Jagiellonian University

Doctoral School of Exact and Life Sciences
Faculty of Mathematics and Computer Science

Ph.D. Thesis

Interpretable Deep Learning with Prototypical Parts for Supervised and Weakly-Supervised Learning

mgr inż. Dawid Damian Rymarczyk

Under the supervision of
prof. dr hab. Jacek Tabor
dr hab. Bartosz M. Zieliński, prof. UJ

Kraków, 2023

I would like to express my heartfelt gratitude to my supervisors, prof. Jacek Tabor and dr hab. Bartosz M. Zieliński. Throughout the last four years, they have provided me with unwavering support and guidance, for which I am truly grateful. Their openness to discussions, inspiring talks, and willingness to help me overcome obstacles has been invaluable. Working under their supervision has been an immensely gratifying experience from which I grew not only as a researcher but also as a person. I am honored to be their student.

I would also like to take a moment to express my gratitude to my family and friends who have been a constant source of support and encouragement throughout my academic journey. Their strong belief in me and their constant push to aim higher and pursue more ambitious goals has been instrumental in my success.

Without their support, this achievement would not have been possible.

Contents

Streszczenie

W ostatnich latach zaobserwowano dynamiczny rozwój metod sztucznej inteligencji, który spowodował automatyzację wielu zadań. W tych zadaniach systemy oparte o sztuczną inteligencję uzyskują skuteczność na poziomie zbliżonym, badź nawet wyższym niż ludzki. Osiągnięto to dla wielu problemów w domenach takich jak rozpoznawanie obrazów czy przetwarzanie języka naturalnego. Stało się to możliwe głównie dzięki rozwojowi mocy obliczeniowej (w szczególności kart graficznych) i metod opartych o sztuczne sieci neuronowe (SSN). Jednakże modele oparte o SSN mimo wielkiej skuteczności mają też szereg wad. Istotną wadą jest ich czarnoskrzynkowy charakter, który rozumie się jako brak wyjaśnienia zwracanych przez model predykcji i sposobu jego rozumowania. Brak zrozumienia decyzji modelu jest szczególnie niepożądany w dziedzinach takich jak medycyna, gdzie decyzje mają znaczący wpływ na ludzkie życie.

W związku z brakiem transparentności SSN opracowano wiele metod wyjaśniających ich decyzje. Można podzielić je na dwie grupy: metody post-hoc oraz metody samowyjaśnialne. Pierwsza grupa metod zakłada opracowanie dodatkowego modelu analizującego decyzje podejmowane przez model czarnoskrzynkowy. Zaletą tego podejścia jest możliwość zastosowania go do wytrenowanego już modelu bez zmian w jego architekturze. Natomiast częsta wadą jest nieprecyzjność i niewiarygodność zwracanych wyjaśnień. W przypadku metod samowyjaśnialnych, mechanizm interpretowalności jest zaszyty w ich architekturę, więc wraz z predykcją zwracane jest jej wyjaśnienie. Dzięki temu zapewniają one większą wiarygodność wyjaśnienia, ale są jednocześnie trudniejsze do wytrenowania, a osiągana przez nie skuteczność jest niższa niż ta uzyskiwana przez metody czarnoskrzynkowe.

Niniejszy doktorat skupia się na metodach samowyjaśnialnych opartych na sztucznych sieciach neuronowych, ze szczególnym uwzględnieniem mechanizmu poolingu atencyjnego oraz części prototypowych. Mechanizm poolingu atencyjnego stosuje się do uczenia ze słabym nadzorem, zwłaszcza do uczenia wieloinstancyjnego (gdzie jedna etykieta przypisana jest do zbioru instancji). Natomiast modele oparte o części prototypowe w trakcie treningu uczą się konceptów wizualnych pochodzących z danych treningowych. Na etapie inferencji zapamiętane przez model koncepty wizualne są porównywane do danych wejściowych w celu dokonania ostatecznej predykcji.

W ramach doktoratu, bazując na modelu opartym o części prototypowe ProtoPNet [11], opracowano trzy nowe architektury, które niwelują ograniczenia modelu bazowego. Pierwszy z nich, ProtoPShare [I], współdzieli części prototypowe pomiędzy klasami poprzez ich łączenie w uprzednio wytrenowanym modelu. Łączenie to odbywa się w oparciu o nową metrykę potrafiącą wykryć podobieństwo semantyczne pomiędzy prototypami, nawet gdy są one odległe w przestrzeni ukrytej. Aby wykluczyć potrzebę trenowania modelu bazowego, wprowadzono ProtoPool [II], który uczy się zbioru części prototypowych wraz z ich przypisaniem do poszczególnych klas, pozwalając na ich współdzielenie. Jest to możliwe dzięki zastosowaniu technik regularyzacyjnych opartych o Gumbel-Softmax oraz wprowadzeniu podobieństwa focal similarity, które wykrywa bardziej charakterystyczne części prototypowe. Uogólnienie tych metod do problemu regresji w przewidywaniu właściwości molekuł zaprezentowano wraz z dedykowanym modelem ProGReST w pracy [III].

Poza rozwojem metod opartych na częściach prototypowych, w ramach doktoratu rozwinięto metodykę poolingu atencyjnego stosowanego w problemach uczenia wieloinstancyjnego i zaproponowano dwa nowe podejścia. Pierwsze z nich, SA-AbMILP [IV] korzysta z mechanizmu self-attention do nauki zależności pomiędzy instancjami w zbiorze. Dzięki temu lepiej sprawdza się dla bardziej złożonych założeń uczenia wieloinstancyjnego, a jednocześnie wyjaśnia jak dana cecha wizualna wpłynęła na decyzję modelu. Drugie z nich, ProtoMIL [V], pozwala na lokalną i globalną interpretowalność dla problemu uczenia wieloinstancyjnego poprzez połączenie ze sobą części prototypowych i poolingu atencyjnego.

Podsumowując, prezentowany doktorat skupia się na metodach interpretowalności dla uczenia głębokiego, a w szczególności na modelach opartych o części prototypowe i mechanizm atencji. W ramach przeprowadzonych badań opublikowanych zostało pięć prac na konferencjach naukowowych posiadających kategorię A* [I, II] i kategorię A [III, IV, V] według rankingu CORE. Doktorant jest pierwszym autorem wszystkich tych publikacji. Ponadto był on kierownikiem grantu NCN Preludium oraz grantu w ramach Inicjatywy Doskonałości Uniwersytetu Jagiellońsiego. Doktorant odbył również staż naukowy w Centrum Wizji Komputerowej Autonomicznego Uniwersytetu Barcelońskiego w grupie badawczej prof. Joosta van de Weijera, oraz jest współautorem aplikacji patentowej złożonej do Europejskiego Biura Patentowego.

Słowa kluczowe: interpretowalność, wyjaśnialna sztuczna inteligencja, uczenie głębokie.

Abstract

Recent years have brought a significant advancement in artificial intelligence methods allowing for the automation of multiple repetitive tasks in many domains, including image recognition and natural language processing. This resulted in performance comparable or superior to humans. These achievements were largely possible owing to recent breakthroughs in computational devices, especially graphical processing units (GPUs), and machine learning techniques utilizing Deep Neural Networks (DNNs). However, despite their remarkable performance, DNNs involve multiple flaws. One major drawback is their black-box nature caused by the lack of explanations for their decisions. The inability to inspect the model's reasoning process is a concerning aspect, particularly in high-stakes decision fields such as medicine, where those decisions may significantly impact human lives.

In response to the lack of transparency of DNNs, various approaches have been developed to provide explanations for their decisions. These approaches may be divided into two categories: post-hoc and self-explainable methods. Post-hoc methods involve training a black-box model and subsequently developing an explainer model on top of it. This approach can be applied to already-trained models, but resulted explanations are frequently unreliable and imprecise. Nonetheless, self-explainable approaches incorporate interpretability as an intrinsic aspect of the model. This way they provide explanations alongside with their predictions. While self-explainable models offer higher quality explanations compared to explainers, they are more challenging to train, and their performance is moderately lower than the black-box models.

This doctoral thesis focuses on developing self-explanaible methods utilizing artificial neural networks, with particular focus on attention pooling mechanism and prototypical-parts methodology. The former is employed in weakly supervised learning, particularly in multi-instance learning, where a set of instances is assigned to a single label. The latter, prototypical-part models learn concepts from the training data during the learning process and compare those concepts to an input sample in the inference phase to obtain the prediction.

As part of this doctoral thesis, three new models were developed to address the limitations of the initial prototypical parts-based model ProtoPNet [11]. The first model, ProtoPShare [I] introduces prototype parts sharing across classes by combining those already trained. For this purpose a new metric was defined detecting semantic similarity between prototypical parts, even when they are far in the latent space. The next model, ProtoPool [II], is more advanced and shares the prototypical parts from scratch. It is possible thanks to regularization techniques based on the Gumbel-Softmax Trick. Moreover, ProtoPool introduces focal similarity, which enables more descriptive prototypical parts than those obtained from baseline methods. Finally, ProGReST [III], generalizes prototypical parts to regression problems of predicting molecular properties.

In addition, two methods have been introduced based on the attention pooling methodology. The first one, SA-AbMILP model [IV] uses a self-attention mechanism to learn dependencies between instances in a bag resulting in better performance in non-standard multiple instance learning assumptions, such as presence-based and threshold-based. This enables to understand which visual features influence the model's decision. The second model, ProtoMIL [V], aims to introduce global level explanation into the MIL classification problem by combining prototypical parts and attention pooling.

To summarize, this doctoral thesis focuses on interpretable deep learning based on prototypical parts-based models and attention mechanism. Five papers were published based on those research, two at A* conferences [I, II], and three at A conferences [III, IV, V] according to CORE ranking (in all of them the Ph.D. candidate is the first author). Additionally, the Ph.D. candidate was a principal investigator of the NCN Preludium grant and the grant from the Jagiellonian University Excellence Initiative. He also completed a research internship at the Computer Vision Center of the Autonomous University of Barcelona in the research group of Professor Joost van de Weijer and is a co-author of a patent application submitted to the European Patent Office.

Keywords: interpretability, explainable artificial intelligence, deep learning.

1. Introduction and Motivation

The advancements in artificial intelligence (AI), particularly in deep learning (DL), have led to an increased use of deep neural network models in various fields such as image recognition [52, 54, 69], natural language processing [14, 77], and drug discovery [12, 24]. In many cases, these computer systems perform on par or even surpass human-level performance [27]. It is mainly possible thanks to the availability of computational resources, software frameworks, and vast digital datasets. Moreover, the widespread adoption of DL models is possible thanks to their properties allowing them to model complex non-linearities and unstructured data representation [27]. For instance, raw pixels of images can be transformed into meaningful latent codes representing the picture.

However, the growing use of DL models in fields such as medicine [47] and justice [70], where high-stakes decisions are made, requires a better understanding of the reasoning process behind the predictions made by these systems [57]. Unfortunately, most DL models have a black-box nature [57], meaning that they do not provide explanations for their predictions, see Figure 1.1. Therefore, there is a need to develop approaches to explain decisions of deep neural networks [57]. This need was acknowledged by regulatory bodies such as the European Union, and as a result, laws have been introduced requiring decision support systems to provide explanations to users [34]. The need to understand model's decision, further strengthened by the regulatory bodies, has given rise to the field of eXplainable Artificial Intelligence (XAI).

