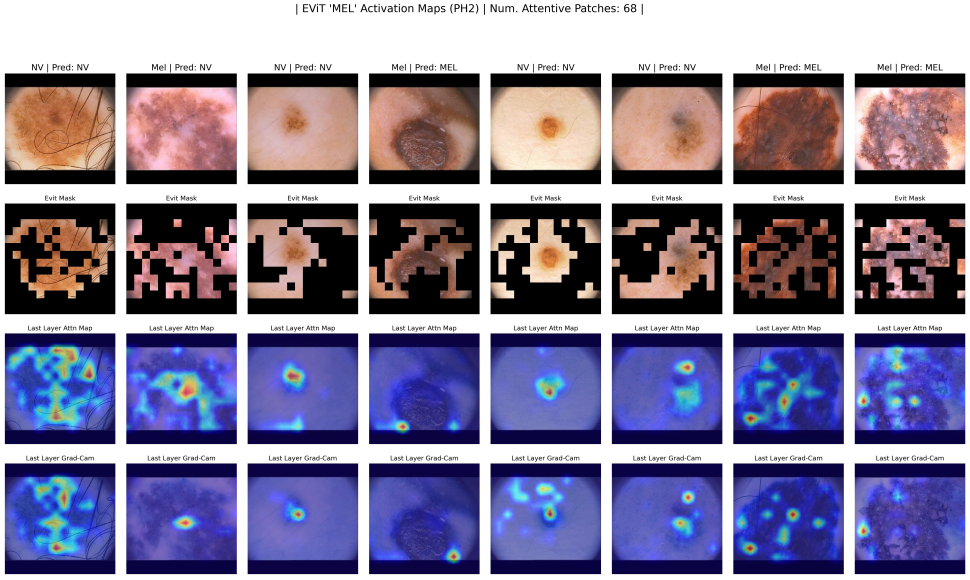

Figure B.2: Visualization of the EViT-S architecture process **with** positional encoding. The model is configured with Kr $= 0.7$ and the default configuration for the placement of token reorganization blocks. The first row displays input images from the PH^2 test set [45]. The second row illustrates the inattentive patch removed by the EViT model. The third row exhibits the attentiveness of the CLS token in the final encoder block. The last row presents the gradient-based visualization of CLS embedding attentiveness.

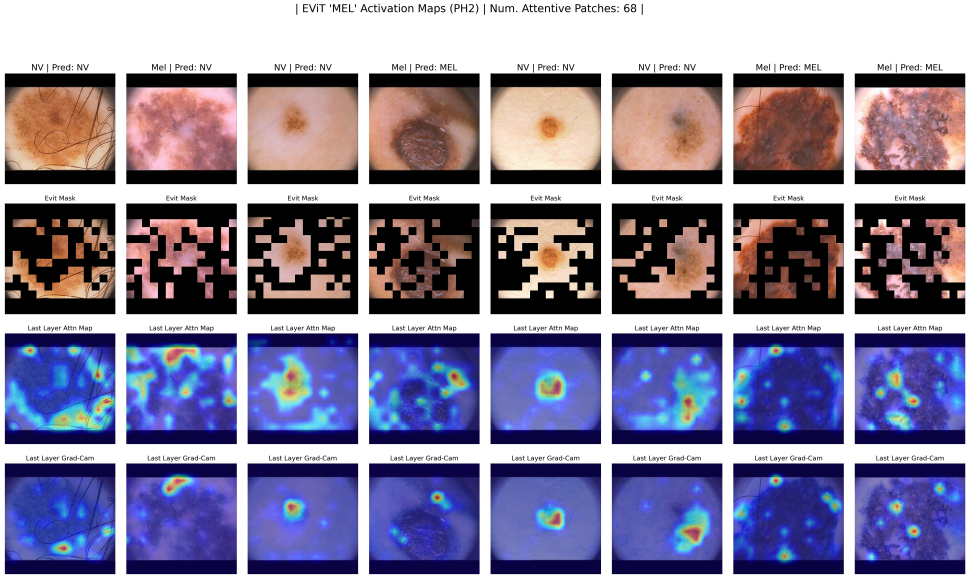

Figure B.3: Visualization of the EViT-S architecture process **without** positional encoding. The model is configured with Kr $= 0.7$ and the default configuration for the placement of token reorganization blocks. The first row displays input images from the PH^2 test set [45]. The second row illustrates the inattentive patch removed by the EViT model. The third row exhibits the attentiveness of the CLS token in the final encoder block. The last row presents the gradient-based visualization of CLS embedding attentiveness.

B.3.2 Additional Visualizations Regarding Section 5.3.2

In this section of the appendix B, we include additional images that visually illustrate the process performed by the MIL branch. These images provide further insight into the processes described in section 4.4.2.

