MODEL AGNOSTIC INTERPRETABILITY FOR MULTIPLE INSTANCE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In Multiple Instance Learning (MIL), models are trained using bags of instances, where only a single label is provided for each bag. A bag label is often only determined by a handful of key instances within a bag, making it difficult to interpret what information a classifier is using to make decisions. In this work, we establish the key requirements for interpreting MIL models. We then go on to develop several model-agnostic approaches that meet these requirements. Our methods are compared against existing inherently interpretable MIL models on three datasets, and achieve an increase in interpretability accuracy of up to 30%. We also examine the ability of the methods to identify interactions between instances and scale to larger datasets, improving their applicability to real-world problems.

1 INTRODUCTION

In Multiple Instance Learning (MIL), data is organised into bags of instances, and each bag is given a single label. This reduces the burden of labelling, as each instance does not have to be assigned a label, making it useful in applications where labelling is expensive, such as healthcare (Carbonneau et al., 2018). However, there are often only a few key instances within a bag that determine the bag label, so it is necessary to identify these key instances in order to interpret a model's decision-making (Liu et al., 2012). Directly interpreting the decision-making process of machine learning methods is difficult due to the complexity of the models and the scale of the data on which they trained, so there is a need for methods that allow insights into the decision-making processes (Gilpin et al., 2018).

In MIL problems, only some of the instances in each bag will be discriminatory, i.e., a significant number of the instances in a bag will be non-discriminatory or even related to other bag classes (Amores, 2013). Identifying the important instances and presenting those as the interpretation of model decision-making filters out the non-discriminatory information and provides the explainee with a reduced and relevant interpretation, which will avoid overloading the user (Li et al., 2015). Our work provides the following novel contributions:

1. **MIL interpretability definition** We identify two questions that interpretability methods for MIL should be able to answer: 1) Which are the key instances for a bag? 2) What outcome (class/value) does each key instance support? In the rest of this work, we will refer to these two questions as the *which* and the *what* questions respectively. As we explore in Section 2, existing interpretability methods are able to answer *which* questions with varying degrees of accuracy, but can only answer *what* questions under certain assumptions.
2. **MIL model-agnostic methods** Existing methods are often model-specific, i.e., they can only be applied to certain types of MIL models. Using model-specific methods constrains the model choice to a specific type of inherently interpretable model, and it is difficult to compare the interpretability performance of different types of models (Ribeiro et al., 2016). Therefore, we propose several model-agnostic methods that can be applied to any MIL model, and are able to answer both *which* and *what* questions.
3. **Method comparison** We compare existing model-specific methods with our model-agnostic methods on three different MIL datasets. Our experiments are also carried out on four types of MIL model, giving a comprehensive comparison of interpretability performance. To the best of the authors' knowledge, this is one of the first studies to compare different interpretability methods for MIL.

The remainder of this work is laid out as follows. Section 2 provides relevant background knowledge and reviews existing MIL interpretability methods. Section 3 outlines the requirements for MIL interpretability and details our approaches that meet these requirements. Next, Section 4 provides our results on three MIL datasets. Section 5 discusses our findings, and Section 6 concludes.

2 BACKGROUND AND RELATED WORK

Identifying the key instances in MIL bags is a form of local interpretability (Molnar, 2020), as the key instances are detected for a particular input to the model. Two of the key motivators for interpreting the decision-making process of a MIL system are reliability and trust — identifying the key instances allows an evaluation of the reliability of the system, which increases trust as the decision-making process becomes more transparent.

The standard MIL assumption (SMIL) is a binary problem with positive and negative bags (Dietterich et al., 1997; Maron & Lozano-Pérez, 1998). A bag is positive if any of its instances are positive, otherwise it is negative. The assumptions of SMIL can be relaxed to allow more generalised versions of MIL, e.g., through extensions that include additional positive classes (Scott et al., 2005; Weidmann et al., 2003). SMIL also assumes there is no interaction between instances. However, recent work has highlighted that modelling relationships between instances is beneficial for performance (Tu et al., 2019; Zhou et al., 2009). Existing methods for interpreting MIL models often rely on the SMIL assumption, so cannot generalise to other MIL problems. In addition, existing methods are often model-specific, i.e., they only work for certain types of MIL models.

Under the SMIL assumption, *which* and *what* questions are equivalent — there are only two classes (positive and negative), and the only key instances are the instances that are positive, therefore once the key instances are identified, it is also known what outcome they support. However, when there are multiple positive classes, unless each class is completely independent (i.e., instances from different classes co-occur in the same bags), answering *which* and *what* questions becomes two distinct problems. For example, some key instances will support one positive class, and some key instances will support another. Therefore, solely identifying *which* are the key instances does not answer the second question of *what* class they support. Existing methods, such as key instance detection (Liu et al., 2012), MIL attention (Ilse et al., 2018), and MIL graph neural networks (GNNs; Tu et al. (2019)) do not condition their output on a particular class, so it is not apparent what class each instance supports, i.e., they can only answer *which* questions. One existing method that can answer *what* questions is mi-Net (Wang et al., 2018), as it produces instance-level predictions as part of its processing. However, these instance-level predictions do not take account of interactions between the instances, so often provide inaccurate interpretations. In this work we aim to overcome the limitations of existing methods by developing model-agnostic methods that can answer both *which* and *what* questions.

In single instance supervised learning, model-agnostic techniques have been developed to interpret models. One such group of methods are SHapley Additive exPlanations (SHAP) that approximate Shapley values as a method of interpretation (Lundberg & Lee, 2017). Arising from game theory, Shapley values describe how to fairly distribute the payout of a game amongst its players based on how much they have each contributed (Shapley, 1953). For a given player, the Shapley value indicates the average contribution of that player over all possible coalitions of players. In supervised learning, this can be used to create explanations based on feature attribution (Merrick & Taly, 2020). The benefit of using a Shapley-based approach is that it accounts for interactions between features. This makes it an attractive solution for interpreting MIL models when there are interactions between instances. In this work, three of our methods utilise SHAP for model-agnostic MIL interpretability.

3 METHODOLOGY

In this section we outline the requirements for MIL interpretability (Section 3.1). In Section 3.2, we propose three methods that meet these requirements under the assumption that there are no interactions between instances (independent-instance methods). In Section 3.3 we remove this assumption, and develop three SHAP-based methods that are able to account for interactions between instances.

3.1 MIL INTERPRETABILITY REQUIREMENTS

A general MIL classification problem has C possible classes, with one class being negative and the rest being positive. This is a generalisation of the SMIL assumption, which is a special case when $C = 2$. An interpretability method that can only provide the general importance of an instance (without associating it with a particular class) can only answer *which* questions. In order to answer *what* questions, the method needs to state which classes each instance supports and refutes. Formally, for a bag of instances $X = \{x_1, \dots, x_k\}$ and a bag classification function F, we want to assign a value to each instance that represents whether it supports or refutes a class c:

$$\mathcal{I}(X, F, c) = \{\phi_1, \dots, \phi_k\}$$

where $\mathcal{I}$ is the interpretability function and $\phi_i \in \mathbb{R}$ is the interpretability value for instance i with respect to class c. Here, we assume that $\phi_i > 0$ means instance i supports class c, $\phi_i < 0$ implies instance i refutes class c, and the greater $|\phi_i|$, the greater the importance of instance i. For some existing methods, such as attention, ϕ_i is the same for all classes, and $\phi_i \geq 0$ in all cases, so these requirements are not met. We later demonstrate these limitations in Section 5. In the next two sections we propose several model-agnostic methods that satisfy these requirements.

3.2 DETERMINING INSTANCE ATTRIBUTIONS

In this section, we propose three model-agnostic methods for interpreting MIL models under the assumption that the instances are independent. As the instances are independent, we can observe the effects of each instance in isolation without worrying about interactions between the instances. In each of these methods, we exploit a property of MIL models: the ability to deal with different sized bags. All MIL models are able to process bags of different sizes, therefore it is possible to remove instances from the bags and observe any changes in prediction, allowing us to understand what instances are responsible for the model's prediction. Below, we propose three methods that use this property in order to interpret a model's decision making.

Single instance Given a bag of instances $X = \{x_1, \dots, x_k\}$ and a bag classification function F, we can take each instance in turn and form a single instance bag: $X_i = \{x_i\}$ for $i \in \{1, \dots, k\}$. For a particular class c, we can then observe the model's prediction on each single instance bag $\phi_i = F_c(X_i)$, where F_c is the output of F for class c. A large value for ϕ_i suggests instance x_i supports c, and a value close to zero suggest it has no effect with respect to class c. If we repeat this over all instances and all classes, we can build a picture of the classes that each instance supports, allowing us to answer *what* questions. This gives the same outcome as the inherent interpretability performed by mi-Net (Wang et al., 2018), but here it is a model-agnostic rather than a model-specific method. However, this method cannot refute classes (i.e., $\phi_i \geq 0$ in all cases).