XAI can be divided into two sub categories: post-hoc methods and self-explainable models [57]. Post-hoc methods aim to explain pre-trained neural networks using a second model, called an explainer. The explainer is built on top of the black-box model, so there is no need to alter the neural network architecture or retrain the models [57]. Popular post-hoc methods include saliency maps [55, 62, 63, 65], perturbation-based explanations [19, 20, 56], concept activation vectors [13,26,37,76], and importance-based methods [5,49]. However, their explanations may be unreliable due to the biases from the black-box model and explainer [1,57].

To address these limitations, self-explainable (gray-box) approaches have emerged. These models have a built-in interpretability component making their explanations more reliable. However, these models are more difficult to train and may have to a certain extent lower performance than black-box approaches [58]. One of the recent attempts to create a gray-box neural network was the self-explainable neural network [3] consisting of a second computational path for explanations. Subsequent approaches utilize bag-of-words [10] or

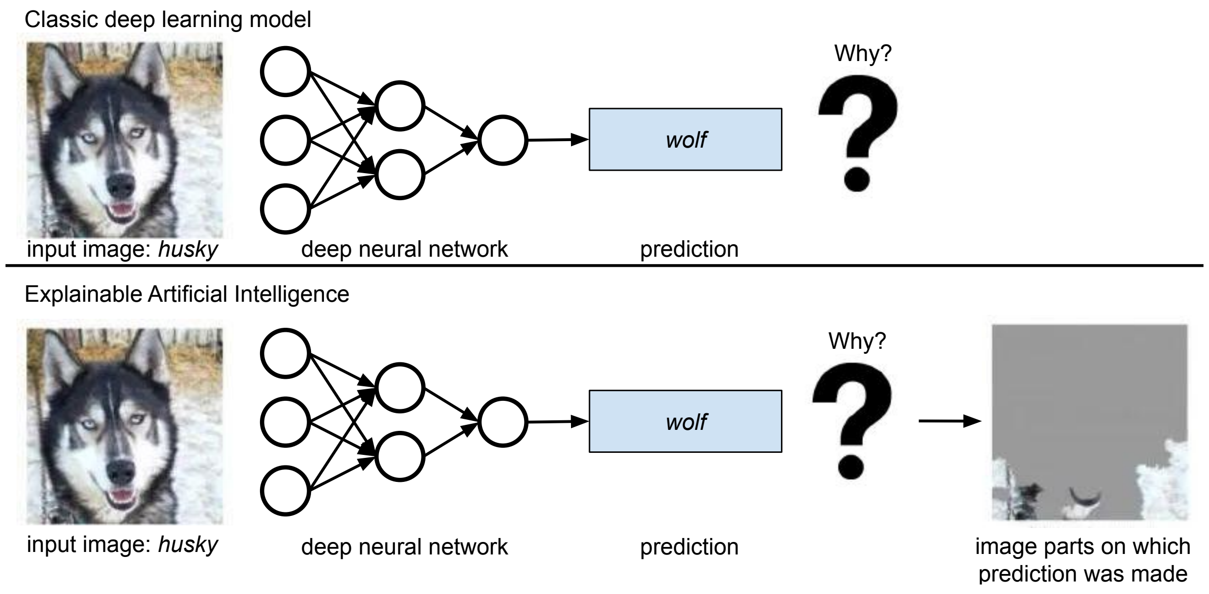


Figure 1.1: In contrast to traditional deep learning methods (top part), the explainable artificial intelligence (XAI) pipeline (bottom part) enables users to obtain explanations for the model's predictions and assess the model's decision-making process. In this example, an image of a husky is misclassified as a wolf as the model learns to associate snow with wolves (a real-life example copied from [56]). It highlights the importance of XAI approaches in understanding and identifying potential biases and limitations of deep learning models.

attention mechanisms [23, 67, 81]. However, all of the above focus only on explaining a single prediction (local explanations) and return no explanation for the classes (no global explanations).

To tackle this issue and enable global explanations, concept-based models have been introduced. They can be divided into two subgroups: Concept Bottleneck Models (CBM) [39, 50] and prototypical parts-based models [11, 16, 28, 46, 53, 60, 61, 71] with a baseline method called Prototypical Parts Network (ProtoPNet) [11]. CBM learns to predict a set of prede[U+FB01]ned concepts and makes predictions based on their presence in the input, whereas ProtoPNet compares the parts of the input data to patterns represented by parts of the training data to measure the similarity between them and uses them to make a prediction (an example of interpretation from ProtoPNet is presented in Figure 1.2). Prototypical parts-based models provide local and global explanations, characterizing data classes, and making them useful for a wide range of applications.

Prototypical parts-based models are widely recognized as an interpretable approach and and from the user's perspective are considered to be the most effective in comprehending the explanations [33], due to the use of parts of the training examples to perform the prediction. For the aforementioned reasons, we decided to follow this direction of research and **aimed to show that it is possible to build more transparent and trustworthy machine learning-based systems based on prototypical parts and attention mechanism.**

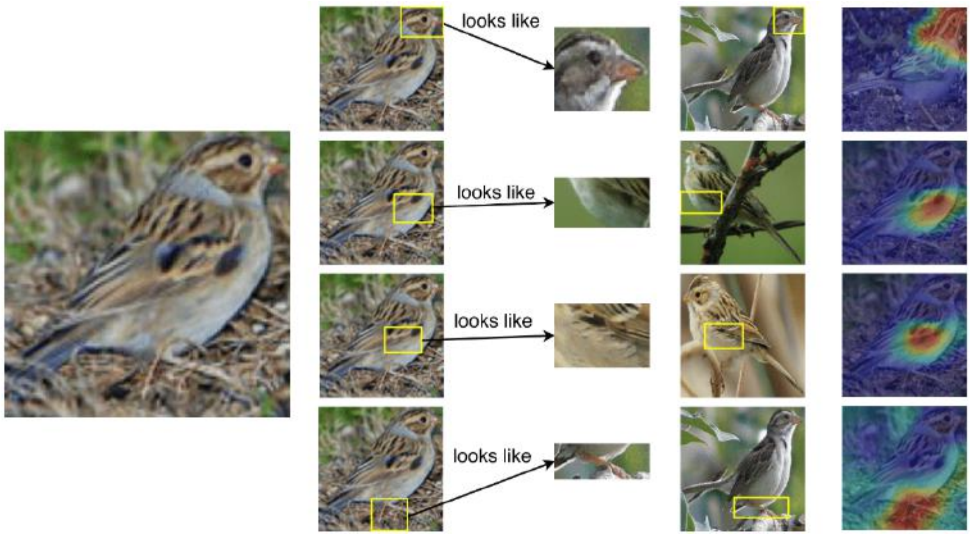


Figure 1.2: ProtoPNet finds similarities between the input image and reference patterns, and presents the explanation in the form of "This looks like that...". As exemplified, "this looks like sparrow, because its wing, head and legs are similar to the wing, head and legs of other sparrows" (copied from: [11]).

To prove this hypothesis broad research has been conducted resulting in five scientific publications that constitute part of this doctoral dissertation:

[I] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, Bartosz Zieliński. “Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification.“ ACM Conference on Knowledge Discovery & Data Mining (KDD), 2021, 1420-1430, CORE A*, 200 MEiN points.

[II] Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, Bartosz Zieliński. “Interpretable image classification with differentiable prototypes assignment.“ European Conference on Computer Vision (ECCV), 2022, 351-368, CORE A*, 140 MEiN points.

[III] Dawid Rymarczyk, Daniel Dobrowolski, Tomasz Danel. “ProGReST: Prototypical Graph Regression Soft Trees for Molecular Property Prediction.“ SIAM Conference on Data Mining (SDM), 2023, 379-387, CORE A, 140 MEiN points.

[IV] Dawid Rymarczyk, Adriana Borowa, Jacek Tabor, Bartosz Zieliński. “Kernel self-attention for weakly-supervised image classification using deep multiple instance learning.“ IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2021, 1721-1730, CORE A,140 MEiN points.

[V] Dawid Rymarczyk, Adam Pardyl, Jarosław Kraus, Aneta Kaczyńska, Marek Skomorowski, Bartosz Zieliński. “ProtoMIL: Multiple Instance Learning with Prototypical Parts for Whole-Slide Image Classification.“ European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), 2022, CORE A, 140 MEiN points.

Contributions of these research articles are as follows:

- Introducting two prototypical parts-based models, ProtoPShare [I] and ProtoPool [II], detecting similarities between data classes and increase interpretability with prototypical parts sharing when compared to ProtoPNet [11].
- Generalizing prototypical parts and soft neural decision trees to the regression problem and applying it to molecular property prediction in the ProGReST model [III].
- Designing two interpretable models for Multiple Instance Learning (MIL) problem, SA-AbMILP [IV] and ProtoMIL [V], with the former modelling the relationships between instances within a bag, and the latter generalizing prototypical parts to MIL.

The structure of the doctoral dissertation is as follows: In Section 2, the research scope of this dissertation is briefly discussed. In Section 3, a detailed description is provided, consisting of two subsections: Section 3.1 is dedicated to the contributions related to prototypical part-based models, and Section 3.2 is focused on the advancements made in MIL problems. The research profile of the Ph.D. candidate is presented in Section 4. Finally, in Section 5, the contributions are outlined.

2. Research scope

Explainable Artificial Intelligence (XAI) is a broad field integrating explainability (post-hoc) and interpretability (self-interpretable) approaches [4]. Despite the fact they are frequently used interchangeably, there is a critical difference between them. Explainability aims to provide explanations for already trained black-box models. To achieve this, an explainer is needed to provide an explanation for a prediction, as it is not part of the prediction system. Interpretability, however, focuses on building inherently interpretable models providing an explanation along with a prediction [57].

Thanks to the advancements made in the field of XAI, a systematic categorization of its methods and properties has been proposed in [4], furthering the current state of research. The taxonomy presented in Figure 2.1 indicates that current research mainly focuses on static explanations without user involvement, and predominantly for models as opposed to data exploration. For model explanations, the methods can be local (explaining a single prediction) or global (characterizing a data class), post-hoc (explainers), or direct (e.g. self-explainable like prototypical parts [11]). For post-hoc methods, examples or features can be used as explanations, such as ExMatchina [33] or GradCAM [62]. The taxonomy also distinguishes surrogate models, i.e. simple methods (such as decision rules) distilled from black-box models. All of the works listed as this dissertation contribution pertain to into static, model-oriented and direct (self-explanatory) approaches.

The first part of this dissertation focuses on the self-explainable models with both, global and local explanations:

- **Sharing prototypical parts with two-staged training (ProtoPShare [I])**. This work introduces an extension to the Protoypical Parts Network (ProtoPNet) [11] allowing for sharing prototypical parts between classes, addressing a shortcoming in the original model. While ProtoPNet is inspired by human communication, it does not account for the fact that real-world objects share common parts. To overcome this limitation, an extension was created, introducing a data-dependent operation consisting in discovering semantically similar prototypical parts and merges them.