Figure B.4 provides a visual representation of the different MIL branch visualizations for the binary melanoma (MEL) versus nevus (NV) problem. Specifically, this figure highlights the heatmaps generated by the *instance-level* MIL model using the **max pooling** operator. The first row shows the input images, followed by the second row, which shows the patch probability heatmap for the melanoma (MEL) class. The third row shows the gradient heatmap for each patch, and the last row shows the identification of the *key patch* as detected by the corresponding MIL model.

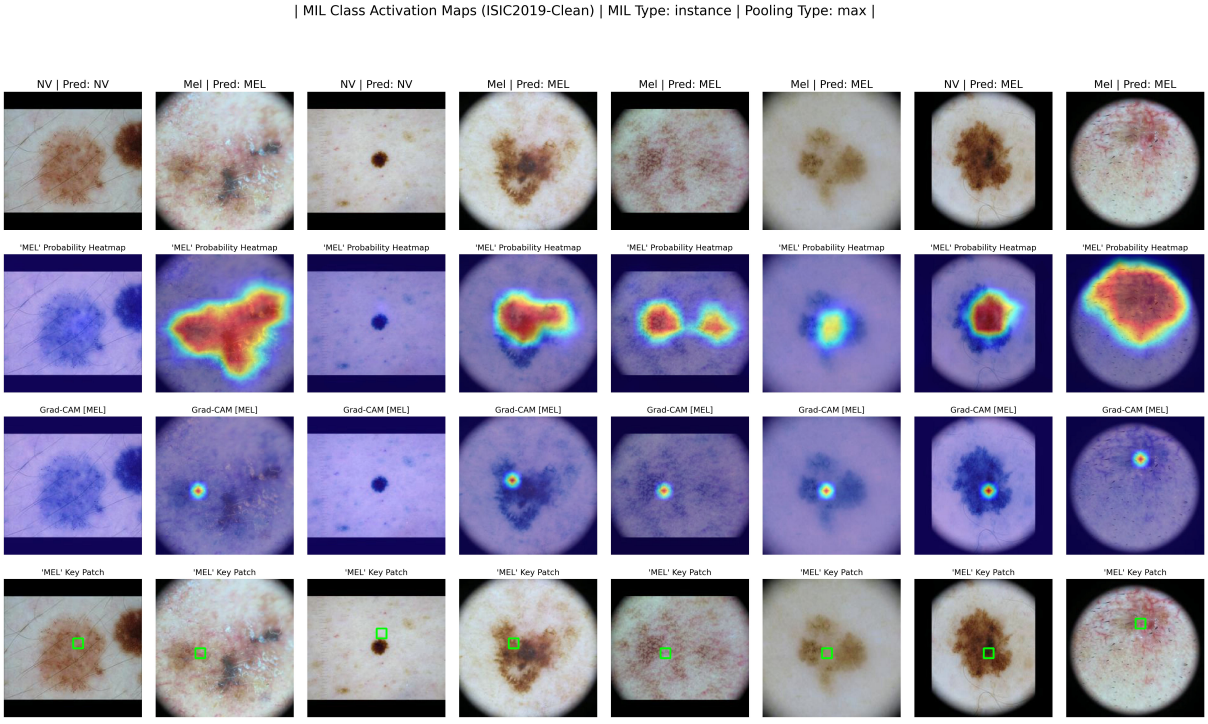

Figure B.4: Visualization of the MIL branch processes, specifically the *instance-level* MIL model using **max pooling**. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The Figure shows the input images in the first row, followed by the patch probability heatmap for the melanoma (MEL) class in the second row. The third row shows the gradient heatmap for each patch, while the last row shows the identification of the *key patch* as detected by the corresponding MIL model.

Figure B.5 provides a visual representation of the different visualizations produced by the *instance-level* MIL model using the **top-k average pooling** operator. In this case, the last row shows the gradients associated with the patches classified as nevus (NV).