Leave one out A natural counterpart to the single instance method is the leave one out method, where each instance is removed from the complete bag in turn, i.e., we form bags $X_i = X \backslash \{x_i\}$. For a particular class c, we can then observe the change in the model's prediction caused by removing x_i from the bag: $\phi_i = F_c(X) - F_c(X_i)$. If the removal of this instance changes the model's prediction, then x_i has some relation to class c — if the prediction decreases, x_i supports c, and if it increases, x_i refutes c, i.e., this method is able to both support and refute different classes. However, if there are other instances in the bag that support or refute class c, we may not observe a change in prediction when x_i is removed, even if x_i is a key instance.

Combined In order to access the benefits of both the single instance and leave one out methods, we can combine their outputs. A simple approach to combine the instance attributions is to take their mean, i.e., $\phi_i = \frac{1}{2}[F_c(\{x_i\}) + F_c(X) - F_c(X \setminus \{x_i\})]$. By combining these outputs, this method should be able to identify the important instances revealed by the single instance method, and also refute certain outcomes as revealed by the leave one out method.

With these three methods, it is assumed that there are no interactions between the instances. While this may be true for some datasets, it is not always the case. Therefore, in the next section, we remove this assumption and propose three SHAP-based methods that are able to deal with the interactions between instances.

3.3 DEALING WITH INSTANCE INTERACTIONS

In order to calculate instance attributions whilst considering interactions between instances, we can utilise Shapley values. When calculated with respect to a particular class, the Shapley value for an instance indicates if the instance supports the class (positive Shapley value), refutes the class (negative Shapley value), or is non-discriminatory with respect to the class (zero Shapley value). This means Shapley values can be used to answer *what* questions and satisfy the requirements we previously outlined in Section 3.1. The Shapley value ϕ_i for the instance $x_i \in X$ with respect to a particular class c is

$$\phi_i = \sum_{S \subseteq X \setminus \{x_i\}} \lambda \left[F_c(S \cup \{x_i\}) - F_c(S) \right], \tag{1}$$
$$\text{where } \lambda = \frac{|S|!(|X| - |S| - 1)!}{|X|!}.$$

The expensive computation of Equation 1 is the summation over the subsets $S \subseteq X \setminus \{x_i\}$. For each subset, two evaluations of the bag classification function F are made — one for S with the instance x_i, and one for S without the instance x_i. The difference in the model output for these two evaluations gives the contribution of the instance to the bag prediction, but only for the given subset S. To account for interactions between instances, this value must be calculated against all possible subsets. Calculating the Shapley values for the entire bag of k instances is $\mathcal{O}(k2^k)$, which is computationally intractable, even for small MIL datasets with only 30 instances per bag. As such, approximate methods have to be used instead, which can be computed in a reasonable amount of time but do not give exact Shapley values. In this work, we utilise SHAP to approximate the instance Shapley values.

There are a range of SHAP methods for approximating Shapley values (Lundberg & Lee, 2017). KernelSHAP is a model-agnostic SHAP method, so is appropriate for MIL model-agnostic explanation. Other SHAP methods are model-specific, i.e., they are tailored for a specific underlying model, therefore are not appropriate for model-agnostic interpretability. KernelSHAP samples coalitions of instances, and then fits a weighted linear regression model against the coalitions and their respective model predictions. A coalition is a binary vector $z \in \{0, 1\}^k$ that represents a subset $S = \{x_i | z_i = 1\}$. Given a collection of n coalitions Z, the weighted linear model g and corresponding loss function to be minimised L are

$$g(z) = \phi_0 + \sum_{i=1}^{k} \phi_i z_i$$
$$L(F_c, g, \pi) = \sum_{z \in Z} [F_c(S) - g(z)]^2 \pi(z),$$
$$\text{where } \pi(z) = \frac{k - 1}{\binom{k}{|z|} |z|(k - |z|)}. \tag{2}$$

Here, $|z|$ is the number of ones in the coalition (which is equivalent to the length of the subset S). The choice of weight kernel π means each coefficient $\phi_i \in \{\phi_1, \ldots, \phi_k\}$ is an approximation of the Shapley value for each instance $x_i \in X$. The benefit of this approach is that the Shapley values are jointly estimated, meaning the complexity of KernelSHAP when applied to MIL is $\mathcal{O}(nk^2)$, which is much more feasible to compute than the exact Shapley value calculation in Equation 1. An additional benefit afforded by using KernelSHAP for MIL comes from the ability of MIL models to deal with different sized bags. For single instance supervised learning models, it is not possible to simply 'remove' a feature from an input as the models expect fixed-size inputs — either a new model has to be re-trained without that particular feature, or appropriate sampling from other data has to be undertaken in order to 'zero out' the feature. This drawback does not occur for MIL, meaning SHAP-based methods work without retraining or needing to access any additional data. Given this approximation method, we consider three different methods for sampling coalitions:

Random sampling A benchmark approach is to sample coalitions at random. We can consider coalition formation as a repeated coin toss, where $P(z_i = 1) = p$ and $P(z_i = 0) = 1 - p$. For true random sampling, we will set $p = 0.5$, meaning $\mathbb{E}[|z|] = \frac{k}{2}$.

Guided sampling The original sampling approach as proposed by Lundberg & Lee (2017), which we term guided sampling, is to sample according to the SHAP weight kernel as shown in Equation 2, i.e., using it as a 'guide' on which coalitions to sample. Single instance coalitions and coalitions with only one instance removed receive the highest weighting according to this kernel, so those should be sampled first. By sampling according the SHAP weight kernel, we sample coalitions containing very few instances, i.e, different to the coalitions taken in random sampling.

Weighted sampling We propose a novel sampling approach termed weighted sampling. In this method, we use prior knowledge to improve upon random sampling by changing the value of p for each instance in the bag, meaning we can change the likelihood that each instance is involved in a coalition. We use the single instance method from Section 3.2 to get a prior understanding of the importance of each instance, and use that to weight the sampling. The independent-instance method produces values $\{\phi_1, \dots, \phi_k\}$, and we convert this to an instance ranking: the new values $r_i \in \{r_1, \dots, r_k\}$ represent the position for instance i (i.e., the instance with the greatest value for ϕ has $r = 0$, the second greatest has $r = 1$ etc.). Then, for each instance x_i, the sample probability p_i is

$$p_i = \frac{1 - 2\alpha}{k - 1} r_i + \alpha, \tag{3}$$

where $\alpha \in [0, 1]$ is a tunable hyperparameter. The value of α determines whether the sampling should be biased towards key instances or not: $\alpha > 0.5$ means higher p values are given to discriminatory samples (as determined by the values $\{\phi_1, \dots, \phi_k\}$), and $\alpha < 0.5$ means higher p values are given to non-discriminatory samples. When $\alpha = 0.5$, every value of $p_i = 0.5$, meaning we have the same sampling procedure as the random sampling method above.

Each of these SHAP-based methods is more expensive to compute that the methods outlined in Section 3.2: $\mathcal{O}(nk^2)$ compared with $\mathcal{O}(k)$. However, when there are interactions between the instances, this extra complexity is required in order to build a better understanding of the classes that each instance supports and refutes, allowing more accurate answering of *what* questions. Still, it is important that the SHAP-based methods are as sample efficient as possible, (i.e., by keeping n as small as possible). It is also important to note that each of these methods are indeed model agnostic — each method only requires access to the bag classification function F, and they make no assumptions about the underlying MIL model. As we demonstrate in the next section, this means each of these methods can be applied to any MIL model.

4 Experiments

We apply our model-agnostic methods to three MIL problems. In this section, we detail the datasets (Section 4.1), models (Section 4.2), evaluation strategy (Section 4.3), and results (Section 4.4). For implementation details, see Appendix A.1.

4.1 Datasets

SIVAL Although there exist a large number of benchmark MIL datasets, many do not have instance labels (Carbonneau et al., 2018). As ground truth instance labels are required to evaluate the efficacy of interpretability methods, this restricts the benchmark datasets we can use. The Spatially Independent, Variable Area, and Lighting (SIVAL; Rahmani et al. (2005)) dataset consists of 25 classes of complex objects photographed in different environments, where each class contains 60 images. Each image has been segmented into approximately 30 segments, and each segment is represented by a 30-dimensional feature vector that encodes information such as the colour and texture of the segment. Each segment is labelled as containing the complex object or containing background. We selected 12 of the 25 classes to be positive classes, and randomly sampled from each of the other 13 classes to form the negative class. For additional details see Appendix A.3.

Four-class MNIST-Bags The SIVAL dataset has no co-occurrence of instances from different classes, i.e., each bag only contains one class of positive instances. Therefore, from an interpretability perspective, the *which* and *what* questions are the same — if we can identify the key instances, then by the nature of the dataset we know which class they support. In order to explore what happens when there are different classes of positive instances in the same bag, we propose an extension of the MNIST-Bags dataset introduced by Ilse et al. (2018). Our extension, four-class MNIST-Bags (4-MNIST-Bags) is setup as follows: class 1 if **8** in bag, class 2 if **9** in bag, class 3 in **8** and **9** in bag, and class 0 otherwise[1]. Note that a bag can only have one label. In this dataset, answering *what* questions goes beyond answering *which* questions, e.g., the existence of an **8** supports classes one and three, but refutes classes zero and two. For further details see Appendix A.4.