  This resulted in a model capable of achieving comparable accuracy to ProtoPNet while allowing for the discovery of similarities between data classes. Extensive numerical experiments demonstrate that the number of prototypical parts that can be merged varies depending on the backbone network, and merging an excessive number of parts can

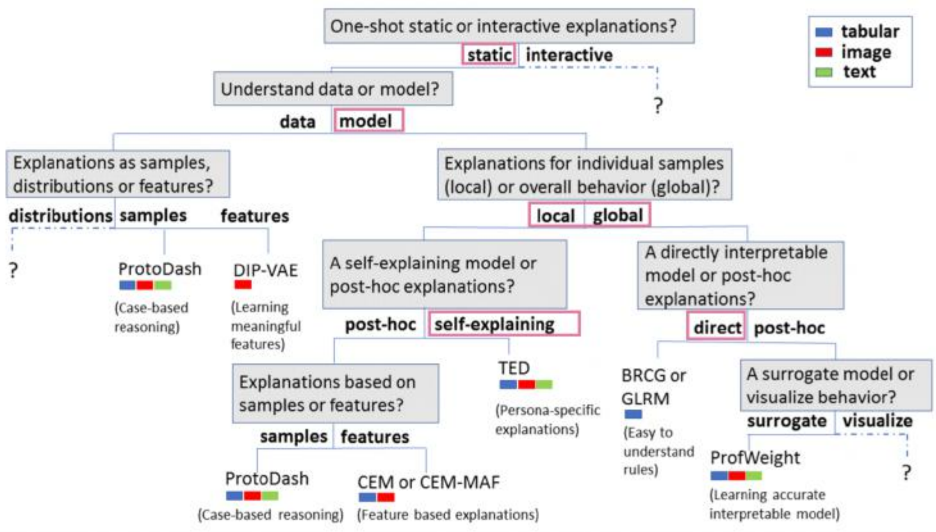

One-shot static or interactive explanations?
static
interactive
tabular
image
text
Understand data or model?
data
model
?
Explanations as samples, distributions or features?
distributions
samples
features
Explanations for individual samples (local) or overall behavior (global)?
local
global
?
ProtoDash
(Case-based reasoning)
DIP-VAE
(Learning meaningful features)
A self-explaining model or post-hoc explanations?
post-hoc
self-explaining
A directly interpretable model or post-hoc explanations?
direct
post-hoc
Explanations based on samples or features?
samples
features
TED
(Persona-specific explanations)
BRCG or GLRM
(Easy to understand rules)
A surrogate model or visualize behavior?
surrogate
visualize
ProtoDash
(Case-based reasoning)
CEM or CEM-MAF
(Feature based explanations)
ProfWeight
(Learning accurate interpretable model)
?

Figure 2.1: Taxonomy of Explainable Artificial Intelligence based on data type, interpretability level and exemplary approaches outlining current state of the XAI field and where this thesis contributes most (in pink boxes) (copied from [4].)

contribute to model collapse. However, reducing the number of prototypes by a factor of five without significant drop in accuracy is achievable (smaller number of prototypical parts increases the models interpretability). A user study shows that our method is more effective in detecting similar semantic concepts compared to other non data-dependent operations. Furthermore, theoretical analysis, stated theorem and its proof demonstrate why prototypical parts merge-pruning is not resulting in significant accuracy drop before model collapse.

In summary, we introduced a novel prototypical parts-based architecture improving the transparency of deep learning models and enabling the detection of inter-class similarities. The reduction in prototypes during the prediction phase makes the model more sparse, which is a crucial factor for trustworthy explainable models [58].

- **Prototypical parts sharing from scratch (ProtoPool) [II]**. In this work, we introduce ProtoPool, which is a natural extension of ProtoPShare [I]. While ProtoPShare requires a two-stage training process to achieve prototypical parts sharing, ProtoPool provides a more optimal solution by enabling the training of shared prototypical parts from scratch. This is made possible through the use of the Gumbel-Softmax trick, which associates prototypical parts from a pool with classes. To ensure that multiple prototypical parts are assigned to a single class, we employ regularization with an orthogonal loss. Additionally, we propose a novel focal similarity function increasing the expressiveness of prototypical parts by focusing on an object's salient features.

  ProtoPool achieves state-of-the-art accuracy, while allowing for the discovery of similarities between data classes and operating on fewer prototypical parts. Also neuroscience-inspired

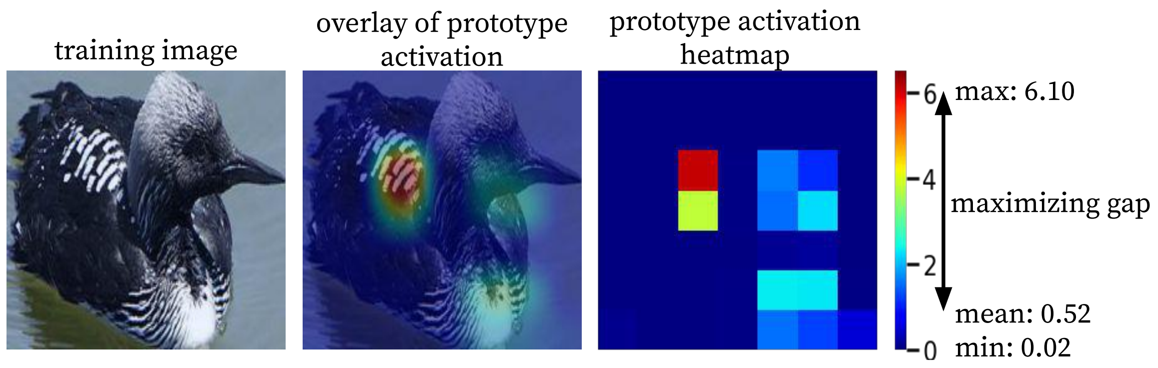


Figure 2.2: Our focal similarity limits high prototype activation to a narrow area. It is obtained by widening the gap between the maximal and average activation (equal 6.10 and 0.52, respectively). ProtoPool with focal similarity generates more salient prototypes than other models. Image copied from our work [II]

intuition for our architectural design is provide, e.g. ProtoPool mimics the simultaneous information processing which is faster than successive processing [35] present for ProtoTree [53]. Furthermore, our user study demonstrates that ProtoPool with focal similarity produces prototypical parts more salient than those generated by baseline methods, and we demonstrate that increasing the number of prototypical parts does not always result in improved model performance, as the model can saturate.

In summary, ProtoPool addresses the shortcomings of ProtoPShare and enables the training of shareable prototypical parts between classes from scratch. This increases the efficiency and accuracy of the model, as well as results in improved transparency with more sparse prototypical part explanations than ProtoPShare. The novel focal similarity function detects more robust and salient prototypical parts, increasing their expressiveness and resulting in higher trustworthiness of the model, as confirmed by our user study.

- **Generalization of prototypical parts approach to the graph regression problem (ProGReST) [III].** ProGReST is a novel solution to the problem of graph regression for molecular property prediction, with an emphasis on interpretability. The model, which is a soft neural tree [22], represents each node as a prototypical part (in this case in the form of a subgraph). During prediction, similarities between the input graph and subgraphs derived from the training dataset are used to provide explanations. We introduce several regularization techniques to effectively train the nodes with meaningful prototypical parts. Moreover, a proxy-projection approach is defined to significantly reduce the training time of the model.

  In our experiments on five chemical datasets, ProGReST achieves state-of-the-art performance compared to other deep learning methods. We also demonstrate that strong regularization of the tree is crucial to obtain the best performing model with meaningful prototypical parts. The proxy-projection approach significantly reduces the push operation

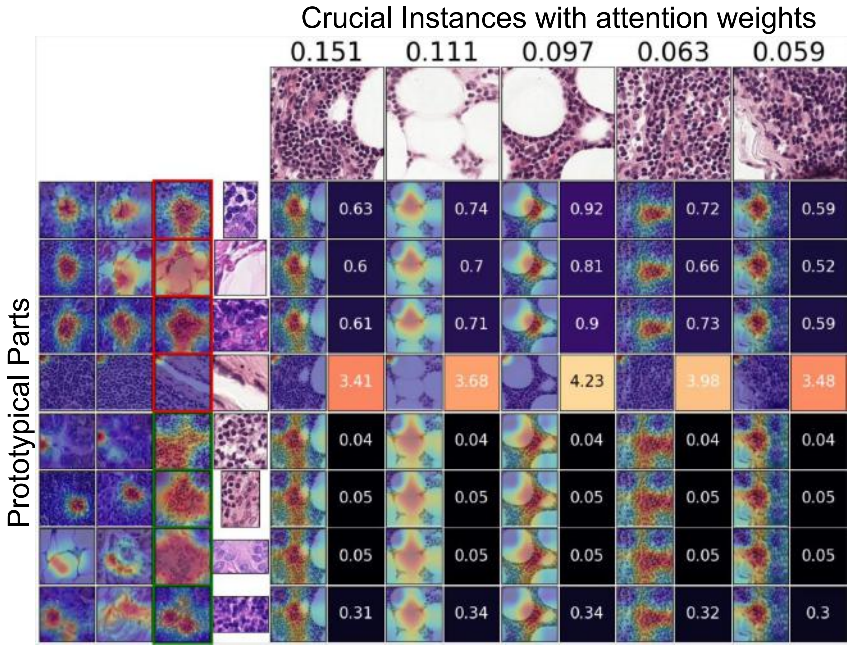


Figure 2.3: Similarity scores between crucial instances of a bag and prototypical parts for a negative bag from the Camelyon16 dataset. There are five instances (columns) and eight prototypical parts (rows). Each prototypical part is represented by the part of training image and three nearest training patches, while each instance is represented by the corresponding patch and its attention weight (a_i). The heatmap in each cell indicates the prototype activation. ProtoMIL strongly activates only one prototype and focuses mainly on nuclei when analyzing the healthy parts of the tissue. These findings provide valuable insights into how ProtoMIL operates and how it is able to effectively classify bags into positive and negative classes. Image copied from our work [V].

time by an order of magnitude, while chemical experts confirm the faithfulness of the explanations provided by the model.

In summary, ProGReST is a novel architecture addressing the problem of graph regression for molecular property prediction in an interpretable manner. Our results are superior to other methods, and the utilized regularizers are crucial to obtain the best performing models. By incorporating prototypical parts into this task, we increased the transparency of DL tools for drug design, and the resulting explanations can guide medicinal chemists in drug candidate design by demonstrating how specific substructures of the compounds influence the final chemical property prediction.

In the second part of this dissertation, we focus on the problem of Multiple Instance Learning (MIL) and propose contributions to make MIL models more interpretable:

- **Interpretable aggregation with self-attention (SA-AbMILP) [IV].** To enhance the interpretability of models suited for Multiple Instance Learning (MIL) problems, the we introduce a new approach called Self-Attention Attention-based MIL Pooling operator (SA-AbMILP). Unlike the basic AbMILP approach [31], SA-AbMILP can handle more challenging MIL assumptions, such as presence-based and threshold-based assumptions, while maintaining interpretability. The model includes a self-attention module before the pooling operator enabling the model to identify the relationship between instances in a bag. This allows the model to consider not only individual instances' contribution to the prediction, but also how they interact with each other. In addition, the authors suggest using different attention kernels as a hyperparameter of the model.