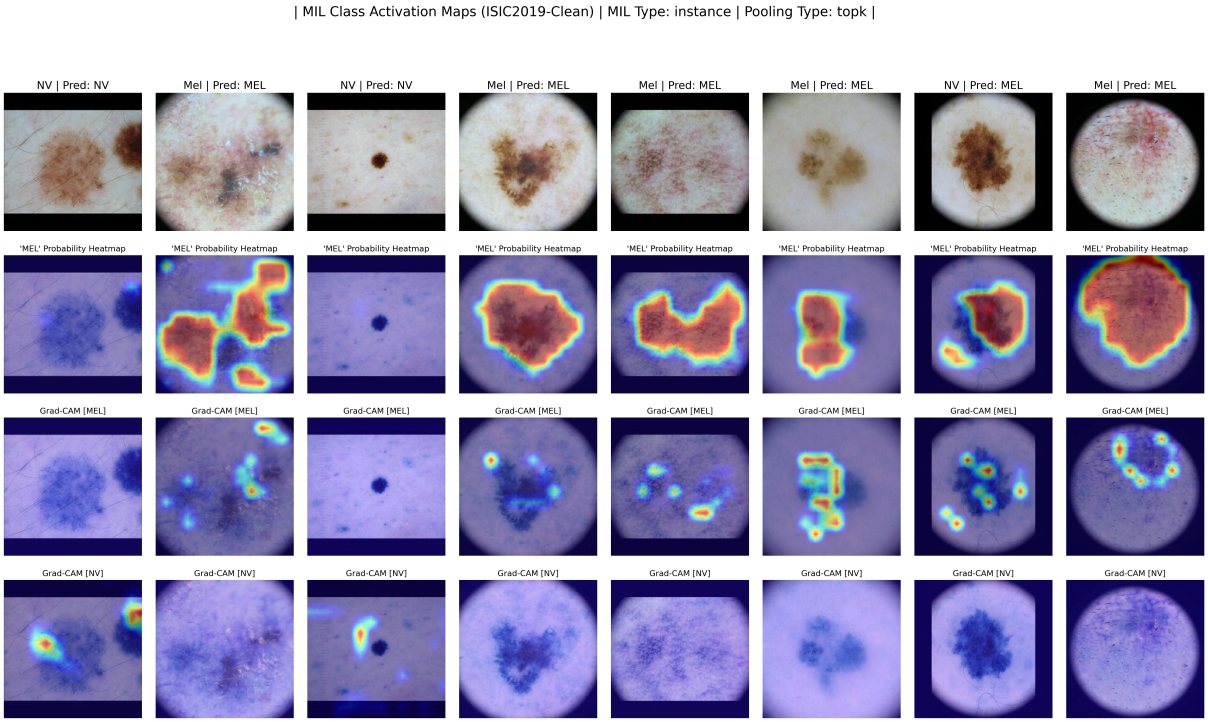

Figure B.5: Visualization of the MIL branch processes, specifically the *instance-level* MIL model using the **top-k average pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The Figure shows the input images in the first row, followed by the patch probability heatmap for the melanoma (MEL) class in the second row. The third row shows the gradient heatmap for each patch. In this case, the last row shows the gradients with respect to the patches that the model predicted to be nevus (NV).

Figure B.6 provides a visual representation of the different visualizations produced by the *instance-level* MIL model using the **average pooling** operator.

