CHAPTER 22

Deep multiple instance learning for digital histopathology

Maximilian Ilse, Jakub M. Tomczak, Max Welling
University of Amsterdam, Amsterdam Machine Learning Lab, Amsterdam, the Netherlands

Contents

Handbook of Medical Image Computing and Computer Assisted Intervention
https://doi.org/10.1016/B978-0-12-816176-0.00027-2

Nowadays, a typical benchmark image data sets contain thousands of images of size up to 256×256 pixels. Current hardware and software allow us to easily parallelize computations and efficiently train a machine learning model. However, in medical imaging only a small number of images is available for training (10^1–10^2 of medical scans) and an image consists of billions of pixels (roughly $10{,}000 \times 10{,}000$ pixels). Moreover, very often only a single label for one image is available. Therefore, a naturally arising question is how to process such large images and learn from weakly-labeled training data. A possible solution is to look for local patterns and combine them into a global decision. Opposite to the classical supervised learning, where one label corresponds to one image, we consider now a situation with one label for a collection (a *bag*) of multiple images (*instances*). We can handle a large image by processing all smaller images in parallel, in a similar manner how a minibatch is processed.

In machine learning the problem of inferring a label for a bag of i.i.d. instances is called the *multiple instance learning* (MIL). The main goal of MIL is to learn a model that predicts a bag label (e.g., a medical diagnosis). An additional task is to find the instances that trigger the bag label a.k.a. *key instances* [17]. Discovering key instances is of special interest due to legal issues. According to the European Union General Data Protection Regulation (taking effect 2018), a user should have the right to obtain an explanation of the decision reached. In order to solve the primary task of a bag classification, different methods are proposed, such as utilizing similarities among bags [4], embedding instances to a compact low-dimensional representation that is further fed to a bag-level classifier [1,2], and combining responses of an instance-level classifier [19,20,30]. From these three approaches only the last one could provide interpretable results. However, it was shown that the instance level accuracy of such methods is low [11], and in general there is a disagreement among MIL methods at the instance level [3]. All these issues force us to rethink the usability of current MIL models for interpreting the final decision.

In this chapter, we aim at explaining the idea of the multiple instance learning and show its natural fit in medical imaging illustrated by the example of the computational pathology. We formally define the MIL problem and outline a theoretical prescription of formulating MIL methods in Sect. 22.3. Next, we present different MIL approaches in Sect. 22.4 and then discuss components of MIL models in Sect. 22.5. Eventually, we present the application of MIL to histopathology data in Sect. 22.6.

22.1. Multiple instance learning

The multiple instance learning framework was originally introduced by [7]. That paper deals with the problem of predicting the drug activity of molecules. Most drugs are small molecules that work by binding to much larger molecules such as enzymes and cell surface receptors. Each drug molecule can adopt different shapes by rotating its bonds, which are called conformations. A drug molecule is labeled "active" if at least one of

its conformations can bind to a binding site. In case of an "inactive" molecule none of its possible conformations can bind to a binding site. Here, a single conformation of a molecule is referred to as an instance and all conformations of a certain molecule are referred to as a bag. The only available observation is if a molecule is "active" or "inactive". In the paper, each conformation was represented by 166 shape features and a bag could contain up to hundreds of conformations. The task was to infer drug activity of unseen molecules. This seminal paper formulated a new problem of classifying sets of objects, called multiple instance learning (MIL).

Andrews et al. [1] proposed a support vector machine based MIL model for the automatic annotation of images and text. Here an image consists of a set of patches and a text consists of a set of words. Each patch or word is referred to as an instance and the image or text is referred to as a bag, respectively. Considering image and text data sets from the MIL perspective pointed out its major advantage, namely, the ability of working with *weakly annotated* data. Annotating whole images is far less time consuming than providing pixel-level annotations. The same kind of reasoning applies to text data as well. In general, documents that contain a relevant passage are considered to be relevant with respect to a specific topic, however, most of the time class labels are not available on the passage level. They are rather associated with the document as a whole. As in [7] the models are optimized using precomputed features, such as, color, texture, and shape features in case of the image data sets and features related to word frequency in case of the text data sets. Furthermore, they were among the first to investigate two different approaches of predicting bag labels. The first approach tries to first predict a label for each instance in a bag. Afterwards these instance labels are used to infer the corresponding bag label. The second approach does not predict instance labels but aims at predicting the bag label directly. In Sect. 22.4, we will discuss these two approaches in greater detail.

In the following years, a variety of extensions of these methods were proposed with a steadily improving performance on a variety of MIL data sets. Ranging from methods based on graphs, where each bag is represented by a graph in which instances correspond to nodes [12], to methods which convert the multiple instance learning problem to a standard supervised learning problem by embedded instance selection [2].

In addition to the classical MIL assumption described above that is discussed in detail in Sect. 22.3, various new MIL assumptions were proposed. For example, instead of single instances that trigger the bag label, there could be a setting where most, if not all, instances contribute to the bag label. In such a scenario, utilizing (dis)similarities among bags instead of instance features is favorable [4]. Even though this approach has certain advantages, it is not necessarily well suited for the field of medical imaging, as we will discuss in Sect. 22.4.

Before we move on to discuss the use of deep neural networks in MIL, we have to highlight another key aspect of MIL. In many real life application we are not only

interested in inferring the labels of before unseen bags, we are also concerned with finding the instances that are responsible for the bag label. These instances are called *key instances*. Being able to point out key instances adds a great deal of interpretability to an MIL model. Moreover, it has been shown that a model that is successfully detecting key instances is more likely to achieve better bag label predictions [17].

22.2. Deep multiple instance learning

Before the advent of deep neural networks, the majority of machine learning systems consisted of two separated entities: a feature extractor and a classifier. A crucial step in the design of such systems was the extraction of discriminant features from the given data. This process was done manually by human researchers, therefore we speak of *handcrafted features*. After the extraction of those features, they were subsequently fed into statistical classifiers, e.g., support vector machines, random forests, and Gaussian processes. The classifier was then trained in a fully supervised manner using a labeled training set.

The clear advantage of deep neural network is that they can be trained from end-to-end. In other words, deep neural networks are able to learn the features that optimally represent the given training data. This concept lies at the basis of many deep learning algorithms: networks composed of many layers that find a mapping from the input space (e.g., images) to the output space (e.g., class label) while learning increasingly higher level features.

Convolutional neural networks (CNNs) are a class of deep neural networks which have been widely applied to image data. CNNs are using (small) convolutional filters to extract local features of an image. Hereby, CNNs are exploiting a key property of images, which is that nearby pixels are more strongly correlated than more distant pixels. As a result, CNNs are more robust to transformations such as translating, rotating, scaling, and elastic deformations than fully connected networks (FCNs). Currently, there is a fast developing research of making CNNs invariant to rotations and other group-theoretic properties [5], with successful applications to medical imaging [25].

As with computer vision, CNNs have become the standard technique for feature extractions in MIL. In contrast to a fully supervised setting, the challenge is to design a system that can be trained end-to-end without instance labels. In other words, the information represented by the bag label has to be backpropagated through the entire network. Deep neural network architectures that are applicable for MIL can be found in [27]. Standard CNNs consist of 3 types of layers: convolutional layers, fully connected layers, and pooling layers. In classical supervised learning, pooling layers are used to reduce the dimensions of the latent space after every layer of neurons. In MIL problem, the pooling layers are also used to pool instance representations to obtain bag representations (i.e., pooling over instances). As we will see in Sects. 22.4 and 22.5, there is a

wide variety of architectures and MIL pooling layers that we can choose from to design a deep MIL model.