To evaluate the effectiveness of the proposed approach, extensive numerical experiments were conducted on five datasets. The obtained results demonstrated that SA-AbMILP outperforms other state-of-the-art MIL models. Moreover, SA-AbMILP and its kernel versions can identify key instances within bags accurately. It was shown through a detailed analysis of the model's explanations based on the artificial dataset simulating the threshold and presence-based MIL assumptions. Furthermore, self-attention maps can be used as parts of the explanations to show how instances interfere with each other.

Overall, SA-AbMILP is an approach improving the interpretability of models suited for MIL. Its ability to model interactions between instances within a bag makes it suitable for more challenging MIL assumptions, such as presence-based and threshold-based. The numerical experiments demonstrated the superior performance of the model, while the analysis of the explanations showed that SA-AbMIL improves transparency over baselines such as AbMILP.

- **Local and global interpertability for MIL (ProtoMIL) [V]**. While previously discussed models, such as AbMILP and SA-AbMILP, provide interpretability only at the local prediction level, we introduced ProtoMIL a novel approach to incorporate global interpretability into Multiple Instance Learning (see Figure 2.3). It is possible through a combination of prototypical parts with attention pooling operator. Moreover, to effectively train ProtoMIL, we propose novel cluster and separation loss functions. They use results from attention mechanism to identify key instances to weight the influence of each instance within a bag on a learning of prototypical parts.

  As a result, ProtoMIL provides the global and local interpretation of the prediction in contrast to baseline models with only local explanations. The global interpretation of the model, in the form of class-specific prototypical parts and key bag instances, is shown through a novel matrix-based visualization. At the same time the performance of the ProtoMIL, measured with accuracy and ROC AUC, is comparable to state-of-the-art methods.

  To conclude, ProtoMIL affords a better understanding of the model's behavior at both local and global levels through the combination of prototypical parts and attention mechanism. While ProtoMIL provides interpretability, it still achieves accuracy comparable to the state-of-the-art methods.

All the previously mentioned works propose novel deep learning architectures based on prototypical parts and attention mechanisms. This unique combination has made them highly interpretable, resulting in increased transparency and trustworthiness of self-explainable approaches. These claims were confirmed by a range of experiments, including user studies, analysis performed by domain-experts, and experiments on multiple datasets.

3. Detailed description of the contributions

3.1. Prototypical parts-based models: ProtoPShare [I], ProtoPool [II] and ProGReST [III]

To make the thesis self-contained, this chapter starts with the description of the ProtoPNet [11] architecture and its training methodology. It is necessary since the contributions of this thesis part are based on prototypical part layer introduced by ProtoPNet [11]. The subsequent sections discuss the limitations of ProtoPNet and how they were addressed by ProtoPShare [I] (Section 3.1.2) and ProtoPool [II] (Section 3.1.3), respectively. The final section (Section 3.1.4) of this chapter introduces the ProGReST [III] model, which is a novel approach to graph regression problems utilizing prototypical parts.

3.1.1. Preliminaries

Motivation. The primary objective of ProtoPNet [11] is to achieve interpretability in image classification. The authors drew inspiration from how humans explain the identity of one object to one another. They base their explanation on the sentence *This looks like that...*. For instance, this is a bird because it has wings, tail, and feathers, as presented n Figure 1.2. This approach is called positive reasoning since it relies only on the features that a given object possesses to explain its identity.

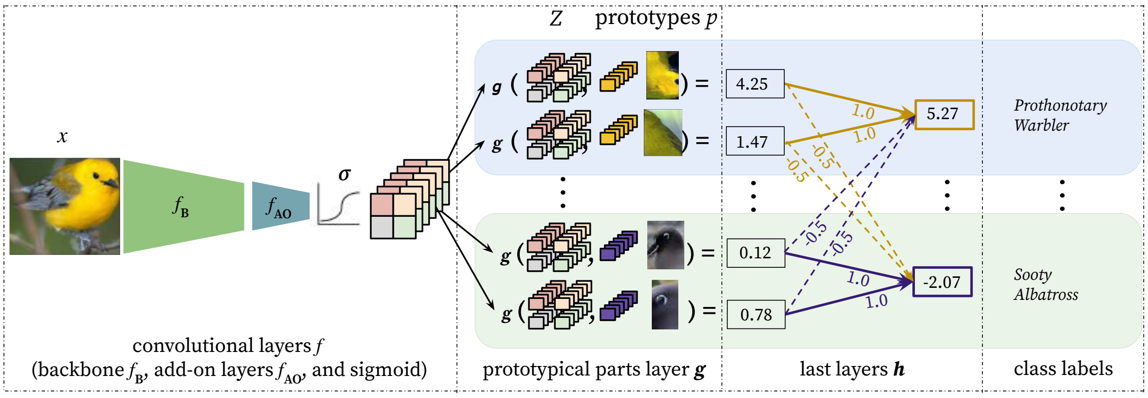


Figure 3.1: Architecture of ProtoPNet.

Architecture. The ProtoPNet model comprises three main parts: convolutional layers f are composed of a pretrained convolutional backbone f_B, such as ResNet, and two additional 1×1

convolutional layers with sigmoid activation f_{AO}, followed by the prototypical parts layer g and the fully connected layer h. The purpose of f_{AO} is to translate the convolutional output to a prototypical part space. The prototypical part layer g contains K prototypes $p_i \in \mathbb{R}^D$ per class which are compared to the latent representations of the input. Finally, the last layer h links the output of the prototypical parts layer, with a specific weight initialization scheme: if prototype p_i is assigned to class c, then h_{ci} is set to 1, otherwise, it is set to -0.5. Figure 3.1 depicts the architecture of ProtoPNet.

Inference. For an input image $x \in \mathcal{X}$, the hidden representation $f_B(x)$ of shape $H \times W \times d$ is generated by the backbone model f_B. The output is then translated into a prototypical parts space of shape $H \times W \times D$ by f_{AO} and result in a set of representation vectors $Z_x = \{z_i \in f(x) : z_i \in \mathbb{R}^D, i = 1, ..., H \cdot W\}$. Each prototypical part p_i of shape $1 \times 1 \times D$ is compared to each of the $H \times W$ representation vectors to calculate their similarities (i.e. the activations of the prototype on the analyzed image). The maximum value from the similarities is then taken to assess the presence of an i-th prototype on the image. The similarity calculation is based on the equation:

$$\max_{z_j \in Z_x} sim(p_i, z_j), \text{ where } sim(p_i, z_j) = \log \frac{|z_j - p_i|_2 + 1}{|z_j - p_i|_2 + \eta} \text{ and } \eta \ll 1. \tag{3.1}$$

Finally, to obtain the predictions, these values are passed through the fully connected layer h.

Training. The training process of ProtoPNet consists of three phases: warm-up, joint learning, and convex optimization of the last layer. In the first phase, the convolutional add-on layers f_{AO} and the prototypical parts layer g are learned. During the second phase, f_{AO} and g are trained jointly with the backbone network pretrained f_B. Finally, in the last phase, the fully-connected layer h is fine-tuned.

The first two phases are trained using the cross-entropy loss function, along with two additional regularization terms: cluster and separation costs [11]. The cluster loss ensures that each training image has a patch representation close to at least one prototypical part of its class. In contrast, the separation cost pushes every patch representation of a training image away from the prototypical parts of the other classes. These costs are defined as follows:

$$L_{clst} = \sum_{x \in \mathcal{X}} \min_{p_j \in P_c} \min_{z \in Z_x} |z - p_j|_2^2; \quad L_{sep} = -\sum_{x \in \mathcal{X}} \min_{p_j \notin P_c} \min_{z \in Z_x} |z - p_j|_2^2.$$

In the convex optimization phase of the last layer h, in addition to the cross-entropy loss, ProtoPNet regularizes the h weights initialized with negative values to be close to 0. This regularization term of the last layer (ll) for each class $c \in C$ is as follows: $L_{ll} = \sum_{c=1}^{C} \sum_{p_j \notin P_c} |h_{c,j}|$.

Finally, to ensure that the learned prototypical parts can be represented as a part of the images from the training dataset, a push operation is performed. For each prototype p_j assigned to the class c, the closest patch representation from all images of the training set of that class is assigned. This is computed as follows:

$$p_j \leftarrow \arg\min ||z - p_j||_2, \text{ where } Z = \{z : z \in f(x_i) \forall i \text{ such that } y_i = c\} \tag{3.2}$$

3.1.2. Discovering class similarities with prototypical parts sharing (ProtoPShare) [I]

This section starts with the ProtoPShare's motivation. Then key contributions are presented together with the main results. Additional details on the method and results are provided in the work [I].

Motivation. The ProtoPNet discussed earlier has several limitations problematic in applications. These limitations arise from two main reasons. Firstly, each prototype is assigned to only one class, leading to a large number of prototypes. This, in turn, negatively affects interpretability, which requires small number of features in explanations that are of low complexity [17, 73]. Secondly, the training process pushes the prototypes of different classes away from each other in the representation space. As a result, prototypes with similar semantics can be far apart from each other, leading to unstable predictions (see Figure 3.2).

Contributions. To address these limitations, we introduced the *Prototypical Part Shared* network (*ProtoPShare*), which shares prototypes between classes via data-dependent merge-pruning. This approach leads to a relatively small number of prototypes and promotes proximity between prototypes with similar semantics. Moreover, this method enables the discovery of similarities between classes.

Details. To enable sharing of prototypical parts across multiple data classes, our pruning process involves several steps. First, we begin with a set P of all prototypes obtained after ProtoPNet training, K number of classes, and ζ percentage of prototypes to merge per pruning step. Next, we compute similarities between prototype pairs and merge ζ percent of the most similar pairs $(p, \tilde{p})$. We discard prototype p and its weights $h(p)$ and instead use prototype $\tilde{p}$ and its weights $h(\tilde{p})$, aggregated with $h(p)$ via summation, for the class to which p was assigned. After each step, we fine-tune layer h. This merge-pruning process enables sharing of prototypical parts across multiple data classes, as shown in Figure 3.3. The theoretical results of merge-pruning operation are provided in the main paper [I].