Colon Cancer To test the applicability of the interpretability methods on larger bag sizes, we apply it to colorectal cancer tissue classification. The ColoRectal Cancer (CRC) dataset is a collection of microscopy images annotated with four classes of nuclei: epithelial, inflammatory, fibroblast, and miscellaneous (Sirinukunwattana et al., 2016). We follow the same setup as Ilse et al. (2018), in which a bag is positive if it contains one or more nuclei from the epithelial class. This means the problem conforms to the SMIL assumption, and there are no interactions between instances. Each microscopy image is 500 x 500 pixels, and was split into 27 x 27 pixel patches to give a maximum of 324 patches per slide (patches were discarded if they contained mostly slide background). Not all of the instances are labelled, so we tailor our assessment to only consider labelled instances, demonstrating how these techniques can still be evaluated without exhaustive instance labelling. For additional details see Appendix A.5.

4.2 Models

To highlight the model-agnostic abilities of the proposed methods, for each dataset, we trained four different types of multi-class MIL model: an embedding-based multiple instance neural network (MI-Net; Wang et al. (2018)), an instance-based multiple instance network (mi-Net; Wang et al. (2018)), an attention-based model (MI-Attn; Ilse et al. (2018)), and a graph neural network model (MI-GNN; Tu et al. (2019)). Aside from MI-Net, each of these models provide their own inherent interpretability method that we can compare our methods against. This means that in total we are evaluating seven different interpretability methods: inherent interpretability, the three independent-instance methods (Section 3.2) and the three SHAP-based methods (Section 3.3). Additional details on the models are given in Appendix A.2.

4.3 Evaluation

We do not assume that each of the interpretability methods has a consistent output domain; we only assume a larger value implies larger support. Therefore, the best approach for evaluating the interpretability methods is to use ranking metrics — rather than looking at the absolute values produced by the methods, we compare the relative orderings they produce. For a particular class, we can place each instance into one of three groups based on their ground truth instance labels: supporting instances, neutral instances, and refuting instances. The ideal instance ordering for that class would then have all of the supporting instances at the beginning, followed by all the neutral instances, and then all of the refuting instances at the end. Given this ground truth ordering, we can then use existing ranking metrics to evaluate how well the different interpretability methods answer *what* questions. The particular metric we use is normalised discounted cumulative gain at n (NDCG@n), which prioritises instances higher in the produced ranking by using a logarithmic weighting. Therefore, methods are heavily penalised if they mistake refuting instances for supporting instances. We set n equal to the number of instances in the bag that support the class in question. Further details are given in Appendix A.6.

4.4 Results

For each dataset, we measured the performance of each interpretability method on each of the four MIL models. For the SIVAL dataset we measured the performance on only the negative class and the

[1]We use **n** to refer to an image from the MNIST dataset that represents the number n, even though that is not the assigned class label in the 4-MNIST-Bags dataset.

bag's true class, but for the 4-MNIST-Bags and CRC datasets we evaluated the interpretability over every class. This distinction was made as we know that each SIVAL bag can only contain instances from one positive class, therefore it is not necessary to evaluate over all possible classes. We present the interpretability results run against the test set for the SIVAL, 4-MNIST-Bags, and CRC datasets in Tables 1, 2 and 3 respectively. The results are averaged over ten repeat trainings of each model, and the standard error of the mean is given in each case. We also provide the test accuracy of each of the underlying MIL models.

Table 1: SIVAL interpretability results. For the SHAP-based methods, number of samples $= 200$, and for WeightedSHAP, $\alpha = 0.05$.

Method	MI-Net	mi-Net	MI-Attn	MI-GNN	Overall
Model Acc	0.819 ± 0.012	0.808 ± 0.011	0.813 ± 0.012	0.781 ± 0.019	0.805 ± 0.007
Inherent	N/A	0.813 ± 0.005	0.717 ± 0.006	0.586 ± 0.009	0.705 ± 0.007
Single	0.825 ± 0.005	0.813 ± 0.005	0.801 ± 0.005	0.734 ± 0.006	0.793 ± 0.005
Leave one out	0.778 ± 0.005	$\mathbf{0.837 \pm 0.005}$	0.736 ± 0.006	0.776 ± 0.006	0.782 ± 0.005
Combined	$\mathbf{0.828 \pm 0.005}$	0.828 ± 0.005	$\mathbf{0.803 \pm 0.005}$	0.762 ± 0.006	$\mathbf{0.805 \pm 0.005}$
RandomSHAP	0.800 ± 0.005	0.808 ± 0.005	0.768 ± 0.005	0.786 ± 0.005	0.790 ± 0.005
GuidedSHAP	0.826 ± 0.005	0.819 ± 0.005	0.790 ± 0.005	0.765 ± 0.006	0.800 ± 0.005
WeightedSHAP	0.819 ± 0.004	0.823 ± 0.004	0.789 ± 0.005	$\mathbf{0.789 \pm 0.005}$	$\mathbf{0.805 \pm 0.005}$

Table 2: 4-MNIST-Bags interpretability results. For the SHAP-based methods, the number of samples $= 150$, and for WeightedSHAP, $\alpha = 0.1$.

Method	MI-Net	mi-Net	MI-Attn	MI-GNN	Overall
Model Acc	0.971 ± 0.002	0.974 ± 0.002	0.967 ± 0.003	0.966 ± 0.002	0.970 ± 0.002
Inherent	N/A	0.723 ± 0.008	0.750 ± 0.010	0.419 ± 0.009	0.630 ± 0.009
Single	0.722 ± 0.008	0.723 ± 0.008	0.778 ± 0.007	0.761 ± 0.008	0.746 ± 0.008
Leave one out	0.811 ± 0.007	0.810 ± 0.007	0.809 ± 0.006	0.786 ± 0.007	0.804 ± 0.007
Combined	0.775 ± 0.007	0.775 ± 0.007	0.816 ± 0.007	0.804 ± 0.007	0.792 ± 0.007
RandomSHAP	0.810 ± 0.006	0.807 ± 0.006	0.823 ± 0.006	0.826 ± 0.006	0.816 ± 0.006
GuidedSHAP	0.773 ± 0.007	0.773 ± 0.007	0.816 ± 0.007	0.805 ± 0.007	0.792 ± 0.007
WeightedSHAP	$\mathbf{0.928 \pm 0.003}$	$\mathbf{0.922 \pm 0.004}$	$\mathbf{0.907 \pm 0.004}$	$\mathbf{0.948 \pm 0.003}$	$\mathbf{0.926 \pm 0.003}$

Table 3: CRC interpretability results. For the SHAP-based methods, the number of samples $= 1000$, and for WeightedSHAP, $\alpha = 0.1$.

Method	MI-Net	mi-Net	MI-Attn	MI-GNN	Overall
Model Acc	0.795 ± 0.042	0.795 ± 0.049	0.830 ± 0.028	0.770 ± 0.032	0.797 ± 0.011
Inherent	N/A	$\mathbf{0.845 \pm 0.010}$	0.692 ± 0.020	0.684 ± 0.017	0.740 ± 0.016
Single	$\mathbf{0.815 \pm 0.012}$	$\mathbf{0.845 \pm 0.010}$	$\mathbf{0.847 \pm 0.010}$	0.786 ± 0.014	0.823 ± 0.012
Leave one out	0.698 ± 0.016	0.682 ± 0.017	0.701 ± 0.017	$\mathbf{0.815 \pm 0.012}$	0.724 ± 0.016
Combined	$\mathbf{0.815 \pm 0.012}$	$\mathbf{0.845 \pm 0.010}$	0.846 ± 0.010	0.803 ± 0.013	$\mathbf{0.827 \pm 0.011}$
RandomSHAP	0.695 ± 0.016	0.690 ± 0.016	0.717 ± 0.015	0.703 ± 0.016	0.701 ± 0.015
GuidedSHAP	$\mathbf{0.815 \pm 0.012}$	$\mathbf{0.845 \pm 0.010}$	0.846 ± 0.010	0.804 ± 0.013	$\mathbf{0.827 \pm 0.011}$
WeightedSHAP	0.699 ± 0.015	0.707 ± 0.015	0.717 ± 0.015	0.702 ± 0.016	0.706 ± 0.016

In the majority of cases, the six methods proposed in this work outperform the existing inherent interpretability methods, and are relatively consistent in performance across all models. For the SIVAL

dataset, the performance of the independent-instance methods is comparable to that of the SHAP-based methods. This is expected, as the SIVAL dataset has a lack of instance interactions, meaning the use of SHAP will not uncover any additional information on instance importance. However, on the MNIST dataset, where there are interactions between instances, the SHAP-based methods are better as they able to identify the instance interactions. In particular, the WeightedSHAP method excels, as it is able to sample more informative coalitions than the other SHAP-based methods.

On the CRC dataset, we observe that the single, combined, and GuidedSHAP methods perform the best. In this dataset, the instances are largely independent, therefore examining each instance in isolation is sufficient to understand which class it supports. This means methods that are able to isolate individual instances, (i.e., the single, combined, and GuidedSHAP methods) perform well. Also, if we consider the witness rate (WR; the proportion of key instances in each bag; Carbonneau et al. (2018)) of the datasets, we find that the CRC dataset has a higher WR (27.47%) than the SIVAL (15.28%) and 4-MNIST-Bags datasets (8.04%). This means, with larger coalitions, it becomes more difficult to isolate the contributions of individual instances, which is why the RandomSHAP and WeightedSHAP methods struggle. However, larger coalitions are beneficial when there are interactions between the instances. In the next section, we analyse the SHAP-based methods in more detail to understand how they identify interactions between instances.