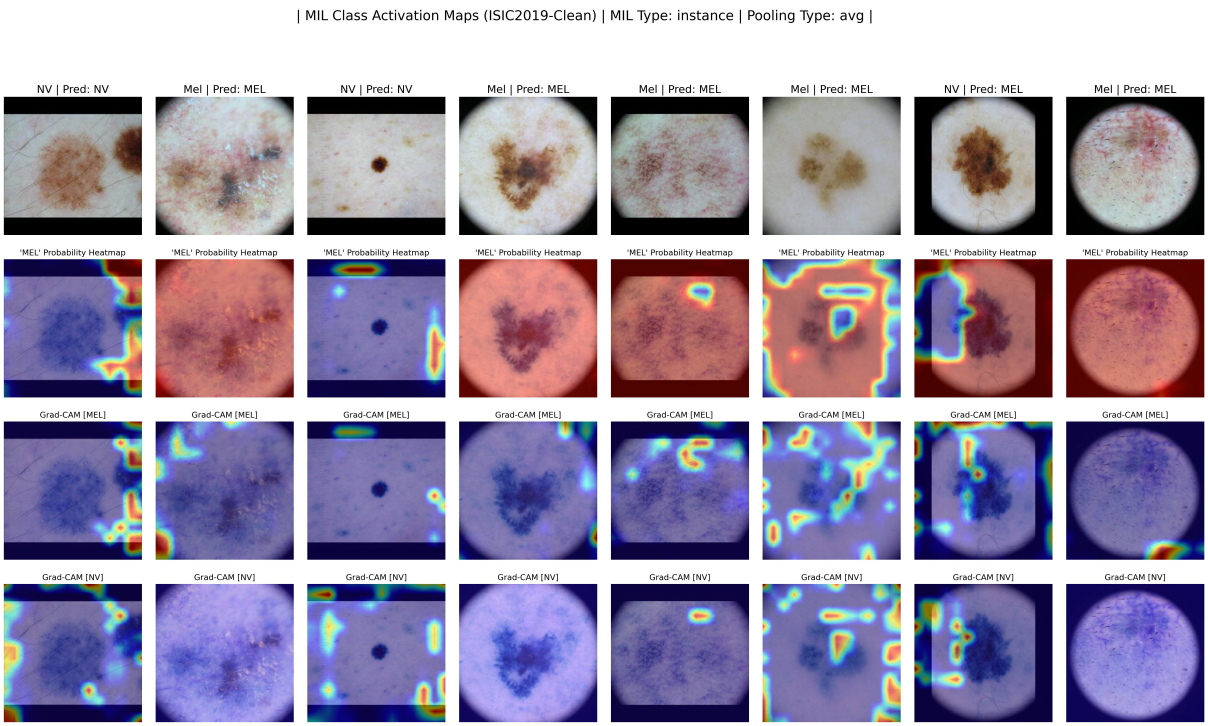

Figure B.6: Visualization of the MIL branch processes, specifically the *instance-level* MIL model with the **average pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The Figure shows the input images in the first row, followed by the patch probability heatmap for the melanoma (MEL) class in the second row. The third row shows the gradient heatmap for each patch. In this case, the last row shows the gradients with respect to the patches that the model predicted to be nevus (NV). It's important to note that the gradient-based visualization method used in these images is inspired by the work of Chefer *et al.* [50] and differs from the standard Grad-Cam approach.

Figure B.7 visually represents the different visualizations generated by the *instance-level* MIL model using the **masked max pooling** operator. It is worth noting that the **max pooling** results shown in Figure B.4 often match those produced by **masked max pooling** in Figure B.7. This suggests that the MIL framework can effectively identify *key patches* without the need for additional domain-specific knowledge.

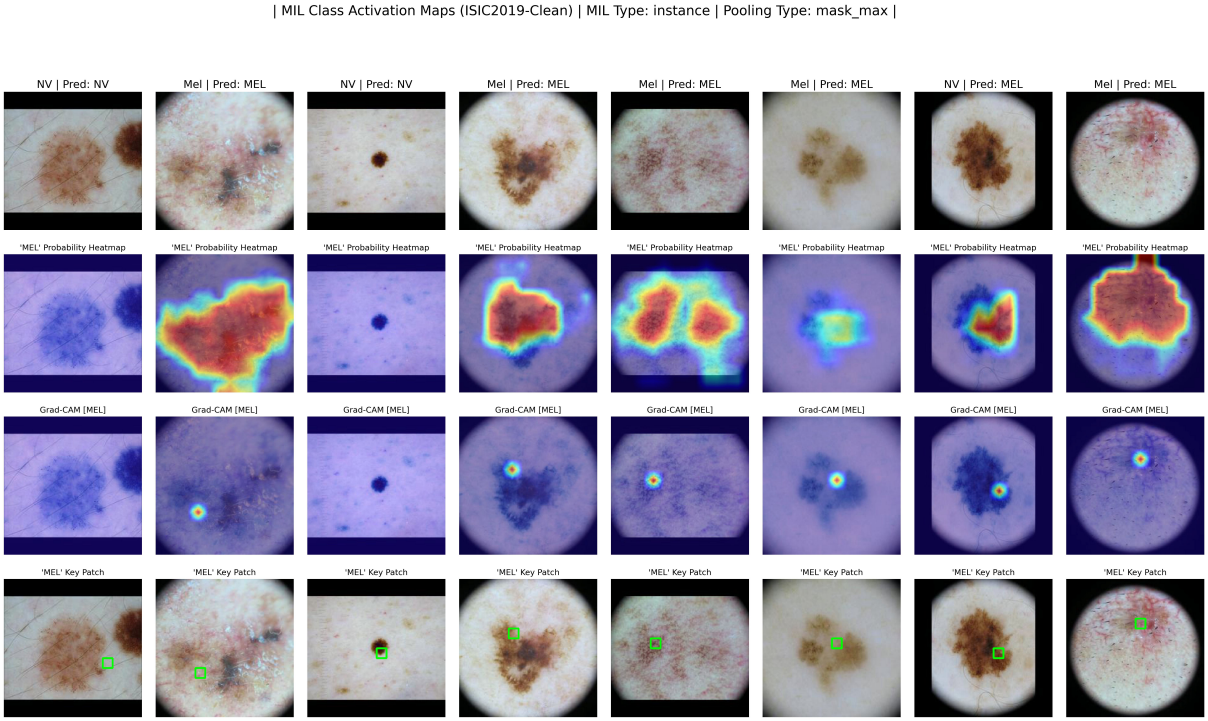

Figure B.7: Visualization of the MIL branch processes, specifically the *instance-level* MIL model using the **masked max pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The Figure shows the input images in the first row, followed by the patch probability heatmap for the melanoma (MEL) class in the second row. The third row shows the gradient heatmap for each patch. In this case, the last row shows the *key patches* identified by the MIL model.

Figure B.8 visually represents the different visualizations generated by the *instance-level* MIL model using the **masked average pooling** operator.