Training with the loss function described in [11] can produce prototypes of similar semantic that are distant in the representation space because the Euclidean distance between prototypical parts does not capture features semantics. To address this issue, we propose a data-dependent

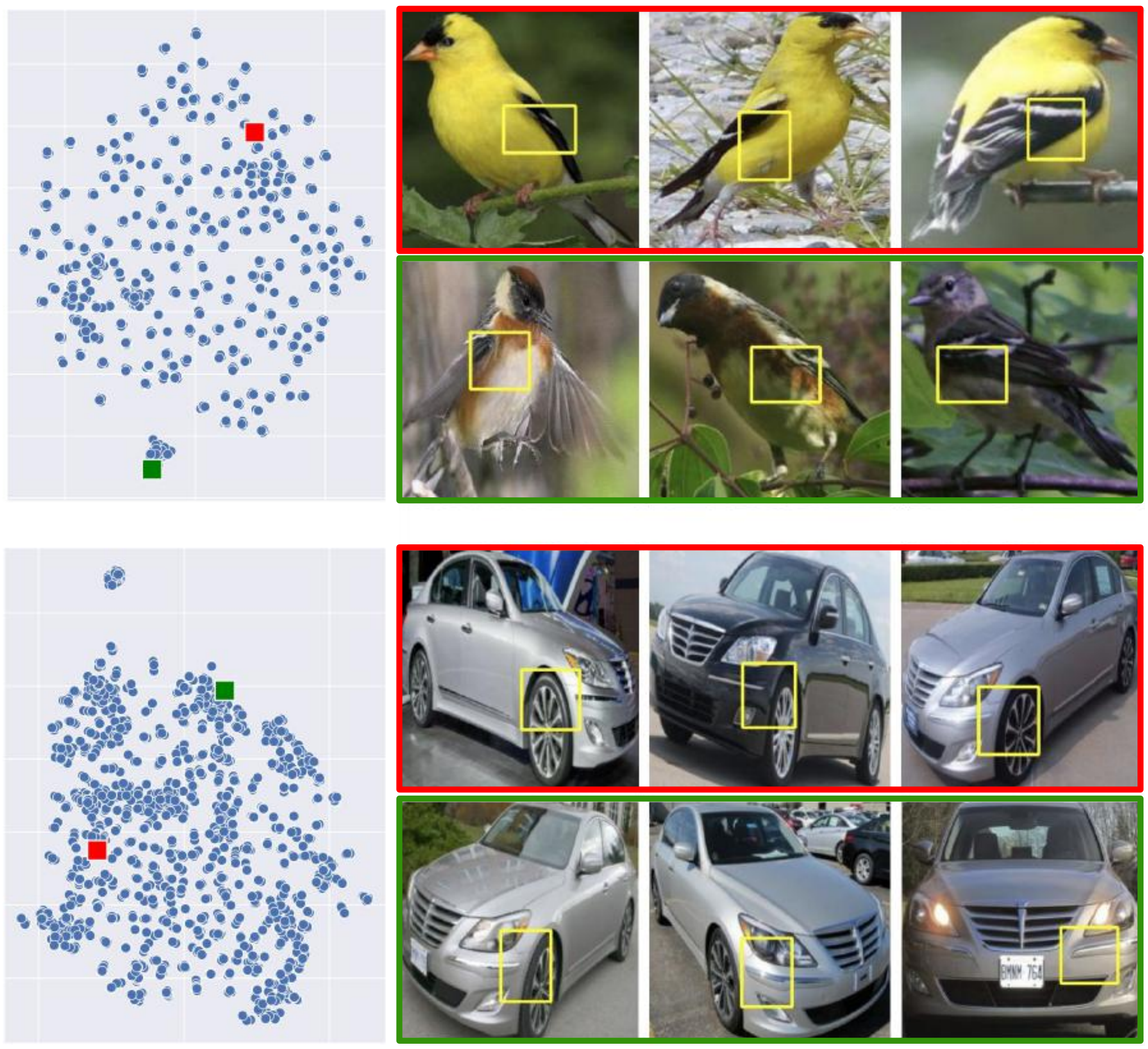


Figure 3.2: Our method aligns prototypical parts that are both similar and distant in the latent space. The left plots display tSNE projections of prototypes trained on CUB-200-2011 and Stanford Cars datasets, with the top and bottom parts representing each dataset, respectively. Two of these prototypes, marked as red and green squares, correspond to semantically similar prototypical parts: a bright belly with grayish wings (top) and a fender (bottom). Despite their semantic similarity, these prototypes are distant in the representation space. Our method enables the detection and alignement of such pairs. Image copied from our work [I]

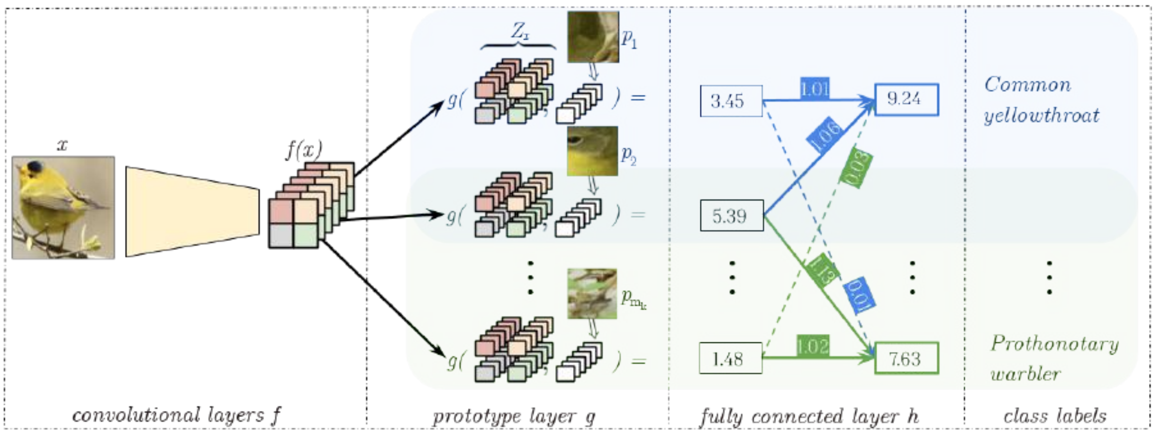


Figure 3.3: Architecture of ProtoPShare. Compared to ProtoPNet it consists of sharable prototypical parts by having multiple positive (close to 1) connections resulting in contribution of a single prototypical part to multiple data classes. Image copied from our work [I]

similarity comparing the difference between $g(z, p)$ and $g(z, \tilde{p})$ for all training patches $x \in X$. Specifically, the similarity for prototype pair $(p, \tilde{p}) \in P^2$ is defined as the compliance of the similarity scores computed for all patches:

$$d_{DD}(p, \tilde{p}) = \frac{1}{\sum_{x \in X} (g(z,p) - g(z,\tilde{p}))^2}. \tag{3.3}$$

This data-dependent similarity enables us to identify prototypes similar in their semantic meaning, even if they are distant in the representation space.

Results ProtoPShare achieves higher accuracy compared to ProtoPNet and its variations for different pruning rates, as shown in Figure 3.4. Moreover, ProtoPShare performs well, even with only 25% of the initial prototypes in the case of ResNet34. This results confirm that data-dependent merge-pruning can reduce the number of prototypical parts needed in classification and still preserve competitive accuracy of the model.

ProtoPShare uncovers class similarity by utilizing shared prototypes and visualizing it through a graph, as depicted in Figure 3.5. A user study was conducted to compare the effectiveness of data-dependent and data-independent approaches. Participants were presented with pairs of prototypes and asked to select the more consistent one. Out of 285 responses from 57 participants, 195 favored the data-dependent approach, 61 preferred data-independent, and 29 could not decide. This demonstrates that the data-dependent approach is considered more consistent than the data-independent one.

Summary. In this work, we introduced ProtoPShare [I], a model for interpretable image classification based on prototypical parts. We also proposed a new data-dependent similarity measure to identify semantic coherence between visual concepts, as well as a merge-pruning operation that enables sharing of prototypical parts. Our model's effectiveness was confirmed

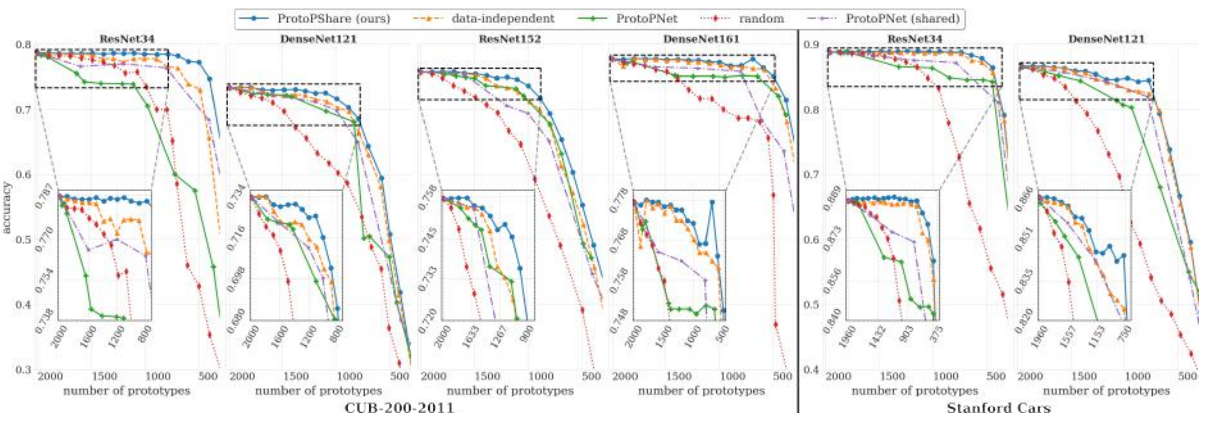


Figure 3.4: This figure shows the accuracy of ProtoPShare on the CUB-200-2011 dataset for different percentages of prototypes merged per pruning step, as the number of prototypes decreases. Higher accuracy is better. It can be observed that ProtoPShare preserves the original accuracy and depending on the backbone network, there is a different number of prototypical to be merged before the model collapse. Image copied from our work [I]

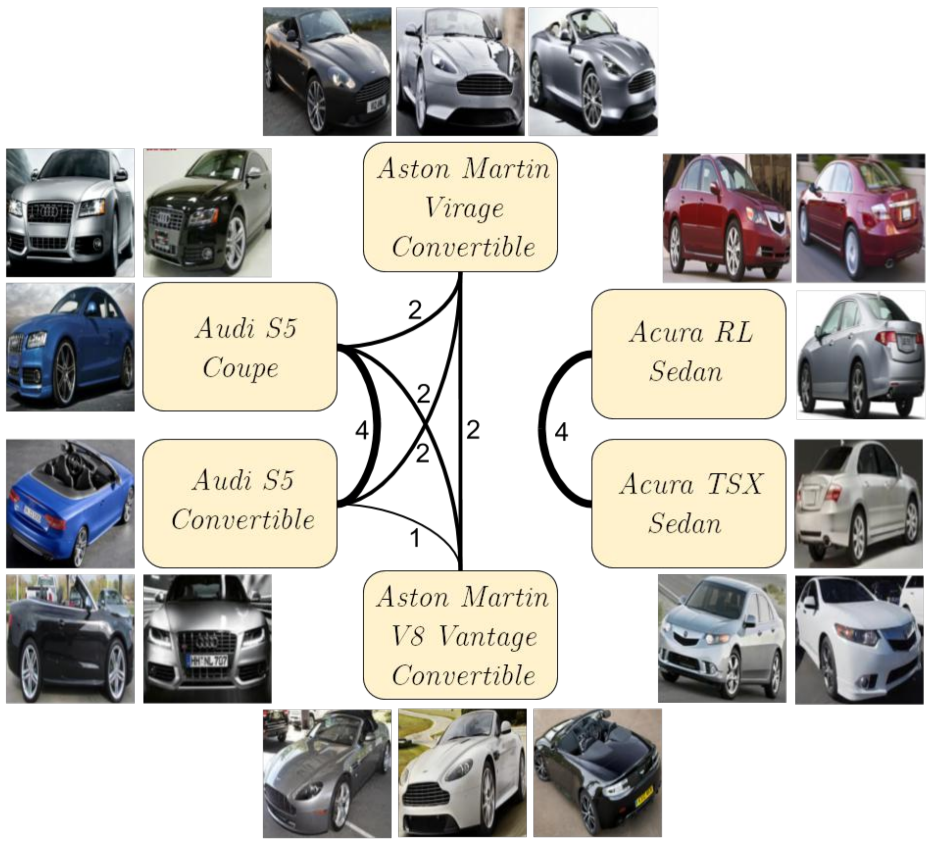


Figure 3.5: The inter-class similarity graph generated by ProtoPShare trained on the Stanford Cars dataset reveals how classes share prototypes. Each node represents a car class, and the edge strength between two nodes corresponds to the number of shared prototypes. For instance, the *Audi S5 Coupe* class shares four prototypes with the *Audi S5 Convertible* class but does not share any prototypes with the *Acura RL Sedan* class. The image illustrates ProtoPShare ability to detect inter-class similarities. Image copied from our work [I]