5 Discussion

The SHAP-based methods are particularly effective on the 4-MNIST-Bags dataset, i.e., when there are interactions between the instances. To understand why, and to show the limitations of existing methods that cannot condition their output for a particular class, we examine the outputs of the interpretability methods on the 4-MNIST-Bags dataset. In Figure 1 we compare the WeightedSHAP output with the attention output. As this bag contains an **8** and a **9**, the correct label for it is class three. Both the WeightedSHAP and attention methods have identified the **8** and **9** instances as key instances, i.e., they have both successfully answered the *which* question. However, WeightedSHAP has correctly identified that the **8** refutes class two and that the **9** refutes class one, effectively answering the *what* question, something that the attention values do not do. We provide further examples, including interpretability outputs for the SIVAL and CRC datasets, in Appendix A.7.

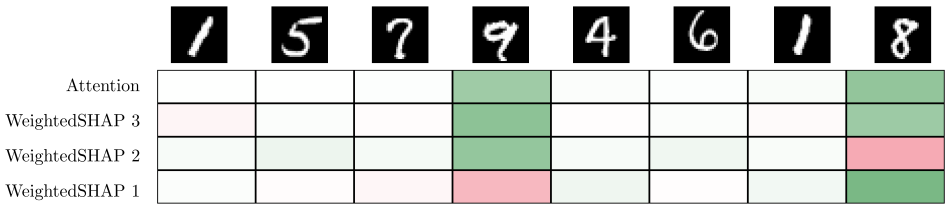

Figure 1: An example of the identification of key instances for the 4-MNIST-Bags experiment. The top row shows the attention value for each instance, and bottom three rows show the output of WeightedSHAP for classes three, two, and one respectively (green = support, red = refute).

The WeightedSHAP method includes an additional parameter α that determines whether the sampling should be biased towards discriminatory or non-discriminatory instances. As shown in Figure 2, when tuning the parameter on the 4-MNIST-Bags dataset, the optimum value for α is around 0.1, meaning it is quite heavily biased towards sampling non-discriminatory instances. The intuition for this is that it is relatively easy to understand the effect of discriminatory instances as they have a larger impact on the model's prediction, whereas the non-discriminatory instances have a much smaller impact. Therefore, a greater number of coalitions containing the non-discriminatory instances are required to fully understand the impact of those instances on the model outcome.

For all of the SHAP-based methods, it is necessary to choose the number of sampled coalitions. The greater the number of coalitions, the more accurate the Shapley value approximation. However, the more samples that are taken, the more expensive it is to compute the Shapley values. As shown in Figure 3, on the 4-MNIST-Bags dataset, the WeightedSHAP method is more sample efficient than the other SHAP-based methods. For all methods, especially GuidedSHAP, it is a case of diminishing returns, where additional samples only lead to a small increase in performance.

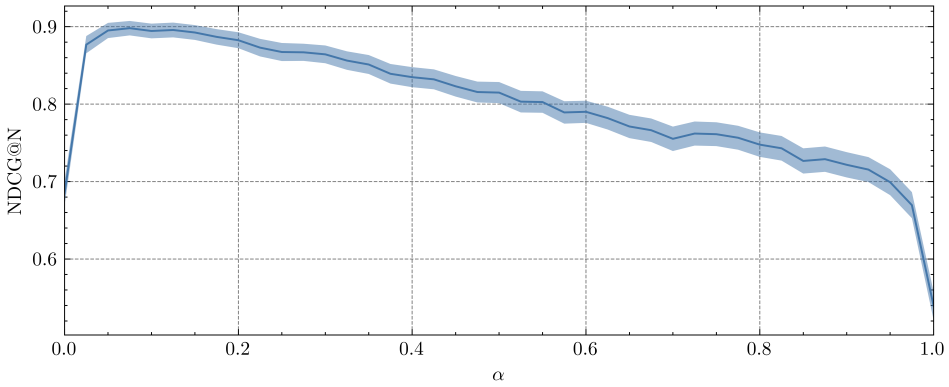

Figure 2: The effect of α on interpretability performance for WeightedSHAP.

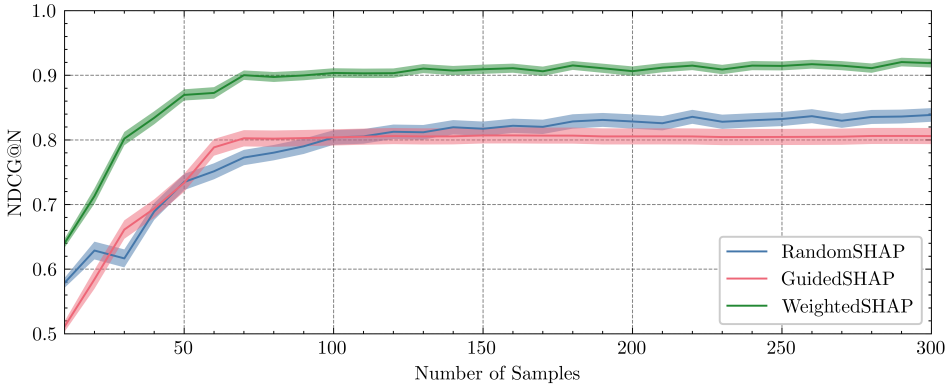

Figure 3: The effect of sample size on interpretability performance for the SHAP-based methods.

In both of these figures, the results are averaged over ten repeat trainings of the MI-Attn model, and the shaded area represents the standard error of the mean. We also carried out the same studies on the SIVAL and CRC datasets; see Appendix A.8.

The different sampling approaches for the SHAP-based methods have distinct advantages. As shown on the 4-MNIST-Bags dataset, the WeightedSHAP method is effective when there are interactions between the instances. However, it performs poorly on the CRC dataset, where there are a large number of independent instances and a high WR. Here, the GuidedSHAP method performs well, as it forms very small coalitions, so is able to isolate the effect of each instance. In the future, better sampling approaches could be developed that combine the benefits of WeightedSHAP and GuidedSHAP. In addition, there are other SHAP-based sampling approaches, such as paired sampling (Covert & Lee, 2020), which could be incorporated to improve the SHAP-based methods for MIL interpretability. Following the advice of Carbonneau et al. (2018), further MIL studies on more complex datasets, such as Pascal VOC (Everingham et al., 2010), could also be insightful for developing MIL models and further evaluating MIL interpretability methods.

6 Conclusion

In this work, we have discussed the process of model-agnostic interpretability for MIL. Along with defining the requirements for MIL interpretability, we have presented our own approaches and compared them to existing inherently interpretable MIL models. By analysing the methods across three datasets, we have shown that independent-instance methods can be effective, but SHAP-based methods are required when there are interactions between the instances. All of our proposed methods are more effective than existing inherently interpretable models, and are able to not only identify *which* are the key instances, but also say *what* classes they support and refute.

REPRODUCIBILITY STATEMENT

For our work, the main details for the methodology are detailed in Section 3. In addition, we provide details that will aid reproducibility in the Appendix. The dataset sources and a general overview of our implementation is given in Appendix A.1. Further information on the MIL models used in this work can be found in Appendix A.2, and specific details on their architectures and training can be found in Appendices A.3, A.4, and A.5 for the SIVAL, 4-MNIST-Bags, and CRC datasets respectively. The complete codebase is included as supplementary material for this submission, and will also be made publicly available on GitHub at the time of publication.

ACKNOWLEDGEMENTS

We used SciencePlots (Garrett, 2021) for formatting our Matplotlib figures.

REFERENCES

Jaume Amores. Multiple instance classification: Review, taxonomy and comparative study. *Artificial intelligence*, 201:81–105, 2013.

Marc-André Carbonneau, Veronika Cheplygina, Eric Granger, and Ghyslain Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77: 329–353, 2018.

Ian Covert and Su-In Lee. Improving kernelshap: Practical shapley value estimation via linear regression. *arXiv preprint arXiv:2012.01536*, 2020.

Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1-2):31–71, 1997.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2): 303–338, 2010.

John D. Garrett. garrettj403/SciencePlots. September 2021. doi: 10.5281/zenodo.4106649. URL `http://doi.org/10.5281/zenodo.4106649`.

Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE, 2018.

Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pp. 2127–2136. PMLR, 2018.

Sirui Li, Weixing Sun, and Tim Miller. Communication in human-agent teams for tasks with joint action. In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems*, pp. 224–241. Springer, 2015.

Guoqing Liu, Jianxin Wu, and Zhi-Hua Zhou. Key instance detection in multi-instance learning. In *Asian Conference on Machine Learning*, pp. 253–268. PMLR, 2012.

Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.

Oded Maron and Tomás Lozano-Pérez. A framework for multiple-instance learning. *Advances in neural information processing systems*, pp. 570–576, 1998.