22.3. Methodology

In (binary) fully supervised learning, one tries to find a mapping from an instance $\mathbf{x} \in \mathbb{R}^D$ to a label $y \in \{0, 1\}$, whereas in MIL one tries to find a mapping from a bag of instances $X = \{\mathbf{x_1}, \ldots, \mathbf{x}_K\}$ to a label $Y \in \{0, 1\}$. It is important to notice that the number of instances K in a bag is not necessarily constant for all bags in X. In MIL we assume that instances in a bag are unordered and independent of each other. Furthermore, we assume that there is a binary label for each instance in a bag, i.e., $y_1, \ldots, y_K$, $y_k \in \{0, 1\}$ for all $k = 1, \ldots, K$, though during training we have no access to these instance labels.

We now can define the main assumption of MIL as follows:

$$Y = \begin{cases} 0, & \text{if and only if } \sum_{k=1}^{K} y_k = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{22.1}$$

Since our label Y is a binary random variable, we use Bernoulli distribution to model the probability of Y given the bag of instances X:

$$p(Y|X) = S(X)^Y (1 - S(X))^{(1-Y)}, \tag{22.2}$$

where $S(X) = p(Y = 1|X)$ is a scoring function of a bag X.

In order to train the scoring function, we utilize the negative log-likelihood of Bernoulli distribution in (22.2), which yields

$$\frac{1}{K} \sum_{k=1}^{K} -\log p(Y_k|X_k) = \frac{1}{k} \sum_{k=1}^{K} -Y_k \log(S(X_k)) - (1 - Y_k) \log(1 - S(X_k)), \tag{22.3}$$

where X_k and Y_k denote the training pair of a bag and a label, respectively.

In the following, we will introduce a framework to construct scoring functions $S(X)$ that successfully map a bag of instances $X = \{\mathbf{x_1}, \ldots, \mathbf{x}_K\}$ to a label $Y \in \{0, 1\}$. Since the instances in a bag are unordered and independent of each other, a valid scoring function has to be permutation invariant by design. In general, a scoring function $S(X)$ is considered permutation invariant (a.k.a. a symmetric function) if and only if

$$S(\{\mathbf{x}_1, \ldots, \mathbf{x}_K\}) = S(\{\mathbf{x}_{\sigma(1)}, \ldots, \mathbf{x}_{\sigma(K)}\}), \tag{22.4}$$

for any permutation σ.

The following two theorems provide sufficient and necessary conditions of defining a permutation invariant scoring function. The difference between Theorems 22.1

and 22.2 is that the former is a universal decomposition while the latter provides an arbitrary approximation.

Theorem 22.1. ([29]) *A scoring function for a set of instances X, $S(X) \in \mathbb{R}$, is a symmetric function (i.e., permutation invariant to the elements in X) if and only if it can be decomposed in the following form:*

$$S(X) = g\Big(\sum_{\mathbf{x}\in X} f(\mathbf{x})\Big), \tag{22.5}$$

where f and g are suitable transformations.

Theorem 22.2. ([18]) *For any $\epsilon > 0$, a Hausdorff continuous symmetric function $S(X) \in \mathbb{R}$ can be arbitrarily approximated by a function in the form $g(\max_{x\in X} f(x))$, where* max *is the elementwise vector maximum pooling function and f and g are continuous functions, that is,*

$$|S(X) - g(\max_{x\in X} f(x))| < \epsilon. \tag{22.6}$$

From Theorems 22.1 and 22.2 we can see how one can design an algorithm to approximate any permutation-invariant scoring function $S(X)$:

1. Embedding all instances into a low-dimensional space using the function f.
2. Combining the embedded instances using a permutation-invariant (symmetric) function, e.g., the sum and max of embedded instances as shown in Eqs. (22.5) and (22.6).
3. Mapping of the combination of embedded instances to a single scalar (the score) using the function g.

Here, the choices of g and f are of crucial importance for the performance of a MIL model. Therefore, in the following, we presume that the functions f and g are parameterized by deep neural networks. Since, in theory, deep neural networks can approximate any nonlinear function.

22.4. MIL approaches

In MIL literature, three approaches prevail, namely: instance-based approach, embedding-based approach, and bag-based approach. In the following, we will explain each of them in detail. Later we will show that there are models that are not necessarily restricted to one of the three approaches. In all cases we will refer to the functions $S(\cdot)$, $f(\cdot)$, and $g(\cdot)$ as introduced in Sect. 22.3.

22.4.1 Instance-based approach

When using the instance-based approach, we try to directly infer instance scores. Consequently, we train a deep neural network, that is shared among instances, to compute

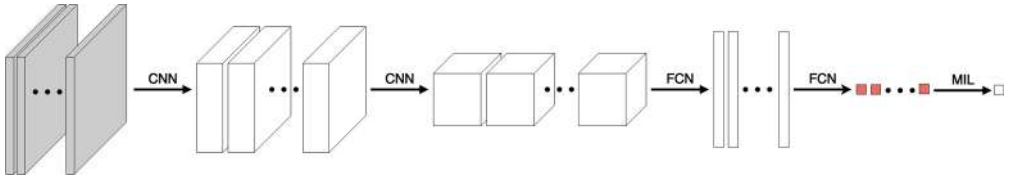

Figure 22.1 Instance-level approach: For each instance in a bag an instance score is obtained using a combination convolutional and fully connected layers. At last, an MIL pooling layer is used to infer the bag label.

a score (a scalar value between 0 and 1) for every instance. In a second step, an MIL pooling layer combines the score for every instance and computes a label for the entire bag of instances. Fig. 22.1 shows a possible architecture for an end-to-end trainable deep neural network that is using the instance-based approach.

The advantage of the instance-based approach is its ability to highlight key instances. This makes the approach highly interpretable, since a practitioner can investigate the highlighted instances, i.e., instances with a high instance score. When compared to the embedding-based approach (Sect. 22.4.2) and the bag-based approach (Sect. 22.4.3), multiple studies showed that the instance-based approach results generally in worse classification performance [27]. Since the instance labels are unknown during training, the deep neural network predicting instance scores might be trained insufficiently and introduces an additional error to the bag label prediction. In case of the instance-based approach, f is parameterized by a deep neural network and g is the identity function (see Eq. (22.5)). In Sect. 22.5 an overview of MIL pooling functions is given. The majority of the presented MIL pooling functions are suitable when an instance-based approach is used.

22.4.2 Embedding-based approach

The embedding-based approach has the same building blocks as the instance-based approach. The main difference of the two approaches lies in the ordering of fully connected layers, used for classification, and the MIL pooling layer. In case of the embedding-based approach, our main goal is finding a compact embedding (latent representation) of a bag. In a second step, we combine the instance embeddings to a single embedding that represents the entire bag. Similar to the instance-based approach, an MIL pooling layer is used to combine the instance embeddings. Though, in this case the MIL pooling layer must be able to handle a vector input in contrast to a scalar value. By sharing the same deep neural network we are guaranteed that all bag embeddings share the same latent space. Fig. 22.2 shows a possible architecture for an end-to-end trainable deep neural network that is using the embedding-based approach. When compared to the instance-based approach, MIL models using the embedding-based approach are known to have a better bag classification performance [27]. However, there is no

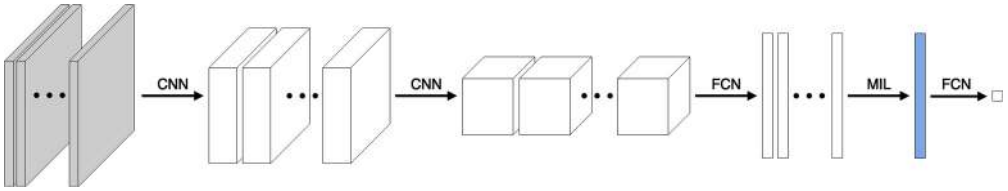

Figure 22.2 Embedded-level approach: First, convolutional and fully connected layers are used to embed each instance in a bag into a low dimensional space. Second, an MIL pooling layer is used to combine the instance embeddings to a single bag embedding. Last, a series of fully connected layers are used to infer the bag label.

way to infer instance scores when using the embedding-based approach. This makes this approach infeasible in a wide variety of settings where interpretability plays a crucial role. In case of the embedded-based approach, f and g are both parameterized by deep neural networks.