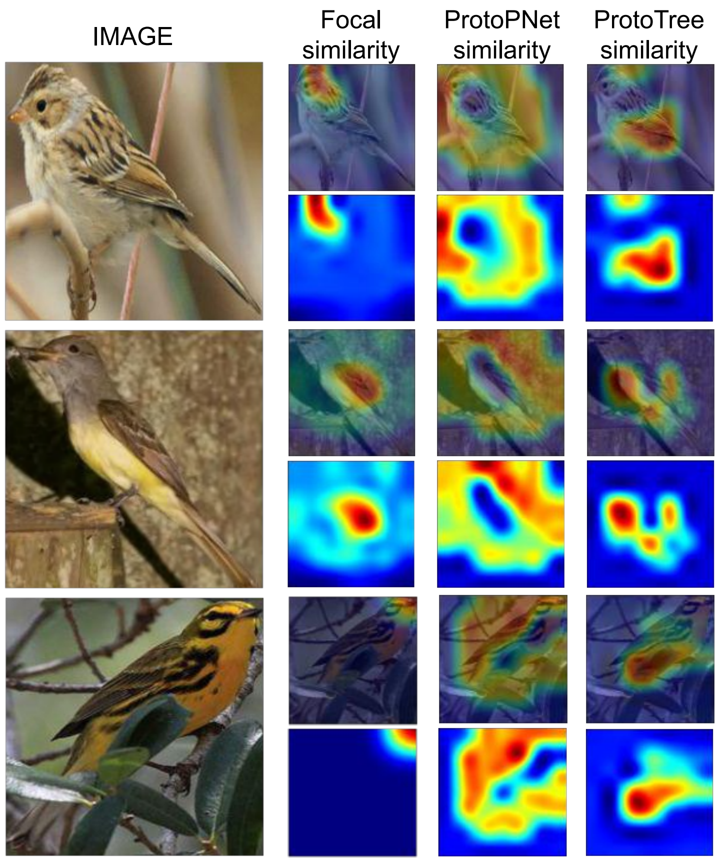


Figure 3.6: Focal similarity is a similarity metric focusing on salient visual features, making interpretation easier to comprehend compared to other metrics that are more distributed through the image. Image copied from our work [II]

on two datasets, and we validated the semantic similarities discovered by the model through a user study.

3.1.3. Learning prototypical parts sharing from scratch (ProtoPool) [II]

Motivation. ProtoPShare [I] and ProtoTree [53] were introduced as solutions to the scalability and unstable predictions of ProtoPNet [11] (discussed above). However, they both have drawbacks: ProtoPShare requires a previously trained ProtoPNet, which extends the training time, whereas ProtoTree builds a decision tree that may lead to explanations based only on prototype absence. For instance, a model can predict a "sparrow" because an image does not contain red feathers, a long beak, and wide wings, which is also true for many other species.

Contributions. To overcome these limitations, we proposed ProtoPool, a self-explainable prototype model for fine-grained image classification. ProtoPool introduces two novel mechanisms enhancing interpretability and training efficiency. Firstly, instead of using hard assignment of prototypes to classes, we employ a soft assignment represented by a distribution over the prototypes. This distribution is randomly initialized and binarized during training using the Gumbel-Softmax trick, eliminating the pruning step required in ProtoPNet, ProtoPShare,

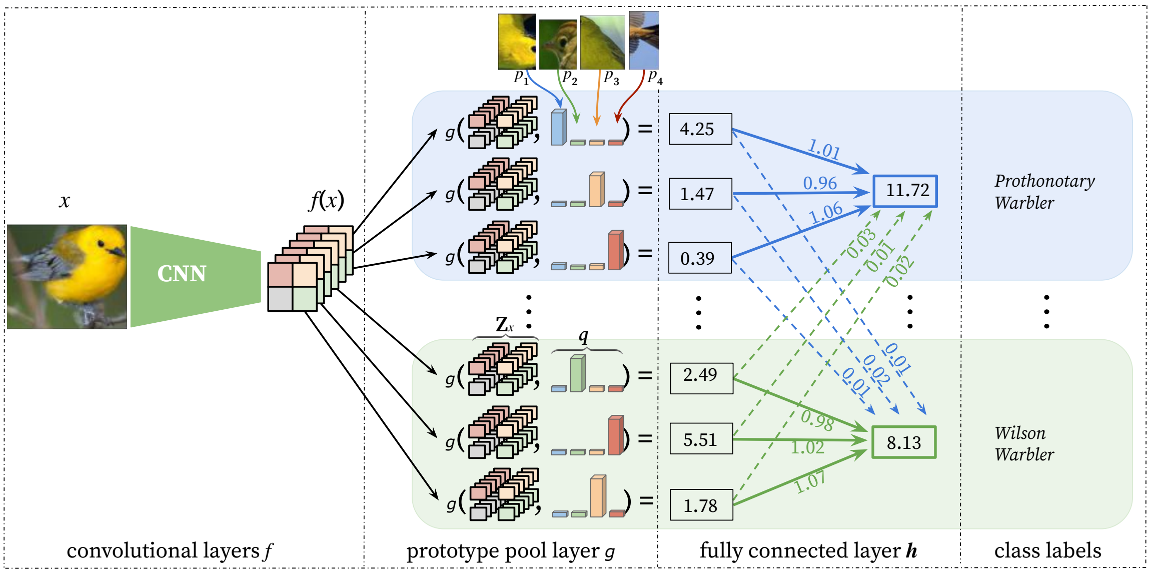


Figure 3.7: Archtecture of ProtoPool. Due to the introduction of slots q the model can learn the assignments of prototypical parts from the pool to data classes. It results in fewer prototypical parts used by the model while it maintains positive reasoning through specific initialization of the fully connected layer h. Image copied from our work [II].

and ProtoTree. Secondly, we introduce a focal similarity function focusing the model on the salient features (see Figure 3.6).

Details. The architecture of ProtoPool, illustrated in Figure 3.7, comprises three main components: convolutional layers f, a prototype pool layer g, and a fully connected layer h. The prototype pool layer g contains a pool of M trainable prototypes $P = \{p_i \in \mathbb{R}^D\}$ and T slots for each class. Each slot is implemented as a distribution $q_t \in \mathbb{R}^M$ of prototypes available in the pool, where q_t assigns a probability to each prototype of being assigned to slot t $(||q_t|| = 1)$. The fully connected layer h is initialized to enforce positive reasoning by setting the weights between each class c and its slots to 1, while the remaining weights of h are set to 0.

To obtain a prediction for a given input image $x \in X$, the convolutional layers f extract the image representation $f(x)$, which can be viewed as a set of $H \cdot W$ vectors of dimension D corresponding to different locations in the image (as shown in Figure 3.7). For the sake of clarity, we denote this set as Z_x. Subsequently, for each slot t, the prototype pool layer computes the aggregated similarity $g_t = \sum_{i=1}^{M} q_t^i g_{p_i}$ between Z_x and all prototypes in the pool, where q_t denotes the distribution of prototypes assigned to slot t, and g_{p_i} is defined in the next paragraph. g_t is then multiplied by the corresponding weight in the fully connected layer h and aggregated to produce the output logits. Finally, the logits are normalized using softmax to obtain the final prediction.

In ProtoPNet [11] and other models using prototypical parts, the similarity of point z to prototype p is defined as: $g_p(z) = \log(1 + \frac{1}{\|z-p\|^2})$, and the final activation of the prototype p with respect to image x is given by $g_p = \max_{z \in Z_x} g_p(z)$. It can be observed that such an approach has two possible disadvantages. Firstly, high activation can be obtained when all the elements in Z_x are similar to a prototype. It is undesirable because the prototypes can then concentrate on the background. The other negative aspect concerns the training process, as the gradient is passed only through the most active part of the image.

To prevent those behaviors in ProtoPool we introduce a novel focal similarity function widening the gap between maximal and average activation

$$g_p = \max_{z \in Z_x} g_p(z) - \operatorname*{mean}_{z \in Z_x} g_p(z), \tag{3.4}$$

as presented in Figure 2.2. The maximal activation of focal similarity is obtained if a prototype is similar to only a narrow area of the image x. Consequently, the constructed prototypes correspond to more salient features, and the gradient passes through all elements of Z_x.

To generate a prototype distribution q, one could apply softmax on the vector of size $\mathbb{R}^M$. However, this could lead to assigning many prototypes to one slot, decreasing interpretability. Therefore, to obtain distributions with only one probability close to 1, a differentiable $arg\max$ function is required. The Gumbel-Softmax estimator [32] provides a perfect match in this case. With M being the number of classes, for $q = (q^1, \dots, q^M) \in^M$ and $\tau \in (0, \infty)$, it is defined as:

$$Gumbel\text{-}softmax(q, \tau) = (y^1, \dots, y^M) \in^M,$$

where $y^i = \frac{\exp\big((q^i+\eta_i)/\tau\big)}{\sum_{m=1}^{M} \exp((q^m+\eta_m)/\tau)}$ and η_m for $m \in 1, .., M$ are samples drawn from standard Gumbel distribution. The Gumbel-Softmax distribution interpolates between continuous categorical densities and discrete one-hot-encoded categorical distributions, approaching the latter for low temperatures $\tau \in [0.1, 0.5]$.

To avoid the issue of one prototype being assigned to multiple slots of a single class and wasting the capacity of the prototype pool layer, we need to introduce additional constraints to the loss function. For this purpose we extend the loss function with

$$L_{orth} = \sum_{i<j}^{T} \frac{\langle q_i, q_j \rangle}{\|q_i\|_2 \cdot \|q_j\|_2}, \tag{3.5}$$

where $q_1, .., q_T$ are the distributions of a particular class. As a result, successive slots of a class are assigned to different prototypes.