Luke Merrick and Ankur Taly. The explanation game: Explaining machine learning models using shapley values. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pp. 17–38. Springer, 2020.

Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

Rouhollah Rahmani, Sally A Goldman, Hui Zhang, John Krettek, and Jason E Fritts. Localized content based image retrieval. In *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*, pp. 227–236, 2005.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

Stephen Scott, Jun Zhang, and Joshua Brown. On generalized multiple-instance learning. *International Journal of Computational Intelligence and Applications*, 5(01):21–35, 2005.

Lloyd S Shapley. *A value for n-person games*. Princeton University Press, 1953.

Korsuk Sirinukunwattana, Shan E Ahmed Raza, Yee-Wah Tsang, David RJ Snead, Ian A Cree, and 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):1196–1206, 2016.

Ming Tu, Jing Huang, Xiaodong He, and Bowen Zhou. Multiple instance learning with graph neural networks. *arXiv preprint arXiv:1906.04881*, 2019.

Xinggang Wang, Yongluan Yan, Peng Tang, Xiang Bai, and Wenyu Liu. Revisiting multiple instance neural networks. *Pattern Recognition*, 74:15–24, 2018.

Nils Weidmann, Eibe Frank, and Bernhard Pfahringer. A two-level learning method for generalized multi-instance problems. In *European Conference on Machine Learning*, pp. 468–479. Springer, 2003.

Zhi-Hua Zhou, Yu-Yin Sun, and Yu-Feng Li. Multi-instance learning by treating instances as non-iid samples. In *Proceedings of the 26th annual international conference on machine learning*, pp. 1249–1256, 2009.

A APPENDIX

A.1 IMPLEMENTATION DETAILS

All code for this work was implemented in Python 3.8, using the PyTorch library for the machine learning functionality. Hyperparameter tuning was carried out using the Optuna libary. Some local experiments were carried out on a Dell XPS Windows laptop, utilising a GeForce GTX 1650 graphics card with 4GB of VRAM. GPU support for machine learning was enabled through CUDA v11.0. Other longer running experiments, such as hyperparameter tuning, were carried out on a remote GPU node utilising a Volta V100 Enterprise Compute GPU with 16GB of VRAM. The longest model to train was the GNN for the CRC dataset, which took just under half an hour. The longest running experiment was the hyperparameter analysis for the CRC dataset at just under 40 hours. This was due to high number of samples for the SHAP-based methods and the fact that the results were average over 10 different models. The randomisation of data splits was fixed using seeding; the seed values are provided in the code for each experiment.

We use the following sources for our data:

- The annotated SIVAL dataset was downloaded from the publicly accessible page `http://pages.cs.wisc.edu/~bsettles/data/`.
- The MNIST dataset was access directly from the PyTorch Python library.
- The CRC dataset was downloaded from the publicly accessible page `https://warwick.ac.uk/fac/cross_fac/tia/data/crchistolabelednucleihe/`.

A.2 MODELS

In this work, we trained four different models for each dataset. In this section, we go into more detail about each of the models. Each model was tuned independently for each dataset, so in the later sections we provide details of the specific architectures for each dataset. We also tuned the learning rate, weight decay, and dropout for each of the models independently for every dataset. Again, these are given in the later sections for each specific dataset.

MI-Net Each instance is embedded to a fixed size, and then these embeddings are aggregated to give a single bag embedding. This aggregation can either take the mean (mil-mean) or max (mil-max). The bag embedding is then classified to give an overall bag prediction. For this model, we tuned the number and size of fully connected (FC) layers, as well as the choice of aggregation function.

mi-Net A prediction is made for each individual instance, and then these are aggregated to a single outcome for the bag as a whole. Similar to the MI-Net model, we tuned the FC layers and the aggregation function.

MI-Attn Each instance is embedded to a fixed size, and then the mil-attn block produces an attention value for each instance embedding. The instance embeddings are then aggregated to a single bag embedding by performing a weighted sum based on the attention values. The bag embedding is then classified to give an overall bag prediction. The mil-attn block is the same as per the MIL attention mechanism proposed by Ilse et al. (2018), i.e., using a single hidden layer. We tuned the the number and size of the FC layers, as well as the size of the hidden attention layer.

MI-GNN The GNN model treats the bag as a fully connected graph and uses graph convolutions to propagate information between instances. Initially, the original instances are embedded to a fixed size. These instance embeddings are then passed onto a $\text{GNN}_{\text{embed}}$ block and a $\text{GNN}_{\text{cluster}}$ block, the output of which is used to reduce the graph representation down into a single embedding using differentiable pooling (gnn-pool). The final bag representation is then classified to give an overall bag prediction. For more details see Tu et al. (2019). We tuned the number and size of the embedding, GNN, and classifier layers.

A.3 SIVAL EXPERIMENT DETAILS

Dataset For the SIVAL dataset, each instance is represented by a 30-dimensional feature vector, and there are around 30 instances per bag. We chose 12 of the 25 original classes to be the positive classes, and randomly selected 30 images from each of the other 13 classes to form the negative class, meaning overall we had 13 classes (12 positive and one negative). In total, we had 60 bags for each of the 12 positive classes, and 390 bags for the single negative class, meaning the class distribution was $\approx 5.4\%$ for each positive class and $\approx 35.1\%$ for the negative class. The arbitrarily chosen positive classes were: *apple*, *banana*, *checkeredscarf*, *cokecan*, *dataminingbook*, *goldmedal*, *largespoon*, *rapbook*, *smileyfacedoll*, *spritecan*, *translucentbowl*, and *wd40can*. We normalised each instance according to the dataset mean and standard deviation. No other data augmentation was carried out. The dataset was separated into train, validation, and test data using an 80/10/10 split. This was done with stratified sampling in order to maintain the same data distribution across all splits.

Training When training models against the SIVAL dataset, we used a batch size of one, i.e., a single bag of, on average, 30 instances. We trained the models to minimise cross entropy loss using the Adam optimiser; the hyperparamater details are given in Table A1. We utilised early stopping based on validation loss — if the validation loss had not decreased for 10 epochs then we terminated the training procedure and used the model that caused the last decrease. Otherwise, the maximum number of epochs was 100. The tuned architectures for each model are given in Tables A2 to A5, and the results for each model are comapred in Table A6.

Table A1: SIVAL training hyperparameters.

Model	Learning Rate	Weight Decay	Dropout
Mi-Net	5×10^{-3}	1×10^{-3}	0.45
mi-Net	5×10^{-4}	1×10^{-5}	0.25
MI-Attn	1×10^{-3}	1×10^{-5}	0.15
MI-GNN	5×10^{-4}	1×10^{-5}	0.2

Table A2: SIVAL MI-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	30	128
2	FC + ReLU + Dropout	128	256
3	mil-max	256	256
4	FC	256	13

Table A3: SIVAL mi-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	30	512
2	FC + ReLU + Dropout	512	256
3	FC + ReLU + Dropout	256	64
4	FC	64	13
5	mil-mean	13	13

Table A4: SIVAL MI-Attn model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	30	128
2	FC + ReLU + Dropout	128	256
3	FC + ReLU + Dropout	256	128
4	mil-attn(256) + Dropout	128	128
5	FC	128	13

Table A5: SIVAL MI-GNN model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	30	128
2a $\text{GNN}_{\text{embed}}$	SAGEConv + ReLU + Dropout	128	128
	SAGEConv + ReLU + Dropout	128	256
	SAGEConv + ReLU + Dropout	256	64
2b $\text{GNN}_{\text{cluster}}$	SAGEConv + Softmax	128	1
3	gnn-pool	64	64
4	FC + ReLU + Dropout	64	128
5	FC	128	13

Table A6: SIVAL model results. The mean performance was calculated over ten repeat trainings of each model, and the standard error of the mean is given.

Model	Train Accuracy	Val Accuracy	Test Accuracy
MI-Net	**0.984 ± 0.004**	**0.850 ± 0.009**	**0.819 ± 0.012**
mi-Net	0.967 ± 0.005	0.835 ± 0.010	0.808 ± 0.011
MI-Attn	0.972 ± 0.007	0.830 ± 0.011	0.813 ± 0.012
MI-GNN	0.932 ± 0.014	0.803 ± 0.014	0.781 ± 0.019

A.4 4-MNIST-BAGS EXPERIMENT DETAILS

Dataset In the 4-MNIST-Bags experiments, the bag sizes were draw from a normal distribution, with a mean of 30 and a variance of 2. We used 2500 training bags, 1000 validation bags, and 1000 test bags. The instances in the training bags were only drawn from the original MNIST training split, and the instances in the validation and test bags were only drawn from the original MNIST test split, i.e., there was no overlap between training, validation, and test instances. The classes were balanced, so, on average, there were 625 bags per class in the training data, and 250 bags per class in the validation and test data. We normalised the MNIST images using the PyTorch normalise transformation, with a mean of 0.1307 and a stand deviation of 0.3081. No other data augmentation was carried out.

Training The training procedure was the same as for the SIVAL experiments: a batch size of one, early stopping with a patience of ten, and a maximum of 100 epochs. The training hyperparamater details are given in Table A7. For each model, we first passed the MNIST instances through a convolutional architecture to produce initial instance embeddings. This encoder was not tuned for each model (i.e., the architecture was fixed, but the weights were learnt). The architecture for this encoder is given in Table A8, and then the model architectures are given in Tables A9 to A12. The encoder produces features vectors with 800 features, therefore the input size to each of the models is 800. The model results are given in Table A13.