22.4.3 Bag-based approach

Bag-based approaches aim at looking for (dis)similarities among bags. They use different metrics like bag distances, bag kernels, or dissimilarities between bags to rephrase an MIL task as a regular supervised problem. Here, the bag is treated as a whole and the implicit assumption is made that bag labels can be related to distances between bags. The biggest challenge of this approach is to find a suitable definition of distance or similarity. Most of the time, a distance or similarity measure is only suited for one task and has to be chosen a priori, i.e., the measure is fixed during training. According to our knowledge, there is no research combining the bag-based approach with deep learning. Additionally, since bags are treated as a whole, there is substantial difficulty to infer instance scores. In the remaining part of this chapter, we will focus on the instance-based approach and embedding-based approach.

22.5. MIL pooling functions

In Sect. 22.4 we emphasized the advantages and disadvantages of the instance-based approach and embedding-based approach. One of the challenges in both approaches, and the MIL problem in general, is the choice of an appropriate MIL pooling functions. In the context of deep neural networks a pooling function is used inside of a pooling layer. Depending on the approach, the pooling layer is responsible for either combing instance scores (instance-based approach) or instance embeddings (embedded-based approach). We turn now to an exploration of the most widely-used MIL pooling functions. The pooling functions introduced in this section will also serve another important purpose, namely, to provide us with opportunity to discuss some key concepts, such as

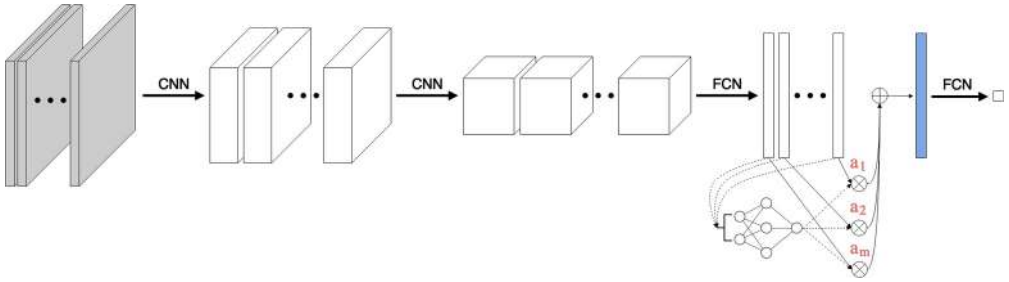

Figure 22.3 Attention-based approach: First, convolutional and fully connected layers are used to embed each instance in a bag into a low dimensional space. Afterwards, the attention mechanism is used to combine the instance embeddings into a single bag embedding. The attention mechanism consists of two fully connected layers that are used to compute an attention weight for each instance. Therefore, instances with a higher attention weight are contributing more to the bag embedding. Last, a series of fully connected layers are used to infer the bag label.

interpretability and *flexibility*, at the end of the section. We shall also show that different pooling functions have different purposes. Understanding those purposes enables us to choose the right pooling function.

Let us consider a bag of instances,

$$X = \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}, \tag{22.7}$$

where instance $\mathbf{x}_k \in \mathbb{R}^D$ is, e.g., an image represented by its raw pixel values. Using the function f_θ, parameterized by a deep neural network shared among instances, we obtain a bag of embeddings,

$$H = \{\mathbf{h}_1, \ldots, \mathbf{h}_K\}, \tag{22.8}$$

where $\mathbf{h}_k = f_\theta(\mathbf{x}_k)$. Depending on the approach we choose, the embedding of an instance is either a scalar in case of the instance-based approach, $h_k \in [0, 1]$, or a vector in case of the embedding-based approach, $\mathbf{h}_k \in \mathbb{R}^M$, where $M < D$.

After having obtained the embeddings for all instances in a bag, we use a pooling function to combine the instance embedding to a bag embedding $\mathbf{z}$. We note that $\mathbf{z}$ has the same dimensionality as each of the instance embeddings, i.e., dimension 1 in case of the instance-based approach, or dimension M in case of the embedding-based approach. The aim of the bag embedding is to capture the most important information of a bag using a low dimensional representation. In contrast to a bag of instance embeddings, a bag embedding can be straightforwardly mapped to the corresponding bag label.

In the following section, we focus on most prominent MIL pooling functions. Our list is not exhaustive and there are many other, more specialized MIL pooling functions, e.g., ISR and Noisy-AND [14].

22.5.1 Max

The most commonly used MIL function is the max function. It can be used in two settings, namely, for a single vector $\mathbf{h} \in \mathbb{R}^K$:

$$z = \max_{k=1,\ldots,K} \{h_k\}, \tag{22.9}$$

or as an elementwise operation over K vectors:

$$\text{for all } m = 1, \ldots, M, \quad z_m = \max_{k=1,\ldots,K} \{h_{km}\}. \tag{22.10}$$

22.5.2 Mean

Another widely used MIL pooling function is the mean function, which, in case of the instance-based approach, averages individual scores:

$$z = \frac{1}{K} \sum_{k=1}^{K} h_k, \tag{22.11}$$

or calculates an average embedding in the embedding-based approach:

$$\mathbf{z} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{h_k}. \tag{22.12}$$

22.5.3 LSE

A continuous relaxation of the max function is the log–sum–exp (LSE) MIL pooling. The LSE function has an additional hyperparameter $r > 0$. In the instance-based approach it is defined as follows:

$$z = r \log \Big[\frac{1}{K} \sum_{k=1}^{K} \exp(r h_k) \Big], \tag{22.13}$$

while in the embedding-based approach it is given by

$$\text{for all } m = 1, \ldots, M, \quad z_m = r \log \Big[\frac{1}{K} \sum_{k=1}^{K} \exp(r h_{km}) \Big]. \tag{22.14}$$

Interestingly, for $r \to \infty$, this function is equivalent to the maximum, and, for $r \to 0$, it results in the mean. In practice the choice of r might be quite problematic, however, it introduces some degree of flexibility of this MIL pooling.

22.5.4 (Leaky) Noisy-OR

Another continuous version of the max function is the Noisy-OR that acts similarly to the logic OR gate. The Noisy-OR is only used with the instance-based approach, where it is defined as

$$z = 1 - \prod_{k=1}^{K}(1 - h_k), \tag{22.15}$$

where $h_k \in [0, 1]$ corresponds to the *success* probability. This function could be further generalized by introducing a *leaky* parameter $h_0 \in [0, 1]$, that is,

$$z = 1 - (1 - h_0)\prod_{k=1}^{K}(1 - h_k). \tag{22.16}$$

22.5.5 Attention mechanism

All pooling functions mentioned in previous subsections have the clear disadvantage that they are predefined and nontrainable, and thus they reduce the flexibility of a MIL model drastically. Therefore, it would be beneficiary to have a fully flexible MIL pooling that can be trained alongside other components of a model. A solution that fulfills this goal is the attention-based MIL pooling [10] that is defined as a weighted sum,

$$\mathbf{z} = \sum_{k=1^K} a_k \mathbf{h}_k, \tag{22.17}$$

with

$$a_k = \frac{\exp\{\mathbf{w}^\intercal \tanh(V)\mathbf{h}_k^\intercal\}}{\sum_{j=1}^{K}\exp\{\mathbf{w}^\intercal \tanh(V)\mathbf{h}_j^\intercal\}}, \tag{22.18}$$

where $\mathbf{w} \in \mathbb{R}^{L\times 1}$ and $\mathbf{V} \in \mathbb{R}^{L\times M}$.