Results Table 3.1 presents a comparison of our ProtoPool with other prototypical part-based models. In addition to the accuracy achieved by each model, we also report the number of

Table 3.1: Comparison of ProtoPool with other prototypical methods trained on the CUB-200-2011 and Stanford Cars datasets. It can be noted that ProtoPool achieves comparable results to models using much more prototypical parts and its superior over models with the same number of prototypical parts. Table copied from our work [II]

CUB-200-2011

Model	Arch.	Proto. #	Acc [%]
ProtoPool (ours)	R34	202	80.3±0.2
ProtoPShare [I]	R34	400	74.7
ProtoPNet [11]	R34	1655	79.5
TesNet [71]	R34	2000	82.7±0.2
ProtoPool (ours)	R152	202	81.5±0.1
ProtoPShare [I]	R152	1000	73.6
ProtoPNet [11]	R152	1734	78.6
TesNet [71]	R152	2000	82.8±0.2
ProtoPool (ours)	iNR50	202	85.5±0.1
ProtoTree [53]	iNR50	202	82.2±0.7
ProtoPool (ours)	Ex3	202×3	87.5
ProtoTree [53]	Ex3	202×3	86.6
ProtoPool (ours)	Ex5	202×5	**87.6**
ProtoTree [53]	Ex5	202×5	87.2
ProtoPNet [11]	Ex5	2000×5	84.8
TesNet [71]	Ex5	2000×5	86.2

Stanford Cars

Model	Arch.	Proto. #	Acc [%]
ProtoPool (ours)	R34	195	89.3±0.1
ProtoPShare [I]	R34	480	86.4
ProtoPNet [11]	R34	1960	86.1±0.2
TesNet [71]	R34	1960	92.6±0.3
ProtoPool (ours)	R50	195	88.9±0.1
ProtoTree [53]	R50	195	86.6±0.2
ProtoPool (ours)	Ex3	195×3	91.1
ProtoTree [53]	Ex3	195×3	90.5
ProtoPool (ours)	Ex5	195×5	91.6
ProtoTree [53]	Ex5	195×5	91.5
ProtoPNet [11]	Ex5	1960×5	91.4
TesNet [71]	Ex5	1960×5	**93.1**

prototypes utilized by the models. Specifically, we compare our approach with ProtoPNet [11], ProtoPShare [I], ProtoTree [53], and TesNet [71].

Our analysis shows that ProtoPool outperforms other models for the CUB-200-2011 dataset, even though some models utilize a significantly larger number of prototypical parts. Furthermore, for the Stanford Cars dataset, our model performs better than ProtoTree and ProtoPShare, despite having a similar number of prototypes, and moderately worse than TesNet, which uses ten times more prototypes. It is worth noting that the higher accuracy of TesNet could be attributed to the prototype orthogonality enforced during training.

We conducted a user study to investigate whether the use of focal similarity results in more salient prototypical parts. We asked a question: *"How salient is the feature pointed out by the AI system?"*. The task was to assign a score from 1 to 5 where 1 meant *"Least salient"* and 5 meant *"Most salient"*. Images were generated using prototypes obtained for ProtoPool with ProtoPNets similarity or with focal similarity and from a trained ProtoTree (see Figure 3.6). The study involved 40 participants hired via Amazon Mechanical Turk (AMT) system. They answered 60 questions (30 per dataset) presented in a random order, which resulted in 2400 answers.

The findings from our user study are presented in Figure 3.8. The results indicate that ProtoPool generated prototypes with higher saliency scores than the other models, as shown

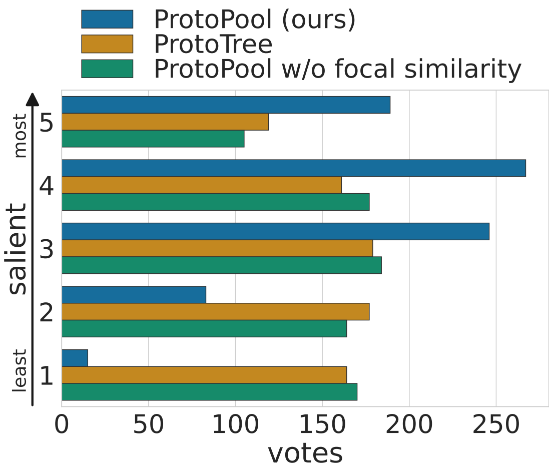


Figure 3.8: Comparison of scores from a user study on prototypes obtained for ProtoPool with and without focal similarity and for ProtoTree. The distribution of user votes for ProtoPool is skewed towards higher values. It means that ProtoPool is able to identify more salient and expressive prototypical parts than the baseline methods. Image copied from our work [II]

by the mostly higher scores ranging from 3 to 5. ProtoPool achieved a mean score of 3.66, while ProtoTree and ProtoPool without focal similarity obtained mean scores of 2.87 and 2.85, respectively.

Summary. In [II], we presented the ProtoPool architecture offering the unique advantage of sharing prototypical parts from scratch in contrast to two-stage ProtoPShare [I]. Additionally, ProtoPool's decision-making process is based on positive reasoning, which is crucial from a user's perspective and gives it an advantage over ProtoTree [53].

Furthermore, we introduced Focal Similarity, a similarity metric capturing more salient features than counterparts as demonstrated by the user study. Our approach improved the interpretability of prototypical parts-based methods while achieving comparable results to existing ones.

3.1.4. Generalization of prototypical parts methodology to the graph regression problem (ProGReST) [III]

Motivation. ProtGNN [80] is a method developed for the graph classification problem utilizing prototypical parts for graph data. While it works successfully in classification problems, it has limited ability to handle regression problems where there are no classes to which prototypical parts can be assigned. Additionally, prototypical-part-based methods use periodic projection operations [11, 80] to ensure that prototypes are close to the training data. However, in ProtGNN, the projection relies on an MCTS algorithm that is computationally expensive and may not always produce meaningful prototypes.

Contributions. To address this issue, we propose the Prototypical Graph Regression Soft Trees (ProGReST) model, designed for graph regression problems and apply it to the molecular property prediction. ProGReST uses prototypical parts and combines them with Soft Neural Trees [22] to construct the prototypical parts model. To address shortcomings of ProtGNN, we reduce training time by introducing proxy projection. However, such a model can be difficult to train that is why define a set of regularizes making the ProGReST interpretable and accurate.

Details. The ProGReST architecture, illustrated in Fig. 3.9, is composed of a graph representation network f, a prototypical regression soft tree layer t, and the final layer h. The prototype regression soft tree layer contains $2^{\mathcal{H}-1}$ prototypes $p_j \in \mathbb{R}^D$, where $\mathcal{H}$ is the depth of the tree. The number of prototypes is the same as the number of nodes of the tree because there is only one prototypical part in each node. All nodes have two children and the tree contains $2^{\mathcal{H}}$ leaves. Each leaf l_i calculates only one value $y_{l_i} \in \mathbb{R}$. The prototypes are trainable parameters.

The regression task involves a dataset of V graphs $x_i \in \mathcal{X}$ with corresponding labels $y_i \in \mathcal{Y} \subset \mathbb{R}$. The graph $x_i \subset \mathbb{R}^{\mathcal{N} \times \mathcal{E}}$ consists of a set of nodes $\mathcal{N}$ and a set of edges $\mathcal{E}$. The model takes an input graph $x_i \in \mathcal{G}$ and returns its predicted value $\hat{y}_i \in \mathcal{Y}$. Before a graph is processed by the model, it is encoded as an array of shape $N \times W$, where N is a number of nodes and W is the size of the vector encoding each node. Then, the graph representation network is used to calculate the input embedding $Z_{x_i} = f(x_i)$ where $Z_{x_i} \in \mathbb{R}^{N \times D}$, D is the prototypical part vector length. The graph representation network is a graph convolutional network (GCN) [38] followed by a node-wise convolution with the sigmoid activation at the end, used to map the input features to the prototype space.

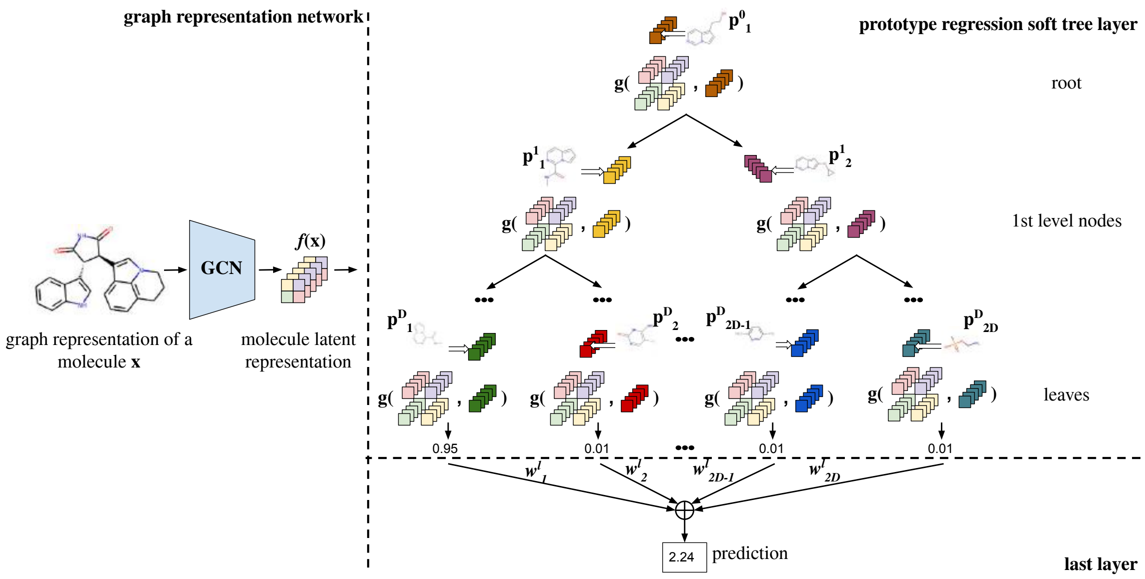
For each input $x_i \in \mathcal{G}$, ProGReST calculates the prototype activation as the similarity between the prototypical part p and the latent graph representation $z \in \mathbb{R}^C$ for each graph node. Then, it calculates the maximum activation a given prototype in the input graph:

$$\hat{z} = \max_{z \in Z_{x_i}} e^{-||z-p||_2} \tag{3.6}$$

Unlike decision trees with nodes routing to only one child, Soft Decision Trees distribute the signal to both children simultaneously, with the probability adding up to 1. We use prototype activation as a probability of routing to a right node as it is in [53]. The probability of routing to the left node is a complement to 1 and equals $1 - \hat{z}$.