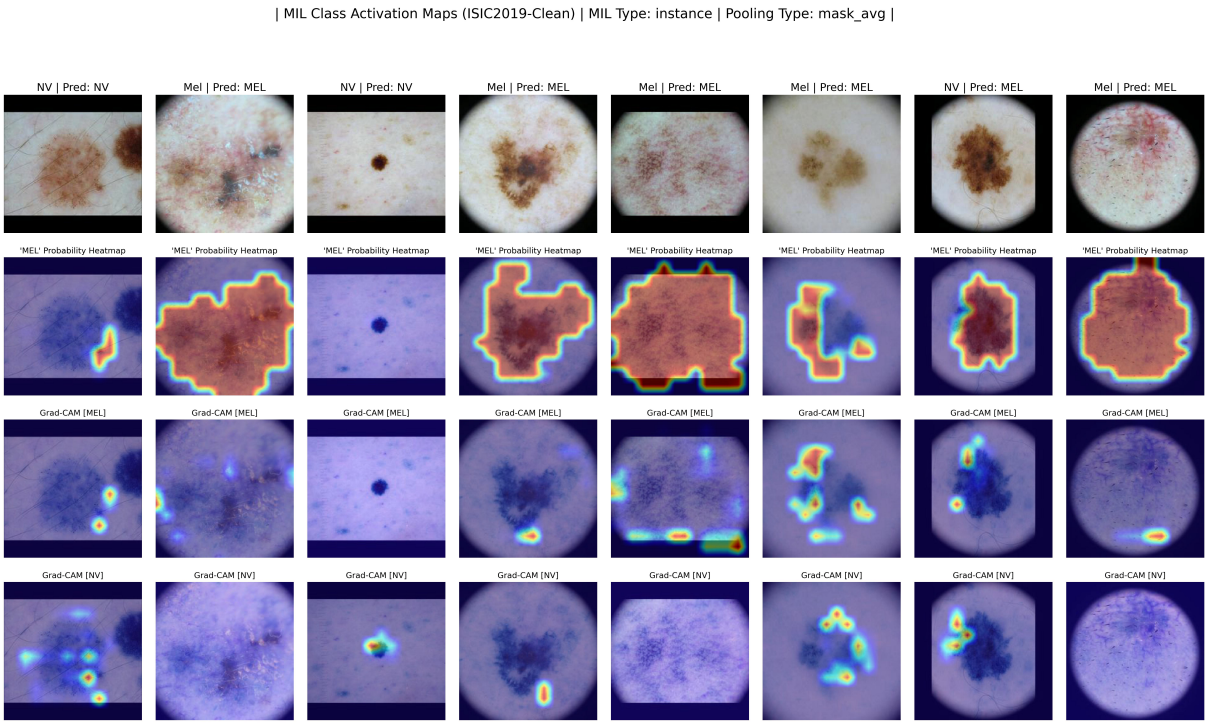

Figure B.8: Visualization of the MIL branch processes, specifically the *instance-level* MIL model with the **masked average pooling** operator. The RN-18 serves as the backbone for this MIL model. The images shown in this figure are from the validation set of the ISIC 2019 dataset [3–5] and are associated with the binary classification task of melanoma (MEL) versus nevus (NV). The figure shows the input images in the first row, followed by the patch probability heatmap for the melanoma (MEL) class in the second row. The third row shows the gradient heatmap for each patch. In this case, the last row highlights the gradients for patches predicted by the model to be nevus (NV). Notably, the gradient-based visualization method used in these images is inspired by Chefer *et al.* [50] and differs from the standard Grad-Cam approach.

Figure B.9 shows the different visualizations generated by the *embedding-level* MIL model using the **column-wise global max pooling** operator. In the *embedding-level* cases, we used Grad-Cam approaches exclusively. Specifically, for the **column-wise global max pooling** operator, we used a Grad-Cam approach inspired by the work of Chefer *et al.* [50]. It is important to note that for the bag of patch embeddings, denoted as $X \in \mathbb{R}^{N \times D}$, we selected the maximum values for each feature along the embedding dimension D, resulting in a tensor $X \in \mathbb{R}^{D}$.