The attention-based MIL pooling layer utilizes an auxiliary network that consists of two fully connected layers. In the first hidden layer, the hyperbolic tangent activation $\tanh(\cdot)$ is used since it is symmetric in its outputs, in contrast to the commonly used activation functions, such as sigmoid or ReLU. The second layer uses the softmax nonlinearity, so that the attention weights sum to 1. The resulting MIL pooling is fully trainable and highly flexible. In addition, the attention weights can be easily interpreted. The higher the attention weight, the higher the relative importance of the object. In case of the positive class, we can use the attention weights to determine the key instances. The attention-based MIL pooling layer can be seen in Fig. 22.3.

Moreover, we can get an extra insight into the attention-based MIL pooling by inspecting its gradient. Calculating the gradient with respect to the parameters of the f transformation, one gets

$$\frac{\partial a_k \mathbf{h}_k}{\partial \theta} = \frac{\partial a_k f_\theta(\mathbf{x}_k)}{\partial \theta} = a_k \frac{\partial f_\theta(\mathbf{x}_k)}{\partial \theta}. \tag{22.19}$$

The attention mechanism can be seen a gradient filter that determines the amount of gradient flow for individual instances [26].

One possible short coming of the attention-based MIL pooling layer is the use of the tanh(·) nonlinearity. The expressiveness of tanh(·) is limited since it is approximately linear for $x \in [-1, 1]$. It was proposed in [10] to additionally use the *gating mechanism* [6] together with tanh(·) nonlinearity, yielding

$$a_k = \frac{\exp\left\{\mathbf{w}^\top \left(\tanh\left(\mathbf{V}\mathbf{h}_k^\top\right) \odot \operatorname{sigm}\left(\mathbf{U}\mathbf{h}_k^\top\right)\right)\right\}}{\sum_{j=1}^{K} \exp\left\{\mathbf{w}^\top \left(\tanh\left(\mathbf{V}\mathbf{h}_j^\top\right) \odot \operatorname{sigm}\left(\mathbf{U}\mathbf{h}_j^\top\right)\right)\right\}}, \tag{22.20}$$

where $\mathbf{U} \in \mathbb{R}^{L \times M}$ are parameters, $\odot$ is an elementwise multiplication and sigm(·) is the sigmoid nonlinearity. The gating mechanism introduces a learnable nonlinearity.

22.5.6 Interpretability

As mentioned before, only the instance-based approach and the attention-based pooling provide interpretability by highlighting the key instances. In case of the MIL, for a positive bag a high instance score or attention weight should ideally correspond with a positive instance label. Nevertheless, the interpretability of the instance-based approach comes with a loss in classification performance.

In the medical imaging domain interpretability is a key attribute of any machine learning method. Gaining insight in the inner workings of a machine learning model is of special interest for human doctors to properly analyze the diagnosis and prescribe appropriate treatment. For example, in digital pathology such a model can be used to highlight Regions Of Interest (ROI) that can be examined by a pathologist. ROIs could serve as indicators for "where-to-look" parts of an image and can drastically decrease the time a pathologist needs per image. Another scenario is the classification of lung CT screenings, where the malignancy of each nodule has to be assessed to derive a final diagnosis for the whole scan [15]. Again, an MIL approach could assist a human doctor in her daily routines.

22.5.7 Flexibility

Classical MIL methods utilized a feature extraction method and one of MIL pooling functions that is nontrainable like max or mean. An MIL pooling that has a hyperpa-

rameter that could be tuned is the LSE. In case of the LSE, if r is chosen to be small, it will approximate the mean and if r is sufficiently large, it will approximate the maximum. However, there are two limitations. First, the LSE can only provide instance scores in the instance-based approach. Second, the hyperparameter r is global, thus, it is not adaptive to new instances. The only MIL pooling that is learnable and adaptive is the attention-based pooling. The attention-based pooling layer can approximate the argmax function over instance if a single a_k equals 1 and the others are 0, and the mean if all attention weights have the same value $1/K$. Furthermore, since the attention weights are functions of the instance embeddings, the attention pooling layer can work as the maximum for positive bags and at the same time as the mean for negative bags.

22.6. Application to histopathology

The examination of a biopsy or surgical specimen can give insights into the state of a disease that other modalities, like CT, MRI, or US, cannot provide. After the specimen has been processed and the tissue have been placed onto glass slides, a pathologist can examine the tissue using a microscope. Unfortunately, the majority of cellular structures are colorless and transparent. Therefore, a wide array of chemicals is used to stain different structures so that they become visible. Staining usually works by using a dye that stains some of the cells components a bright color, together with a counterstain that stains the remaining of the cell in a different color. The most common staining system is Hematoxylin and Eosin (H&E). On the one hand, Hematoxylin stains nuclei blue since it binds to nucleic acids in the cell nucleus. On the other hand, Eosin stains the cytoplasm pink. Fig. 22.4 shows an example of H&E stained tissue.

In recent years, with the advent of digital microscopy, it has become possible to convert glass slides into digital slides. This results in large scale whole-slide images (WSI) of tissue specimen, containing billions of pixels and tens of thousands of cells. The reading of a WSI is a laborious task. Even a highly trained pathologist needs several hours in order to read a single slide thoroughly. Therefore, deep learning, with its capabilities of processing huge amounts of data, holds a great promise to support pathologists in their daily routines.

Already, deep learning methods have shown human-like performance on a set of (restricted) tasks. Common tasks for deep learning systems are: cell segmentation and cell type classification, segmentation and grading of organs, detecting and classifying the disease at the lesion- or WSI-level [16]. Because of the variability of the staining of WSI, deep learning methods have also been applied for normalization of histopathology images.

22.6.1 Data augmentation

Deep neural networks have millions of parameters that need to be tuned in order to learn a mapping from an input to an output, e.g., from an image to a label. To obtain a good final performance, the amount of examples in the training set needs to be proportional to the number of parameters in the network. Also, the number of parameters needed is proportional to the complexity of the task the model has to perform.

However, in the medical domain, where labeling is very time consuming and requires special training, data sets usually consists only of a few hundred or thousand of examples. Furthermore, the tasks, when compared to classical computer vision tasks, are typically significantly more challenging. For example, a histopathology data set might contain only a few dozen WSI. However, a single WSI will usually contain a billion pixels. To increase the number of training examples, data augmentation techniques are used. In the following section, we will look into methods to artificially enlarge data sets. With the help of data augmentation we can reduce overfitting of machine learning models, due to the increased amount of training data.

Another way of looking at the data augmentation is from the perspective of equivariance or invariance. By providing a network with all possible variations of the input data, the network can learn to disregard certain changes in the input. For instance, a translation in the input space should not change the predicted class label (invariance), furthermore it should only result in a translation of the segmentation by the same amount of pixels (equivariance).

Next, we will describe the most common data augmentation techniques utilized in deep learning methods applied to histopathology images. In contrast to other domains like computer vision, medical imaging is a field where safety plays a crucial role. Therefore, all the described data augmentation methods should be carefully considered and used only if they reflect possible real-life variants of the data. If the presented methods are used too extensively the model will become too confident when encountering abnormal changes. We might still see a performance gain but severe side effects may occur when these models are used in practice.