Table A7: 4-MNIST-Bags training hyperparameters.

Model	Learning Rate	Weight Decay	Dropout
MI-Net	1×10^{-4}	1×10^{-3}	0.3
mi-Net	1×10^{-4}	1×10^{-4}	0.3
MI-Attn	1×10^{-4}	1×10^{-4}	0.15
MI-GNN	5×10^{-5}	1×10^{-5}	0.3

Table A8: 4-MNIST-Bags convolutional encoding architecture. For the convolutional (Conv2d) and pooling (MaxPool2d) layers, the numbers in the brackets are the kernel size, stride, and padding.

Layer	Type	Input channels	Out channels
1	Conv2d(5, 1, 0) + ReLU	1	20
2	MaxPool2d(2, 2, 0) + Dropout	20	20
3	Conv2d(5, 1, 0) + ReLU	20	50
4	MaxPool2d(2, 2, 0) + Dropout	50	50

Table A9: 4-MNIST-Bags MI-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	800	128
2	FC + ReLU + Dropout	128	512
3	mil-mean	512	512
4	FC	512	4

Table A10: 4-MNIST-Bags mi-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	800	512
2	FC + ReLU + Dropout	512	128
3	FC + ReLU + Dropout	128	64
4	FC	128	4
5	mil-mean	4	4

Table A11: 4-MNIST-Bags MI-Attn model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	800	64
2	FC + ReLU + Dropout	64	256
3	mil-attn(64) + Dropout	256	256
4	FC + ReLU + Dropout	256	64
5	FC	64	4

Table A12: 4-MNIST-Bags MI-GNN model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	800	64
2	FC + ReLU + Dropout	64	64
3a $\text{GNN}_{\text{embed}}$	SAGEConv + ReLU + Dropout	64	128
	SAGEConv + ReLU + Dropout	128	128
	SAGEConv + ReLU + Dropout	128	128
3b $\text{GNN}_{\text{cluster}}$	SAGEConv + Softmax	64	1
4	gnn-pool	128	128
5	FC + ReLU + Dropout	128	64
6	FC	64	4

Table A13: 4-MNIST-Bags model results. The mean performance was calculated over ten repeat trainings of each model, and the standard error of the mean is given.

Model	Train Accuracy	Val Accuracy	Test Accuracy
MI-Net	**0.995 ± 0.001**	0.972 ± 0.002	0.971 ± 0.002
mi-Net	0.993 ± 0.001	**0.973 ± 0.002**	**0.974 ± 0.002**
MI-Attn	0.991 ± 0.002	0.967 ± 0.003	0.967 ± 0.003
MI-GNN	0.984 ± 0.002	0.968 ± 0.002	0.966 ± 0.002

A.5 CRC EXPERIMENT DETAILS

Dataset For the CRC dataset, we have 100 images, with 50 in the negative class (no epithelial nuclei), and 50 in the positive class (at least one epithelial nuclei). Each image is 500 x 500 pixels, and was split into 27 x 27 pixel patches to give 324 patches per slide. The patches were created by applying a non-overlapping grid over the image, where each cell of the grid was a 27 x 27 pixel region. The alternative would be to extract pages centred on each marked nuclei (as done by Ilse et al. (2018)), however this method then requires the nuclei to be marked for unseen data. This accounts for the difference between our results and the results of Ilse et al. (2018) — extracting the patches using a fixed grid will include patches that do not contain any marked nuclei, increasing the amount of non-discriminatory data in each bag and thus making the problem harder to learn. We removed slide background patches by using a brightness threshold — any patch with an average pixel value above 230 (using pixel values 0 to 255) was discarded. This left an average of 264 patches per image, and one image was discarded as it had zero foreground patches (this was also manually verified). The images were normalised using the dataset mean (0.8035, 0.6499, 0.8348) and standard deviation (0.0858, 0.1079, 0.0731). During training, we also applied three transformations to patches at random: horizontal flips, vertical flips, and 90 degree rotations. The dataset was separated into train, validation, and test data using a 60/20/20 split. This was done with stratified sampling in order to maintain the same data distribution across all splits.

Training The training procedure was the same as for the SIVAL and 4-MNIST-Bags experiments: a batch size of one, early stopping with a patience of ten, and a maximum of 100 epochs. The training hyperparamater details are given in Table A14. Following the same procedure as the 4-MNIST-Bags experiments, for each model, we first passed the patches through a convolutional architecture to produce initial instance embeddings. The architecture for this encoder is given in Table A15, and then the model architectures are given in Tables A16 to A19. The encoder produces features vectors with 1200 features, therefore the input size to each of the models is 1200. The model results are given in Table A20.

Table A14: CRC training hyperparameters.

Model	Learning Rate	Weight Decay	Dropout
MI-Net	5×10^{-4}	1×10^{-3}	0.3
mi-Net	5×10^{-4}	1×10^{-2}	0.25
MI-Attn	1×10^{-3}	1×10^{-6}	0.2
MI-GNN	1×10^{-3}	1×10^{-2}	0.35

Table A15: CRC convolutional encoding architecture. For the convolutional (Conv2d) and pooling (MaxPool2d) layers, the numbers in the brackets are the kernel size, stride, and padding.

Layer	Type	Input channels	Out channels
1	Conv2d(4, 1, 0) + ReLU	3	36
2	MaxPool2d(2, 2, 0) + Dropout	36	36
3	Conv2d(3, 1, 0) + ReLU	36	48
4	MaxPool2d(2, 2, 0) + Dropout	48	48

Table A16: CRC MI-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	1200	64
2	FC + ReLU + Dropout	64	512
3	mil-max	512	512
4	FC + ReLU + Dropout	512	128
4	FC + ReLU + Dropout	128	64
4	FC + ReLU + Dropout	64	2

Table A17: CRC mi-Net model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	1200	64
2	FC + ReLU + Dropout	64	64
3	FC + ReLU + Dropout	64	64
4	FC	64	2
5	mil-max	2	2

Table A18: CRC MI-Attn model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	1200	64
2	FC + ReLU + Dropout	64	64
3	FC + ReLU + Dropout	64	256
4	mil-attn(128) + Dropout	256	256
5	FC	256	2

Table A19: CRC MI-GNN model architecture.

Layer	Type	Input size	Output size
1	FC + ReLU + Dropout	1200	64
2	FC + ReLU + Dropout	64	128
3a $\text{GNN}_{\text{embed}}$	SAGEConv + ReLU + Dropout	128	128
3b $\text{GNN}_{\text{cluster}}$	SAGEConv + Softmax	128	1
4	gnn-pool	128	128
5	FC + ReLU + Dropout	128	128
6	FC	128	2

Table A20: CRC model results. The mean performance was calculated over ten repeat trainings of each model, and the standard error of the mean is given.

Model	Train Accuracy	Val Accuracy	Test Accuracy
MI-Net	**0.880 ± 0.041**	0.805 ± 0.034	0.795 ± 0.042
mi-Net	0.856 ± 0.041	0.810 ± 0.043	0.795 ± 0.049
MI-Attn	0.870 ± 0.014	**0.860 ± 0.017**	**0.830 ± 0.028**
MI-GNN	0.791 ± 0.039	0.765 ± 0.031	0.770 ± 0.032

A.6 Evaluation metric

In Section 4.3 we describe how we use normalised discounted cumulative gain at n (NDCG@n) to evaluate the interpretability methods. Here, we provide further detail on this ranking metric. Given interpretability outputs $\{\phi_1, \ldots, \phi_k\}$ for a particular class, we rank the outputs from highest to lowest, i.e., in order of how much they support that class. Then, we evaluate this ranking according to the metric:

$$\text{NDCG@n} = \frac{1}{\text{IDCG}} \sum_{i=1}^{n} \frac{\text{rel}(i)}{log_2(i+1)},$$
$$\text{where IDCG} = \sum_{i=1}^{n} \frac{1}{log_2(i+1)}.$$

IDCG is the ideal discounted cumulative gain that normalises the scores across different values of n. The relevance function $\text{rel}(i)$ is as follows:

$$\text{rel}(i) = \begin{cases} 1 & \text{if the } i^{\text{th}} \text{ instance in the ranking supports the class,} \\ -1 & \text{if the } i^{\text{th}} \text{ instance in the ranking refutes the class,} \\ 0 & \text{otherwise.} \end{cases}$$

A.7 Additional outputs

In this section, we provide additional interpretability outputs from our studies, similar to the one in Section 5. First, we provide additional outputs for the 4-MNIST-Bags dataset. Then, we show the interpretability outputs for the SIVAL and CRC datasets. The SIVAL outputs are created by ranking the instances in a bag according to some interpretability function (e.g., by using WeightedSHAP), and then selecting the top n, where n is the known number of key instances in the bag. Then, the corresponding segment for each instance is weighted by the function $log_2(i+1)^{-1}$, where i is the instance's position in the ranking (this is the same scaling used in NDCG@n). The other instances all received a weighting of zero. The brightness of each segment in the output image then corresponds with its relative importance according to the interpretability method. For the CRC interpretability

outputs, the important patches are found by using an interpretability method to output the top n patches that support a particular class, where n is the number of known important instances for that class. We then highlight these patches while dimming the other patches to produce an interpretation of the model's decision-making.