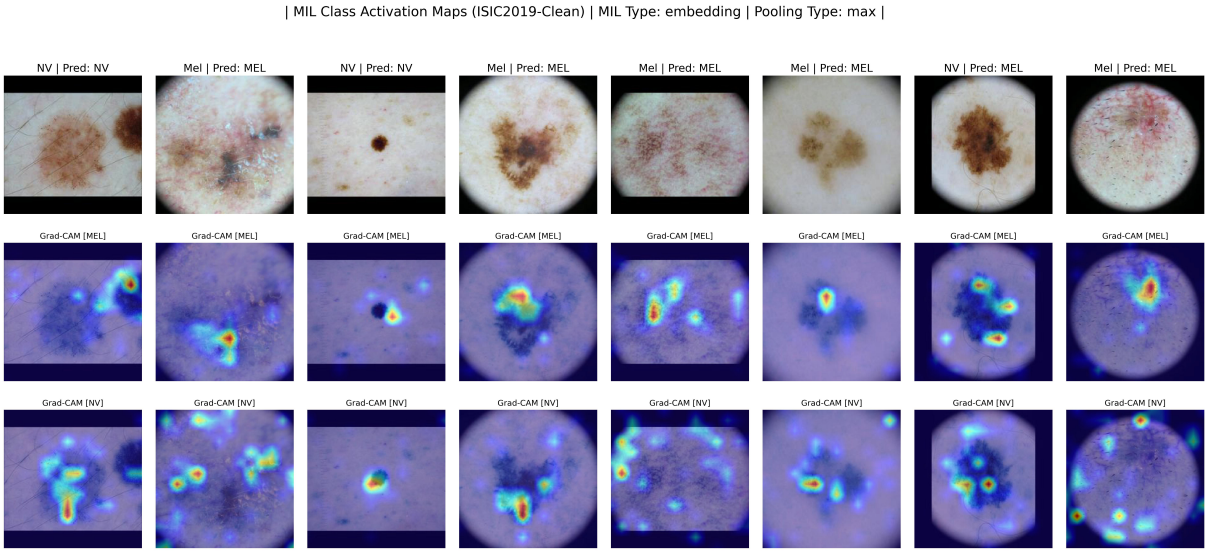

Figure B.9: Visualization of the MIL branch processes, specifically the *embedding-level* MIL model using the **column-wise global max pooling** operator. The backbone used for this MIL model is the RN-18. The images used for visualization are taken from the validation set of the ISIC 2019 dataset [3–5] and refer to the binary classification task of melanoma (MEL) vs. nevus (NV). The first row of the figure shows the input images, while the second and third rows show Grad-Cam [50] visualizations for the melanoma (MEL) and nevus (NV) classes, respectively. It is worth noting that these images use a gradient-based visualization method inspired by the work of Chefer *et al.* [50], which distinguishes them from the standard Grad-Cam technique.

Figure B.10 visually represents the different visualizations generated by the *embedding-level* MIL model using the **column-wise global top-k average pooling** operator.

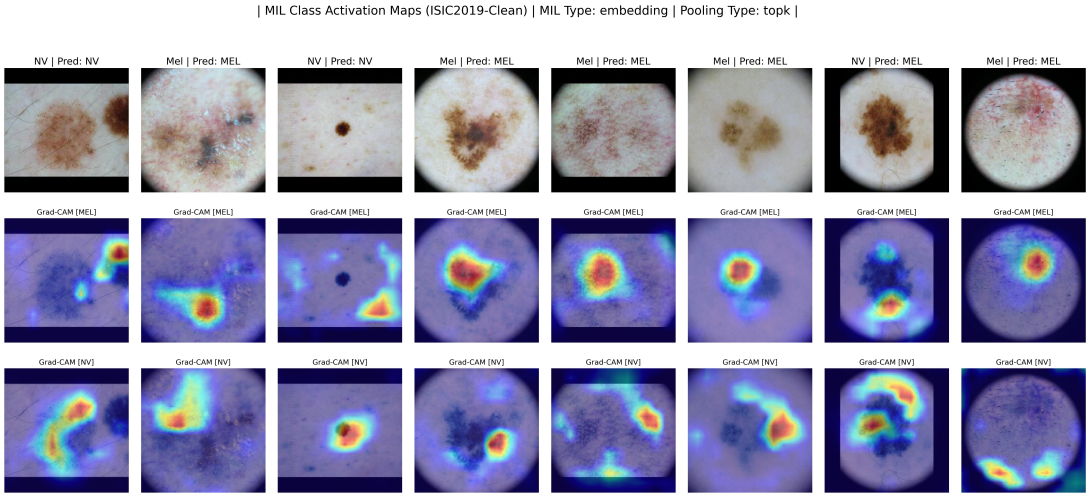

Figure B.10: Illustration of different visualizations using the *embedding-level* MIL model with the **column-wise global top-k average pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The first row of the figure shows the input images, while the second and third rows illustrate Grad-Cam [28] visualizations for the melanoma (MEL) and nevus (NV) classes, respectively. Note that in this case we used the standard Grad-Cam algorithm [28].

Figure B.11 visually represents the different visualizations generated by the *embedding-level* MIL model using the **column-wise global average pooling** operator.