22.6.1.1 Cropping

In many practical scenarios one relies on existing neural network architectures, e.g., inception networks [24], that are proven to have a good performance. In order to use those models, we have to adjust the size of the images to match the input dimensions of the network. In medical imaging it is quite likely that the images are larger than the input dimensions of the model. Here, multiple smaller patches are extracted from the original images by randomly shifting the center of the cropped area by a few pixels left and right, and up and down. By shifting the input patch the network becomes robust to translations in the input space. As mentioned above, a translation in the input space should not lead to unforeseen changes of the output of a network.

22.6.1.2 Rotating and flipping

For a human observer a rotated or flipped image still contains the same content. Unfortunately, (convolutional) neural networks are very sensitive to such transformations [5]. In other words, the rotation of the input can lead to very different class predictions. To prevent such a behavior rotations and flips to the input images are applied. In case of rotation by an arbitrary angle the problem of interpolation arises, since the two images (original and rotated) are no longer sharing the same grid. In addition, one has to fill the corners in case where these are empty for the rotated image. Therefore, the simplest way of data augmentation is a rotation by 90, 180 and 270 degrees, as seen in Fig. 22.4. In addition, it is a common practice to flip images around the vertical and horizontal axis.

22.6.1.3 Blur

In the majority of cases, blurriness of digital histopathology images is introduced during the digitalization of the WSI. Since sections of the tissue slice are unevenly aligned, due to the different thickness of tissue sections, with the microscope focal plane the degree of blurriness varies across the WSI. By training with blurred images the model should become invariant to blurriness. In practice a Gaussian filter is used to artificially blur images:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left\{\frac{x^2 + y^2}{2\sigma^2}\right\}. \tag{22.21}$$

The degree of blurriness can be tuned by changing the value of σ. Fig. 22.4 shows an example of an artificially blurred histopathology image.

22.6.1.4 Color

As described before, different dyes are applied to WSI in order to make certain cellular structures visible. Since staining is a nondigital process, it has a high variability, it is a common practice to slightly modify the colors of a histopathology image. There are two mainly used methods to deal with staining variability. The bottom row of Fig. 22.4 shows one example image for each method.

Color decomposition

Here, the RGB channels of a WSI are decomposed into the Hematoxylin, Eosin, and DAB channels. It is assumed that the obtained channels are uncorrelated. Afterwards the magnitude of H&E for a pixel is multiplied by two i.i.d. Gaussian random variables with expectation equal to one. The third, DAB channel remains constant. Finally, we map the channels back into the RGB color space.

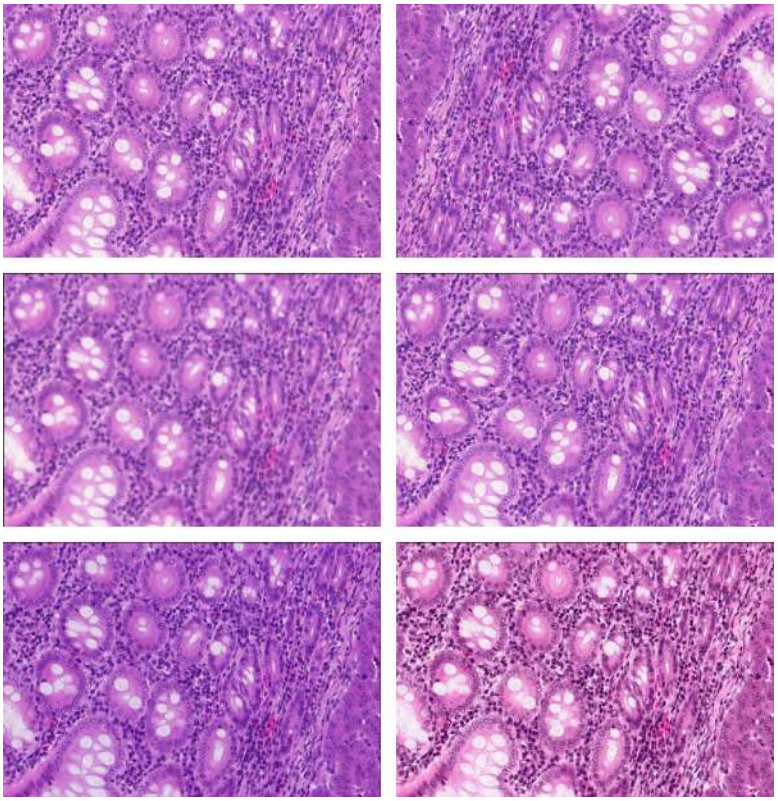

Figure 22.4 Different data augmentation methods applied to a histopathology image. Clockwise, starting at the upper left corner: No augmentation. Rotation: Rotation by 180°. Elastic Deformation: Deformation using a 3×3 grid and a displacement of 30 pixels. Color normalization: $l\alpha\beta$ channels are normalized using a target image. Color Augmentation: multiplying the magnitude of H&E for a pixel by two i.i.d. Gaussian random variables with expectation equal to one and $\sigma = 0.02$. Blur: Gaussian blur with σ set to 1.3.

Color normalization

The main idea of the color normalization technique is that the characteristics of all images in the data set are changed towards the characteristics of a predefined target image [21]. First, we map the RGB channels of the original image to a new color space, called $l\alpha\beta$. This new color space has two main advantages: **(i)** the three channels in the new color space have minimal correlation, in contrast to RGB where channels show a high amount of correlation, **(ii)** the $l\alpha\beta$ color space is logarithmic, thus, uniform changes lead to equally detectable changes in intensity. Second, we whiten the image in $l\alpha\beta$ space by subtracting the mean and dividing by the standard deviation, **(iii)** we multiply the channels by the standard deviation of the target image and add the mean

of the target image. The resulting image now features the image characteristics of the target image. Last, we map the image bag to RGB space.

22.6.1.5 Elastic deformations

Elastic deformations have been proven to be useful especially for segmentation tasks of cellular structures [22]. First, we subdivide the image in patches. Second, we make use of random displacement field to deform images. A displacement field is a matrix that causes pixel values in each patch to be moved to new locations. A 2D matrix with uniformly distributed random numbers will lead to random displacements of the pixels of the original image. In order to guarantee a smooth displacement, we have to convolve the displacement matrix with a Gaussian blur in (22.21). Applying elastic deformation to the images in our training set the network learns to be invariant such deformations. This is particularly since deformation is one of the most common variation in tissue and realistic deformations can be simulated accurately, see Fig. 22.4 for an example image.

22.6.1.6 Generative models

One of the most promising new approaches for data augmentation is the use of generative models. Recent publications were able to show that generative models for data augmentation are also applicable in the field of medical imaging [28]. There are two major classes of deep generative model: Generative Adversarial Networks [9] and Variational Autoencoders [13]. Both classes of models have shown their effectiveness to learn from unlabeled data to generate photo realistic images. After training, generative models are able to synthesize images with particular characteristics. In this paper we do not use this approach for data augmentation, however, we want to highlight its huge potential.

22.6.2 Performance metrics

In Sect. 22.6.3, we are going to compare the performance of different settings and models on two histopathology data sets. As we defined in Sect. 22.3, in the MIL problem the distribution of the bag labels is a Bernoulli distribution. Therefore, we can make use of a wide array of performance metrics that assume binary class labels. In the following, we consider the illustrative example of classifying WSI using the labels malignant (1) or benign (0). Furthermore, we assume that only 10% of the WSI contain malignant changes. On the one hand, such a ratio is not an unrealistic one in practice; on the other hand, it will help us to highlight the shortcomings of some of the metrics when dealing with unbalanced data sets. In this section we will make use of the following notation: True positive (TP), label is 1 prediction is 1; False positive (FP), label is 0 prediction is 1; True negative (TN), label is 0 prediction is 0; False negative (FN), label is 1 prediction is 0.