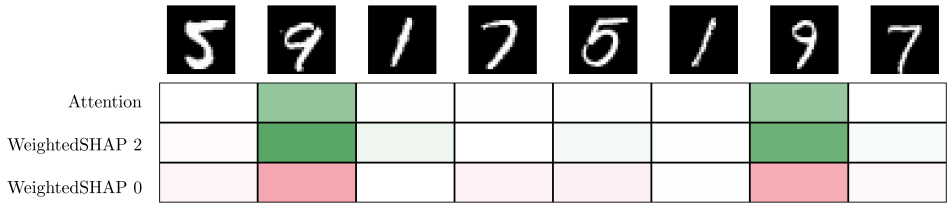

Figure A1: The interpretability output for a class two bag on the 4-MNIST-Bags experiment. The top row shows the attention value for each instance, and bottom two rows show the output of WeightedSHAP for classes two, and zero respectively (green = support, red = refute). We can see that both the attention method and WeightedSHAP have identified the nines as important, but only WeightedSHAP is able to also say that the nines support class two and refute class zero.

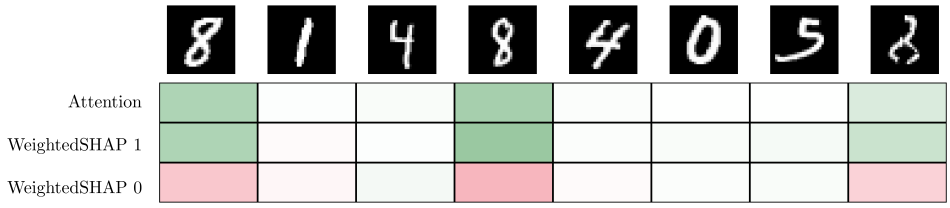

Figure A2: The interpretability output for a class one bag on the 4-MNIST-Bags experiment. We can see that both the attention method and WeightedSHAP have identified the eights as important, but only WeightedSHAP is able to also say that the eights support class one and refute class zero. Also note that the colours are less strong for the final eight — the digit is partially obscured, so the model is less confident.

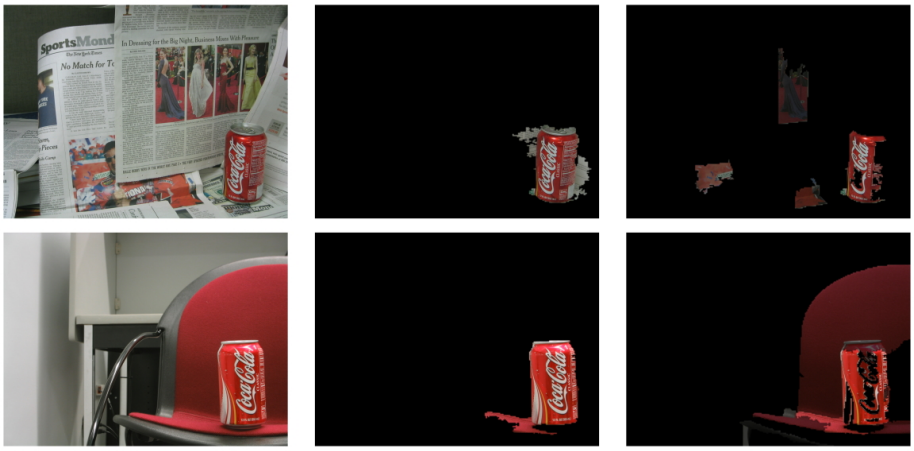

Figure A3: The interpretability output for two examples of the *cokecan* class in the SIVAL dataset. The first column shows the original images, the second column the ground truth segments that contain the object, and the final column is the output from WeightedSHAP. The interpretations show the model is relying heavily on the colour red as an indicator of the object's location, as it also picks up on the red in the background as being important.

Figure A4: The interpretability output for two examples of the *dataminingbook* class in the SIVAL dataset. Again, the interpretations show the model is relying heavily on the colour orange as an indicator of the object's location, as it also picks up on the similar colours in the background, including the reflection of the book in the table.

Figure A5: The interpretability output for two examples of the *goldmedal* class in the SIVAL dataset. Here, the interpretations show the model is ignoring the gold medal itself, and instead focusing on the red ribbon, i.e., if it were shown the medal without the ribbon, it would likely misclassify it.

Figure A6: The interpretability output for two examples of the *wd40can* class in the SIVAL dataset. Here, the interpretations show the model is predominantly focusing on the strong yellow colour at the top of the can, and sometimes picks out the letters and red cap. The rest of the bottle is largely ignored, meaning if the top half were obscured, the model would likely misclassify it.

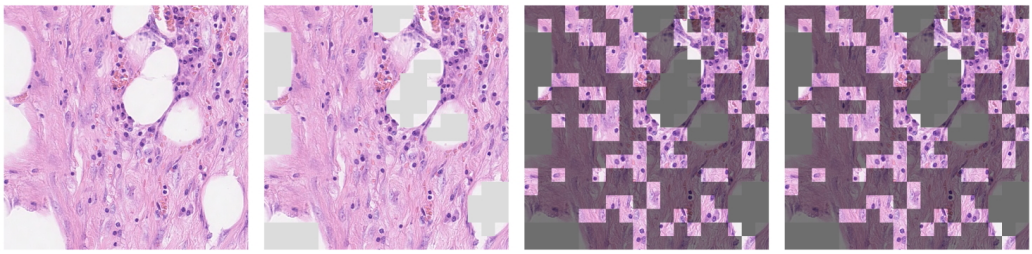

Figure A7: The interpretability output for an image in the CRC dataset. From left to right, the figures shows: the original image, the image with background patches removed, the ground truth patches for the image's label, and the interpretability output from GuidedSHAP. In this example, the image is class zero (i.e., non-epithelial), meaning the highlighted patches contain mostly fibroblast and inflammatory nuclei. As shown here, GuidedSHAP has identified that the model is using the same patches as the ground truth to make a decision, indicating that the model has learnt to correctly identify fibroblast and inflammatory nuclei as supporting instances for non-epithelial images.

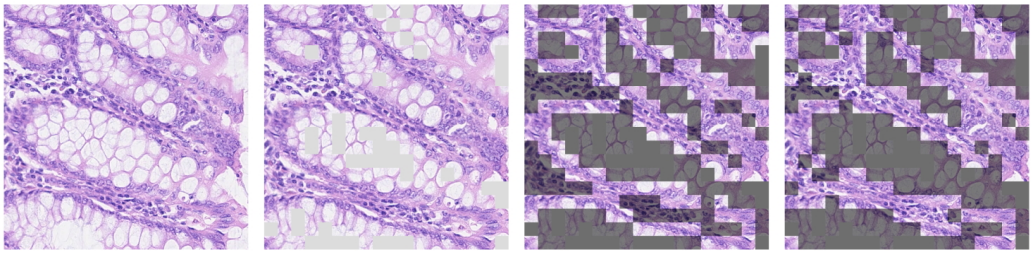

Figure A8: The interpretability output for a positive image in the CRC dataset. In this example, the ground truth patches all contain at least one epithelial nuclei. GuidedSHAP has identified that the model is using most of the same patches as the ground truth to make a decision, indicating that the model has learnt to correctly identify epithelial nuclei as supporting class one.

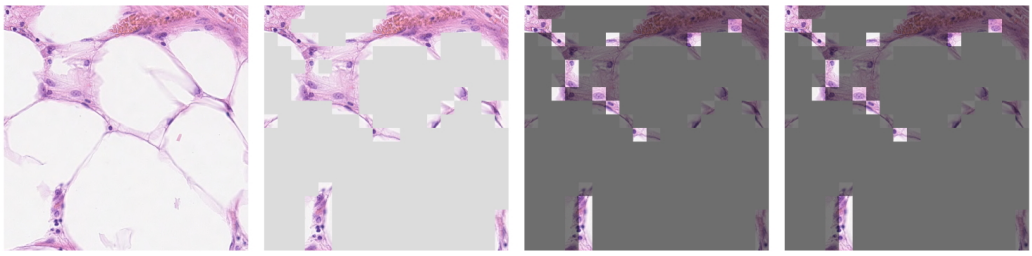

Figure A9: The interpretability output for a negative image in the CRC dataset. In this example, the patches are sparse — there are a lot of background patches that are removed, and the key patches are spread out (as opposed to being connected as in the previous examples). Again, GuidedSHAP is able to correctly identify the patches that support the negative.