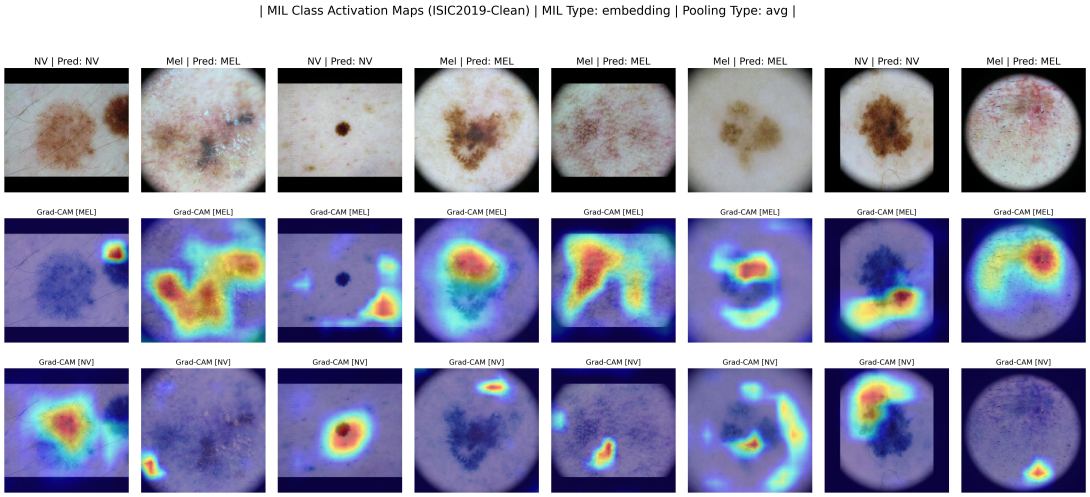

Figure B.11: Illustration of different visualizations using the *embedding-level* MIL model with the **column-wise global average pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The first row of the figure shows the input images, while the second and third rows illustrate Grad-Cam [28] visualizations for the melanoma (MEL) and nevus (NV) classes, respectively. Note that in this case, we used the standard Grad-Cam algorithm [28].

Figure B.12 visually represents the different visualizations generated by the *embedding-level* MIL model using the **column-wise global masked pooling** operator.

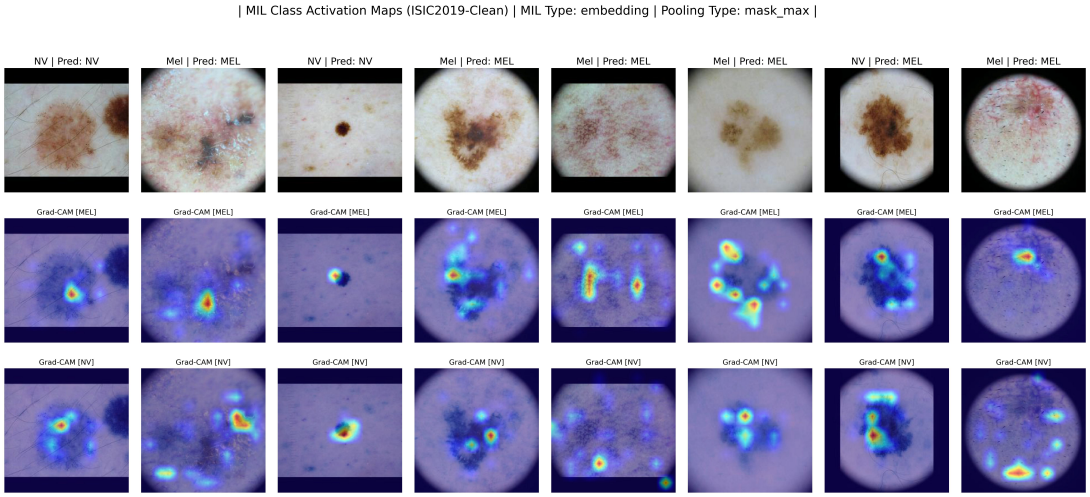

Figure B.12: Illustration of different types of visualizations using the *embedding-level* MIL model with the **column-wise global masked average pooling** operator. The backbone used for the MIL model is the RN-18. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The first row of the figure shows the input images, while the second and third rows illustrate Grad-Cam [50] visualizations for the melanoma (MEL) and nevus (NV) classes, respectively. Note that the gradient-based visualization method used in these images is inspired by the work of Chefer *et al.* [50].

Figure B.13 visually represents the different visualizations generated by the *embedding-level* MIL model using the **column-wise global masked average pooling** operator.