22.6.2.1 Accuracy

The most widely-used measure is the classification accuracy. The accuracy is defined by the fraction of correctly classified data points:

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{22.22}$$

Considering our example of classifying WSIs, we can easily see that accuracy will fail to assess the performance of a model that is always predicting 0. In this case the model would achieve an accuracy of 0.9. The second drawback of the classification accuracy is the necessity of choosing a classification threshold for the output values of the model to provide either 0 or 1. The output of the model is a real number between 0 and 1, $\hat{Y} \in [0, 1]$, while the labels of the WSI are binary, $Y \in \{0, 1\}$. To be able to compute TP, FP, TN, and FN, we need to threshold the model's output. A commonly used threshold is 0.5, i.e., if $\hat{Y} \geq 0.5$, the final prediction will be equal to 1, otherwise it will be 0. Since the threshold is not represented by the loss function, there is no guarantee that, e.g., 0.5 will lead to the best possible classification performance of the model.

22.6.2.2 Precision, recall and F1-score

After noticing possible limitations of the classification accuracy as a performance metric, we now look at precision and recall, and one particular combination of the two, namely, the F1-score. Precision is the ratio of WSI that were correctly classified as malignant and all WSI that were classified as malignant,

$$\text{precision} = \frac{TP}{TP + FP}. \tag{22.23}$$

Precision is the ratio of WSI that were correctly classified as malignant and the all correctly classified WSI as either malignant or benign,

$$\text{recall} = \frac{TP}{TP + FN}. \tag{22.24}$$

In case of the unbalanced WSI image and a classifier always predicting benign for all WSIs, we can see that the precision equals 0 as well as the recall. It is easy to see that for an imbalanced data set precision and recall are better suited than accuracy. Often, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. The $F1$-score is one particular and commonly used way of combining precession and recall into one performance measure,

$$\text{F1-score} = 2\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{22.25}$$

One common criticism of the measures mentioned earlier is that the true negative, TN, is not taken into account. This results in zero precision and recall for our WSI example. Again, precision, recall and F1-score suffer from the same problem of having only one single classification threshold. In the next section we introduce a performance measure that does not suffer from this issue.

22.6.2.3 Receiver Operating Characteristic Area Under Curve

In contrast to the performance metrics described above, the Receiver Operating Characteristic Area Under Curve (ROC AUC, commonly denoted as AUC) is not restricted to a single classification threshold. The ROC curve allows us to compare multiple pairs of sensitivity and specificity values, where the sensitivity, or the true positive rate, is given by

$$\text{sensitivity} = \frac{TP}{TP + FN}, \tag{22.26}$$

and the specificity, or the true negative rate, is given by

$$\text{specificity} = \frac{TN}{TP + FN}. \tag{22.27}$$

The ROC curve can be drawn using the sensitivities as the y-coordinates and the specificities as the x-coordinates. As the classification threshold decreases, the specificity decreases as well, while the sensitivity increases, and vice versa. Each point on the graph is called an operating point. Each operating point is generated using a different classification threshold. Since the ROC curve displays the sensitivity and the specificity at all possible classification thresholds, it can be used to evaluate the performance of a machine learning model independently of the classification threshold. As in case of the F1-score, we are interested in finding one metric that combines all the information represented by the ROC curve. This is most commonly done by measuring the area under the receiver operating characteristic curve. It is a measure of the overall classification performance of a machine learning model interpreted as the mean value of sensitivity for all possible values of specificity. The AUC can have any value between 0 and 1, since both the x and y axes are strictly positive. The closer the value of AUC is to 1, the better the overall classification performance of a machine learning model.

If we were to rely on pure chance to classify data samples, the ROC curve would fall along the diagonal line segment from $(0, 0)$ to $(1, 1)$. The resulting ROC curve has an area of 0.5. At last, we want to note that having machine learning models with the same classification performance result in ROCs and the same AUC value does not mean that the machine learning models are identical.

22.6.3 Evaluation of MIL models

In this section we evaluate MIL methods in a quantitative manner, using the performance metrics introduced in Sect. 22.6.2. We investigate the performance of the two deep MIL approaches with different pooling layers on two histopathology data sets, called Breast Cancer and Colon Cancer. In both cases we frame the learning task as a MIL task where we try to infer the binary labels of a bag of patches. The two data sets comprise images extracted from WSI, where the size of the extracted images range from 500×500 pixels to 896×768 pixels. In order to process such large images, we will make use of MIL methods. Here, each image is represented as a collection of smaller patches containing a nuclei and adjacent tissue, where each image is treated as a bag while each patch is an instance. During training we only use information about the label of the original image. By focusing on single cells and their nuclei, the MIL models are able to learn various cell attributes such as shape, size, and smoothness of the boundary. These morphological properties of cells are crucial for classifying dysplastic changes. The diagnosis of dysplastic changes in surveillance biopsies is one of the strongest independent risk factors for progression.

One crucial part of the presented MIL experiments is extracting patches from histopathology images. There are two common classes of methods to generate an MIL data set. The first class makes use of a preceding detector. The detector can be either a human or a computer program, e.g., another machine learning model, that finds regions of interest and discards parts of the image that are considered irrelevant for the task. The Colon Cancer data set is an example of this class, here the cells were presegmented by a pathologist. The second class of methods is subdividing the image into patches using a rigid grid. Depending on the number of resulting patches and computational resources, sampling methods can be used to reduce the number of patches. Here, it is common practice to discard patches that are not containing any cells. The Breast Cancer data set is an example of this generation method. In this case the model not only has to differentiate between benign and malignant tissue but also find important cellular structures. The presented results show that, in general, MIL works better the higher the quality of the provided instances.

For both data sets we use the same experimental setup. We will see how the two MIL approaches shape the architecture of the MIL models. By carefully studying the experimental setup we are able to make connections between the mathematical framework introduced in Sect. 22.3 and a real-world application.

22.6.3.1 Experimental setup

Tables 22.1 and 22.2 show the architecture of different MIL models for the embedded-based approach and the instance-based approach. This particular architecture, including the size and number of filters and layers, is proven to have good classification performance on patches containing single cells [23]. For both approaches the network has

Table 22.1 Model architecture for embedding-based approach. The neural network consists of convolutional layers (conv(kernel size, stride, padding)-number of filters + activation function), fully connected layers (fc-output dimensions + activation function) and MIL pooling layers. Here, ReLU represents the Rectified Linear Unit activation function and sigm the sigmoid activation function.

Layer	Type
1	conv(4,1,0)-36 + ReLU
2	maxpool(2,2)
3	conv(3,1,0)-48 + ReLU
4	maxpool(2,2)
5	fc-512 + ReLU
6	dropout
7	fc-512 + ReLU
8	dropout
9	mil-pooling
10	fc-1 + sigm

the same number of layers. Therefore we can have a fair comparison of the different approaches. An important difference, as discussed in Sect. 22.4, is the order of layers. In case of the embedded-based approach, the patches are embedded using two convolutional layers and two fully connected layers. All layers are shared among the instances. An MIL pooling layer is used to combine the embeddings to a single embedding representing the entire bag. Lastly, a single fully connected layer is used for classifying the bag.

In contrast, using the instance-based approach, each instance is processed using two convolutional layers and three fully connected layers. The output of the last layer is a single score, a real number between 0 and 1, for each instance. In a final step all scores are combined by a MIL pooling layer. The combination of the instance scores results in the final bag score.