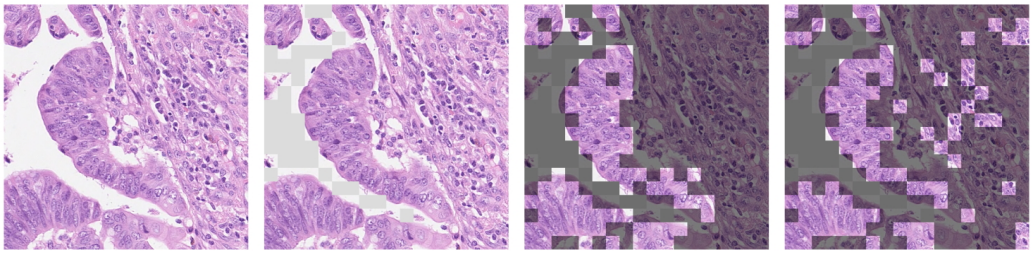

Figure A10: The interpretability output for a positive image in the CRC dataset. In this example, GuidedSHAP has identified some of the ground truth patches, but also additional patches that are not labelled as epithelial in the ground truth labelling. As the labelling was not exhaustive, these patches may contain epithelial nuclei without being labelled as such. By using GuidedSHAP, it is apparent that the model is using these patches in its decision-making, therefore further investigation into the types of nuclei in these patches would reveal more information about how the model makes its decisions; something that would not be possible without the interpretability output.

A.8 ADDITIONAL EXPERIMENTS

In this section, we provide additional experimental outputs for the SIVAL and CRC datasets. We perform the same hyperparameter investigations as we did for the 4-MNIST-Bags dataset in Section 5. Figures A11 and A12 show the effect of changing α for WeightedSHAP, and Figures A13 and A14 show how interpretability performance changes with sample size for the three SHAP-based methods. All the experiments are averaged over 10 repeats, and use the MI-Attn model as the underlying MIL method to interpret.

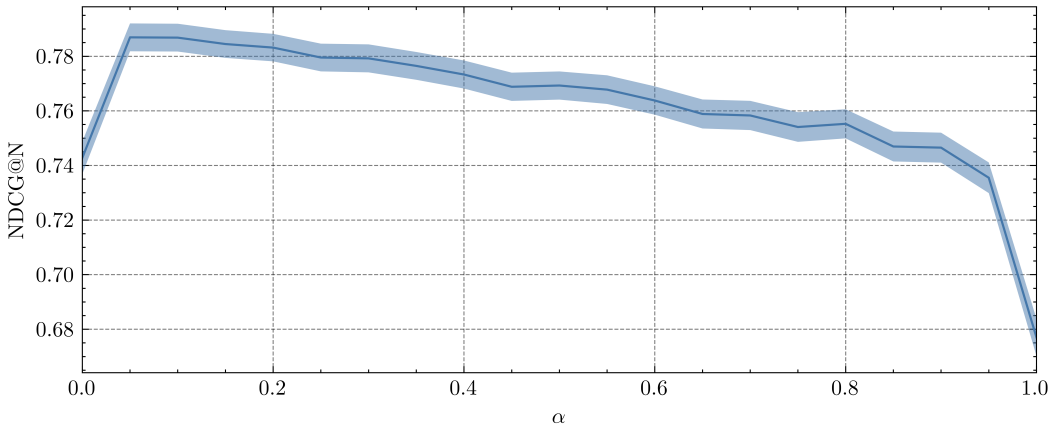

Figure A11: The effect of α on interpretability performance for WeightedSHAP on the SIVAL dataset. Similar to the 4-MNIST-Bags result, we can see that the optimal value for $\alpha < 0.5$ at 0.05. This means the sampling is biased towards non-discriminatory instances.

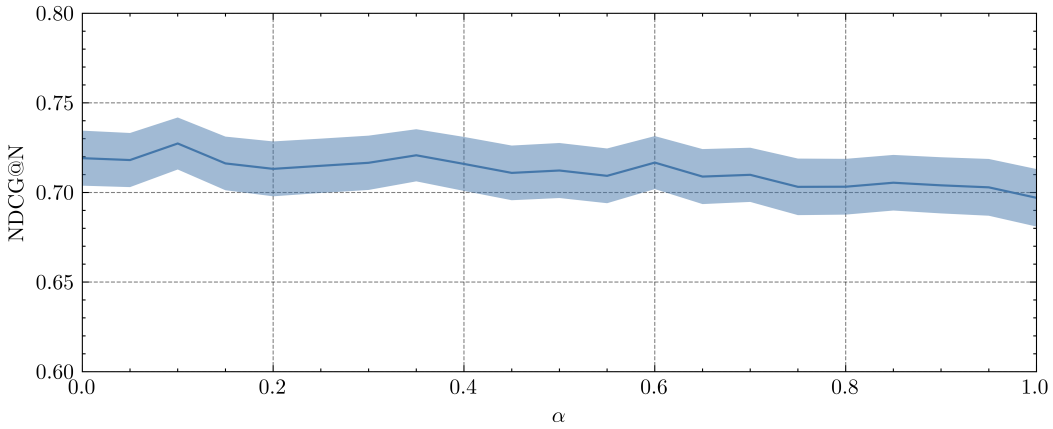

Figure A12: The effect of α on interpretability performance for the CRC dataset. WeightedSHAP performs much worse on this dataset, and as such, the value for α has much less affect than in the SIVAL and 4-MNIST-Bags experiments. However, the best values are still seen for $\alpha < 0.5$.

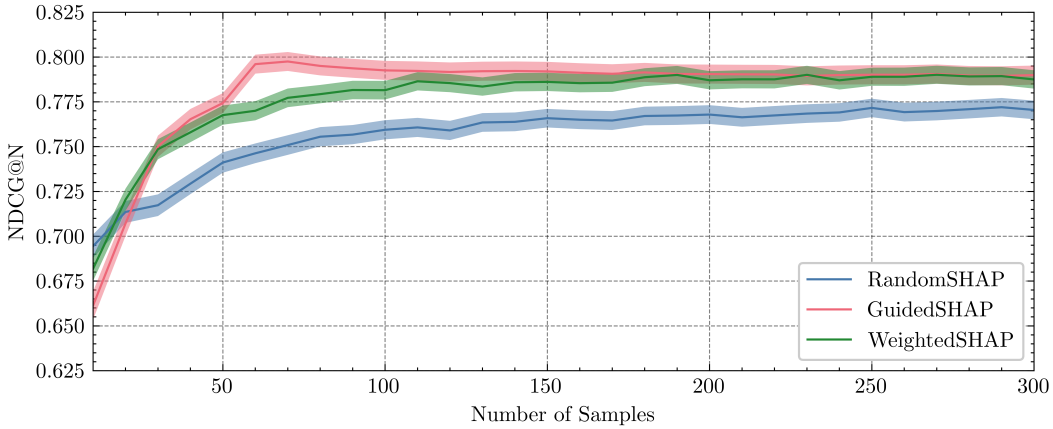

Figure A13: The effect of sample size on interpretability performance for the SHAP-based methods on the SIVAL dataset. The benefit of using WeightedSHAP is not seen here, as the instances are independent, however WeightedSHAP is still more sample efficient than RandomSHAP. The GuidedSHAP method has a peak at around 60 to 70 instances, which is approximately twice the bag size, i.e., at this point both the single instance bag and leave one out bag will have been sampled for all instances. Further work is required in order to understand why further sampling reduces performance from this point, and why this phenomenon does not occur for other datasets.

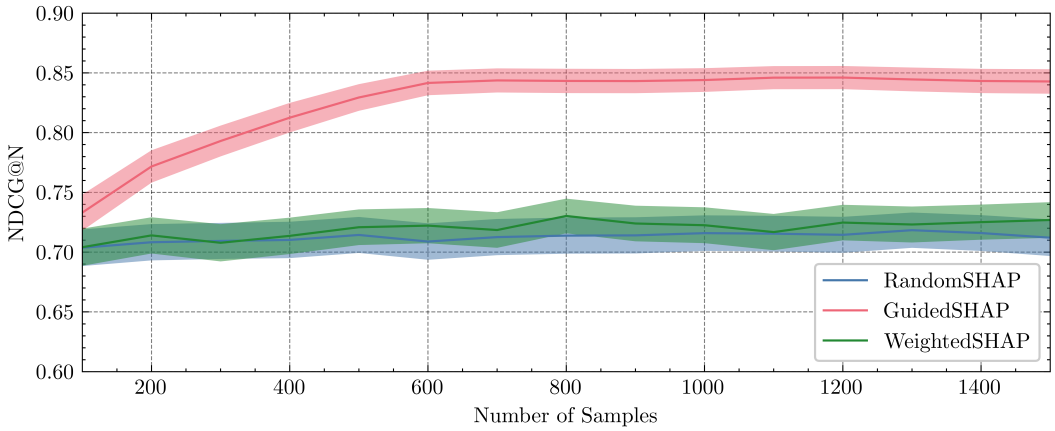

Figure A14: The effect of sample size on interpretability performance for the SHAP-based methods for the CRC dataset. Here, we observe that GuidedSHAP improves with more samples in a similar manner to the SIVAL and 4-MNIST-Bags experiments, but the RandomSHAP and WeightedSHAP methods do not follow the same trends as seen previously. In fact, further samples barely improve their performance at all, which is because both of these methods form large coalitions, so are unable to isolate the effect of each instance. For this dataset, due to the instance independence, the amount of additional information gained by taking an additional large sample is very small compared to the information gained by taking a small sample.