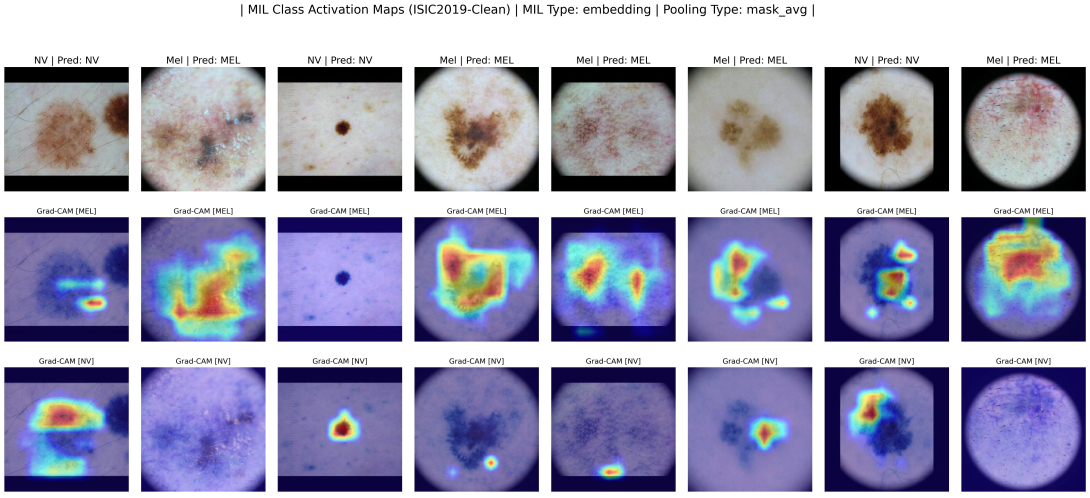

Figure B.13: Illustration of different visualizations using the *embedding-level* MIL model with the **column-wise global masked average pooling** operator. The images are taken from the validation set of the ISIC 2019 dataset [3–5], and belong to the binary problem of melanoma (MEL) vs. nevus (NV). The first row of the figure shows the input images, while the second and third rows illustrate Grad-Cam [28] visualizations for the melanoma (MEL) and nevus (NV) classes, respectively.

Figure B.14 serves as an additional example for Figure 5.5 present in section 5.3.

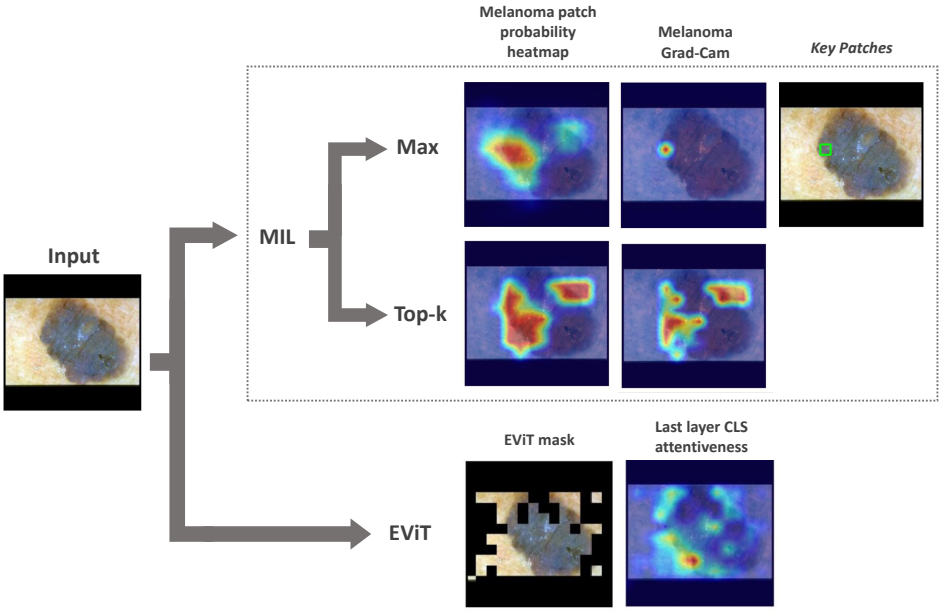

Figure B.14: Visualization of the ROIs identified by the two branches of the proposed framework. This Figure shows two MIL models, one consisting of a *instance-level* MIL model using the *max* pooling operator, and the other using the *top-k average* pooling operator. The model using the *top-k average* pooling operator used $k = 25\%$, which corresponds to selecting the 49 patches with the highest probability of melanoma. Both MIL models use the EN-B3 architecture as a backbone. The EViT is defined according to the configuration set in section $4.4.1$, has Kr $= 0.7$ and is set **without** positional encoding. In the MIL branch, we first see the melanoma (MEL) patch probability heatmap. Then, by visualizing the gradients with respect to the MEL class, we obtain the ROIs. In the EViT branch, we see first the application of the EViT mask, and then the attention weights with respect to the CLS token on the last EViT layer. This image was sampled from the Derm7pt test set [53].