All models are trained using the same hyperparameters. For training we use the Adam optimizer with the default settings $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a learning rate equal to 0.0001. Since the number of samples in both data sets is very limited, weight decay and the early stopping were used to prevent overfitting. As mentioned frequently in this chapter, one of the biggest challenges in medical imaging is the difficulty of obtaining high quality labels. This results in comparably small data sets. In order to compensate for the small number of training examples data augmentation methods are used. First, the center of a patch is randomly shifted by a small number of pixels. Second, each

Table 22.2 Model architecture for instance-based approach. The neural network consists of convolutional layers (conv(kernel size, stride, padding)-number of filters + activation function), fully connected layers (fc-output dimensions + activation function) and MIL pooling layers. Here, ReLU represents the Rectified Linear Unit activation function and sigm the sigmoid activation function.

Layer	Type
1	conv(4,1,0)-36 + ReLU
2	maxpool(2,2)
3	conv(3,1,0)-48 + ReLU
4	maxpool(2,2)
5	fc-512 + ReLU
6	dropout
7	fc-512 + ReLU
8	dropout
9	fc-1 + sigm
10	mil-pooling

patch was randomly rotated and flipped. Third, the patches are randomly blurred using a Gaussian kernel. Fourth, the H&E staining of each patch are randomly adjusted.

22.6.3.2 Colon cancer

Colorectal cancer (colon cancer) is known to originate from epithelial cells lining the colon and rectum. Therefore, tagging epithelial cells is highly relevant from a clinical point of view. One way to explore these cell types is to use special biomarkers which can highlight certain cells in the cancer tissue. However, such an approach is time consuming and requires the selection of the appropriate markers a priori. Furthermore, a single histopathology image can contain thousands of epithelial cells, where malignant epithelial cells often appear highly cluttered together with irregular shaped nuclei. For the former reason a fully supervised approach that relies on class labels for every single cell is not well suited. Using MIL we make use of the ability of MIL models to find cells of a certain type while only using image labels during training. The weak image labels can be obtained, e.g., by knowing the area where the tissue was extracted from. The Colon Cancer data set has been made publicly available by the University of Warwick and comprises 100 H&E images of colorectal adenocarcinomas [23]. The images come from a variety of tissue appearance from both benign and malignant regions. In each image the nuclei of all cells are marked. In total there are 22,444 nuclei with 4 associated class labels: epithelial, inflammatory, fibroblast, and miscellaneous. For every cell a patch

Table 22.3 Results on Colon Cancer data set. Experiments were run 5 times and an average (± one standard error of the mean) is reported.

METHOD	ACCURACY	PRECISION	RECALL	F-SCORE	AUC
Instance+max	0.842 ± 0.021	0.866 ± 0.017	0.816 ± 0.031	0.839 ± 0.023	0.914 ± 0.010
Instance+mean	0.772 ± 0.012	0.821 ± 0.011	0.710 ± 0.031	0.759 ± 0.017	0.866 ± 0.008
Embedding+max	0.824 ± 0.015	0.884 ± 0.014	0.753 ± 0.020	0.813 ± 0.017	0.918 ± 0.010
Embedding+mean	0.860 ± 0.014	0.911 ± 0.011	0.804 ± 0.027	0.853 ± 0.016	0.940 ± 0.010
Attention	$\mathbf{0.904} \pm 0.011$	$\mathbf{0.953} \pm 0.014$	$\mathbf{0.855} \pm 0.017$	$\mathbf{0.901} \pm 0.011$	$\mathbf{0.968} \pm 0.009$
Gated-Attention	$\mathbf{0.898} \pm 0.020$	$\mathbf{0.944} \pm 0.016$	$\mathbf{0.851} \pm 0.035$	$\mathbf{0.893} \pm 0.022$	$\mathbf{0.968} \pm 0.010$

of 27×27 pixel was extracted. Furthermore, a bag of patches is given a positive label if it contains one or more nuclei from the epithelial class. While the primary task is to predict the bag label of an unseen image, assuming presegmented cells, we are even more interested in highlighting all epithelial cells. A task that only the attention-based MIL pooling layers and MIL pooling layers used with an instance-level approach can solve.

The results of experiments on the Colon Cancer data set are presented in Table 22.3. First, we notice that models of the instance-level approach are performing worse than all other models, disregard of the used MIL pooling layer. Furthermore, attention-based MIL pooling layers perform the best across all metrics. It is most likely that they benefit from the properties described in Sect. 22.5.5. In case of the embedding-level approach the mean pooling layer is performing significantly better than the max pooling layer. This finding indicates that the mean pooling layer is better suited if a bag contains a high number of positive instances, as it is the case for the Colon Cancer data set.

In Fig. 22.5, we compare the ability of the best performing instance-level approach model and the best performing attention-based model with respect to their ability to discover key instances. We can easily see that the attention-based pooling layer is able to highlight more epithelial cells than the max-instance pooling layer. By using an auxiliary network, the MIL pooling layer is not restricted to either have interpretable instance scores or discriminative instance embeddings. As a result the attention-based MIL pooling layers show the best classification performance together with having highly interpretable attention weights a_k.

22.6.3.3 Breast cancer

After abnormal findings during a mammography, a biopsy is used to gain further insight into the conditions of a patient. A biopsy removes tissue from the suspicious area of the breast. After the extraction, the tissue is stained and fixed to a glass slide. At last, the obtained histopathology slide is digitalized. The Breast Cancer data set is part of a bigger corpus of medical data, the UCSB Bio-Segmentation Benchmark [8]. It consists of 58 weakly labeled 896×768 H&E images. Unlike the Colon Cancer data set, the image label is 1 if an image contains malignant changes. An image is labeled as benign

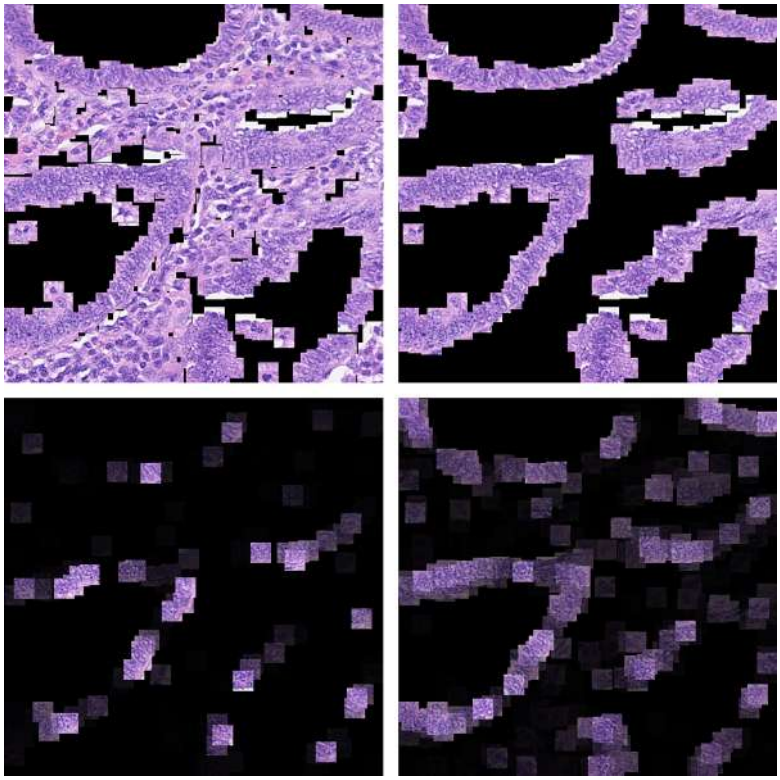

Figure 22.5 Clockwise, starting in the upper left corner: Instances: 27 × 27 patches centered around all marked nuclei. Ground truth: Patches that belong to the class epithelial. Attention heatmap: Every patch multiplied by its attention weight. Instance+max heatmap: Every patch multiplied by its instance score. We rescaled the attention weights and instance scores using $a'_k = (a_k - \min(\mathbf{a}))/(\max(\mathbf{a}) - \min(\mathbf{a}))$.

(0) if no abnormal changes are present. In case of the Breast Cancer data set the images were not presegmented. We divide every image into 32 × 32 patches. This results in 672 patches per bag. A patch is discarded if it contains 75% or more of white pixels. Compared to the Colon Cancer data set this is considerably more challenging task since not all patches will contain a centered cell. The lower performance values in Table 22.4 reflect the challenging nature of the task. In addition, the data set contains fewer images than the Colon Cancer data set.

Once more the attention-based pooling layers are among the best performing models. Even though the embedded-mean model leads to similar classification results on the bag level, it does not provide any interpretable results on the instance level. In addition, it is important to notice that one of the most common MIL pooling layers, the max pooling layer, is having low values across all metrics.

Table 22.4 Results on Breast Cancer data set. Experiments were run 5 times and an average (± one standard error of the mean) is reported.

METHOD	ACCURACY	PRECISION	RECALL	F-SCORE	AUC
Instance+max	0.614 ± 0.020	0.585 ± 0.03	0.477 ± 0.087	0.506 ± 0.054	0.612 ± 0.026
Instance+mean	0.672 ± 0.026	0.672 ± 0.034	0.515 ± 0.056	0.577 ± 0.049	0.719 ± 0.019
Embedding+max	0.607 ± 0.015	0.558 ± 0.013	0.546 ± 0.070	0.543 ± 0.042	0.650 ± 0.013
Embedding+mean	$\mathbf{0.741} \pm 0.023$	$\mathbf{0.741} \pm 0.023$	0.654 ± 0.054	0.689 ± 0.034	$\mathbf{0.796} \pm 0.012$
Attention	$\mathbf{0.745} \pm 0.018$	0.718 ± 0.021	$\mathbf{0.715} \pm 0.046$	$\mathbf{0.712} \pm 0.025$	0.775 ± 0.016
Gated-Attention	$\mathbf{0.755} \pm 0.016$	$\mathbf{0.728} \pm 0.016$	$\mathbf{0.731} \pm 0.042$	$\mathbf{0.725} \pm 0.023$	$\mathbf{0.799} \pm 0.020$

References

[1] Stuart Andrews, Ioannis Tsochantaridis, Thomas Hofmann, Support vector machines for multiple-instance learning, in: NIPS, 2003, pp. 577–584.

[2] Yixin Chen, Jinbo Bi, James Ze Wang, MILES: multiple-instance learning via embedded instance selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (12) (2006) 1931–1947.

[3] Veronika Cheplygina, Lauge Sørensen, David Tax, Marleen de Bruijne, Marco Loog, Label stability in multiple instance learning, in: MICCAI, 2015, pp. 539–546.

[4] Veronika Cheplygina, David M.J. Tax, Marco Loog, Multiple instance learning with bag dissimilarities, Pattern Recognition 48 (1) (2015) 264–275.

[5] Taco S. Cohen, Max Welling, Group equivariant convolutional networks, in: ICML, 2016.

[6] Yann N. Dauphin, Angela Fan, Michael Auli, David Grangier, Language modeling with gated convolutional networks, arXiv preprint arXiv:1612.08083, 2016.

[7] Thomas G. Dietterich, Richard H. Lathrop, Tomás Lozano-Pérez, Solving the multiple instance problem with axis-parallel rectangles, Artificial Intelligence 89 (1–2) (1997) 31–71.

[8] Elisa Drelie Gelasca, Jiyun Byun, Boguslaw Obara, B.S. Manjunath, Evaluation and benchmark for biological image segmentation, in: IEEE International Conference on Image Processing, 2008, pp. 1816–1819.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative adversarial networks, in: NIPS, 2014.

[10] Maximilian Ilse, Jakub M. Tomczak, Max Welling, Attention-based deep multiple instance learning, in: ICML, 2018.

[11] Melih Kandemir, Fred A. Hamprecht, Computer-aided diagnosis from weak supervision: a benchmarking study, Computerized Medical Imaging and Graphics 42 (2015) 44–50.

[12] Melih Kandemir, Chong Zhang, Fred A. Hamprecht, Empowering multiple instance histopathology cancer diagnosis by cell graphs, in: MICCAI, 2014, pp. 228–235.

[13] Diederik P. Kingma, Max Welling, Auto-encoding variational Bayes, in: ICLR, 2013.

[14] Oren Z. Kraus, Jimmy Lei Ba, Brendan J. Frey, Classifying and segmenting microscopy images with deep multiple instance learning, Bioinformatics 32 (12) (2016) i52–i59.

[15] Fangzhou Liao, Ming Liang, Zhe Li, Xiaolin Hu, Sen Song, Evaluate the malignancy of pulmonary nodules using the 3D deep leaky noisy-or network, URL http://arxiv.org/abs/1711.08324.

[16] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, Clara I. Sánchez, A survey on deep learning in medical image analysis, Medical Image Analysis 42 (2017) 60–88.

[17] Guoqing Liu, Jianxin Wu, Zhi-Hua Zhou, Key instance detection in multi-instance learning, in: JMLR, vol. 25, 2012, pp. 253–268.

[18] Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas, PointNet: deep learning on point sets for 3D classification and segmentation, in: CVPR, 2017.

[19] Jan Ramon, Luc De Raedt, Multi instance neural networks, in: ICML Workshop on Attribute-value and Relational Learning, 2000, pp. 53–60.
[20] Vikas C. Raykar, Balaji Krishnapuram, Jinbo Bi, Murat Dundar, R. Bharat Rao, Bayesian multiple instance learning: automatic feature selection and inductive transfer, in: ICML, 2008, pp. 808–815.
[21] Erik Reinhard, Michael Ashikhmin, Bruce Gooch, Peter Shirley, Color transfer between images, IEEE Computer Graphics and Applications (2001).
[22] Olaf Ronneberger, Philipp Fischer, Thomas Brox, U-Net: convolutional networks for biomedical image segmentation, in: MICCAI, 2015.
[23] Korsuk Sirinukunwattana, Shan E. Ahmed Raza, Yee-Wah Tsang, David R.J. Snead, Ian A. Cree, Nasir M. Rajpoot, Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images, IEEE Transactions on Medical Imaging 35 (5) (2016) 1196–1206.
[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, Going deeper with convolutions, in: CVPR, 2015.
[25] Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, Max Welling, Rotation equivariant CNNs for digital pathology, in: MICCAI, 2018.
[26] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, Xiaoou Tang, Residual attention network for image classification, in: CVPR, 2017.
[27] Xinggang Wang, Yongluan Yan, Peng Tang, Xiang Bai, Wenyu Liu, Revisiting multiple instance neural networks, Pattern Recognition 74 (2016) 15–24.
[28] Jelmer M. Wolterink, Tim Leiner, Ivana Isgum, Blood vessel geometry synthesis using generative adversarial networks, URL http://arxiv.org/abs/1804.04381.
[29] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, Alexander Smola, Deep sets, in: NIPS, 2017.
[30] Cha Zhang, John C. Platt, Paul A. Viola, Multiple instance boosting for object detection, in: NIPS, 2006, pp. 1417–1424.