To determine the probability in a leaf l_k, we need to traverse through the path $\mathcal{P}_i$ consisted of its parents:

$$l_k(x) = \prod_{\hat{n} \in \mathcal{P}_k} \hat{z}_{\hat{n}}(x), \tag{3.7}$$



graph representation network
prototype regression soft tree layer
root
1st level nodes
leaves
last layer
GCN
f(x)
graph representation of a molecule x
molecule latent representation
0.95
0.01
0.01
0.01
prediction
2.24

Figure 3.9: Archtecture of ProGReST. Firstly, model generates latent representation of the molecular graph, then prototypical parts activation are calculated and aggregated to obtain a prediction. Thanks to the usage of soft neural tree, we were able to generalize prototypical parts methodology to the regression problem. Image copied from our work [III]

where $\hat{z}_{\hat{n}}$ is the similarity value in node $\hat{n}$. Then, the final prediction is made by summing up probabilities from the leaves multiplied by weights w_k^l of the last layer h:

$$\hat{y}(x) = \sum_{l_k \in \ell} w_k^l \cdot l_k(x). \tag{3.8}$$

In order to ensure that our prototypes are representative of the training dataset distribution, a projection is needed to swap the prototypical parts with a vector from the graph's latent representation. We propose proxy projection for periodic prototypical parts assignments during the training. The proxy projection works by finding the closest vector from the latent graph representation to a given prototypical part and replacing it. The proxy projection is defined as follows:

$$p \leftarrow \arg\min_{z \in Z_j} ||z - p_j||_2, \text{ where } Z = \hat{z} : \forall i, \hat{z} \in f(x_i).$$

Results. As Table. 3.2 shows, our ProGReST model not only outperforms its baseline model (GCN) but also achieves the best results on four of the five datasets. The prototypical-part-based approach for molecular activity prediction not only brings the interpretability of the predictions into the process, but also achieves superior results. This is in contrast to other prototypical-part-based methods such as [53] in computer vision where the introduction of interpretability reduces the model accuracy.

Table 3.2: Results of molecular property prediction. ProGrest achieves the state-of-the-art results on all except one datasets. Also, the Table shows that it can work with different graph neural networks backbone such as transformers (RMAT). Table copied from our work [III].

Method	**Caco-2** ↓	**PPBR** ↓	**LD50** ↓	**VDss** ↑	**HL** ↑
RDKit2D + MLP [30]	0.393 ± 0.024	9.994 ± 0.319	0.678 ± 0.003	0.561 ± 0.025	0.184 ± 0.111
AttrMasking [29]	0.546 ± 0.052	10.075 ± 0.202	0.685 ± 0.025	0.559 ± 0.019	0.151 ± 0.068
Morgan + MLP [30]	–	12.848 ± 0.362	0.649 ± 0.019	0.493 ± 0.011	0.329 ± 0.083
ContextPred [29]	0.502 ± 0.036	9.445 ± 0.224	0.669 ± 0.030	0.485 ± 0.092	0.129 ± 0.114
NeuralFP [41]	0.530 ± 0.102	9.292 ± 0.384	0.667 ± 0.020	0.258 ± 0.162	0.177 ± 0.165
AttentiveFP [75]	0.401 ± 0.032	9.373 ± 0.335	0.678 ± 0.012	0.241 ± 0.145	0.085 ± 0.068
CNN [30]	0.446 ± 0.036	11.106 ± 0.358	0.675 ± 0.011	0.226 ± 0.114	0.038 ± 0.138
SimGCN [8]	–	–	–	0.582 ± 0.031	**0.392** ± 0.065
ProGReST+GCN (Our)	0.367 ± 0.022	9.722 ± 0.200	0.611 ± 0.009	0.586 ± 0.012	0.295 ± 0.058
GCN (Baseline) [38]	0.599 ± 0.104	10.194 ± 0.373	0.649 ± 0.026	0.457 ± 0.050	0.239 ± 0.100
ProGReST+RMAT (Our)	**0.360**± 0.069	**9.256**± 0.287	0.597 ± 0.072	**0.620** ± 0.069	0.337 ± 0.049
RMAT [51]	0.363 ± 0.030	9.909 ± 0.388	**0.569** ± 0.092	0.487 ± 0.083	0.360 ± 0.063

Table 3.3: Comparison of the time needed to perform the projection. Our novel proxy projection is much faster (two orders of magnitude) than the ProtoGNN MCTS-based projection resulting in a significant speed-up of the model's training. Table copied from our work [III].

Depth	**4**	**5**
Proxy Projection	$\mathbf{10.1s}$	$\mathbf{16.2s}$
MCTS-based Projection	$1760.0s$	$4371.5s$

We compared the computational efficiency of the Proxy and MCTS-based projections. Unlike ProtGNN, ProGReST can identify the critical vector of graph latent representation using a similarity function as we do not perform pooling of latent representation. From the results shown in Tab. 3.3, we observe that MCTS-based projection is significantly slower than our Proxy Projection.

To establish the interpretability of the prototypical parts learned by our model, we conducted a qualitative study where a chemist evaluated the relevance of the discovered molecular features. Initially, a chemist was presented with a small subset of compounds from the Caco-2 dataset, and the atoms forming the learned prototypical parts were highlighted. Subsequently, the compounds containing the same prototypes were displayed to verify the alignment between the similarity function utilized in the model and the chemist's knowledge-based intuition.

Several useful prototypical parts were identified through visual inspection. For instance, in the Caco-2 dataset, there are prominent molecular features correlating well with the ability of compounds to penetrate the epithelial barrier. The model detected a set of ketone and amine groups impacting the compound's hydrophilicity and can form hydrogen bonds with the lipid layers, which can significantly alter permeability. Other prototypical parts included aromatic rings and aliphatic side chains, which are also related to hydrophobicity and can be correlated with the compound's bulkiness, decreasing its ability to pass the barrier. These structures are depicted in Fig. 3.10.

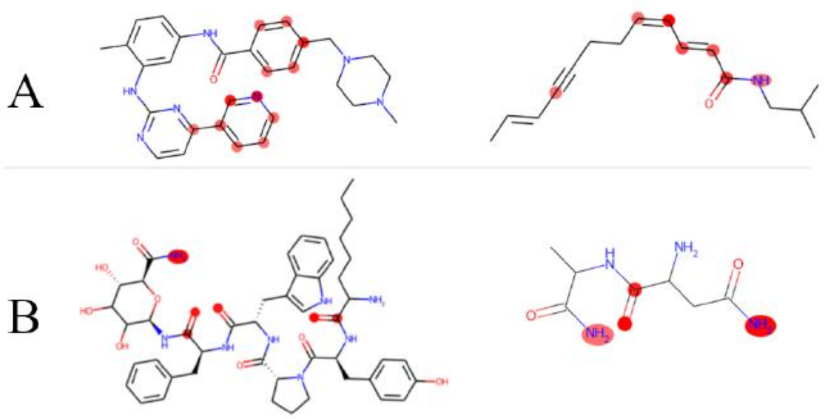


Figure 3.10: Two examples of the learned prototypical parts. The atoms marked with a red circle are part of the prototype. The compounds on the left are the reference compounds, and the ones on the right are matched by the similarity of the prototypical parts. In prototype A, we see aromatic rings and conjugated bonds (alternating single and double bonds). Prototype B consists of ketones (=O) and amides (-C(=O)NH_2). Image copied from our work [III].

Summary. Our work generalizes the prototypical-part to graph regression problem, and we demonstrated its effectiveness on the molecular property prediction task. The results we obtained show that our method outperforms existing approaches and achieves state-of-the-art results. Additionally, our approach provides insights into the regression model as an interpretable subgraph and tree structure, enabling us to better understand the underlying mechanisms.

We also introduced a new technique called proxy-projection, which significantly reduces the training time required for our model. This innovation is a significant contribution to the field, as it makes the training of large-scale models feasible and practical.

3.1.5. Summary

This section outlines the contributions made by the Ph.D. candidate in advancing the field of interpretable deep learning models through the use of prototypical parts. In the following section, we delve into the candidate's contributions towards improving interpretability in multiple-instance learning problems, including the generalization of prototypical parts to this domain. Those models directly improve the transparency and trustworthiness of deep learning-based models for various set of problems while maintain or achieve superior performance compared to the state-of-the-art.

3.2. Interpretable multiple instance learning

This chapter aims to provide a comprehensive coverage of the multiple-instance learning problem and its basic method, Attention-based Multiple Instance Learning Pooling (AbMILP) [31] since the contributions of Ph.D candidate are strongly related to them (Section 3.2.1). Subsequent sections show the limitations of AbMILP and present the solutions proposed by SA-AbMILP [III] (Section 3.2.2) and ProtoMIL [IV] (Section 3.2.3).

3.2.1. Preliminaries

Multiple Instance Learning (MIL) [15] is a type of supervised machine learning where each sample is represented by a bag of feature vectors, called instances, of a fixed length L. In contrast to typical supervised learning, where each sample has a separate feature vector, in MIL, a sample (called bag) may contain a variable number of objects (called instances). The standard MIL assumption is that the label of the bag, denoted by $y \in \{0,1\}$, is positive if at least one of its instances is positive, where each instance $\mathbf{h_i}$ has a hidden binary label $y_i \in \{0,1\}$.

$$y = \begin{cases} 0, & \text{iff } \sum\limits_{i=1}^{n} y_i = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{3.9}$$

The standard assumption adopted by AbMILP is too restrictive and not suitable for many real-world problems. For instance, consider the evaluation of digestive tract health using the NHI scoring system [45], where a biopsy is given a score of 2 if more than 50% of crypts are infiltrated with neutrophils and there is no damage or ulceration in the epithelium. Such tasks require more complex types of MIL [21] relying on multiple assumptions (or concepts).

Let $\hat{C} \subseteq C$ be the set of required instance-level concepts, and let $p : X \times C \to N$ be the function counting how often the concept $c \in C$ occurs in the bag $x \in N$. Then, in *presence-based assumption*, the bag is positive if each concept occurs at least once:

$$y = \begin{cases} 1, & \text{iff for each } c \in \hat{C} : p(x,c) \geq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{3.10}$$

In the case of *threshold-based assumptions*, the bag is positive if concept $c_i \in C$ occurs at least $t_i \in \mathbb{N}$ times:

$$y = \begin{cases} 1, & \text{iff for each } c_i \in \hat{C} : p(x,c_i) \geq t_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3.11}$$

Attention-based MIL Pooling (AbMILP) [31] is a type of pooling method computing the weighted average of instances in a bag using a neural network to determine the instance weights.

Mathematically, given a bag $x = \{\mathbf{h_i}\}_{i=1}^{N}, \mathbf{h_i} \in \mathbb{R}^{L\times 1}$, AbMILP is defined as follows:

$$\mathbf{z} = \sum i = 1^n a_i \mathbf{h_i}, \tag{3.12}$$

where a_i is the weight assigned to instance i and is computed using the following softmax function:

$$a_i = \frac{\exp\left(\mathbf{w}^T tanh(\mathbf{V}\mathbf{h_i})\right)}{\sum_j^n \exp\left(\mathbf{w}^T tanh(\mathbf{V}\mathbf{h_j})\right)}, \tag{3.13}$$

where $\mathbf{w} \in \mathbb{R}^{M\times 1}$ and $\mathbf{V} \in \mathbb{R}^{M\times L}$ are trainable neural network layers, and $tanh$ is used to prevent exploding gradients. Additionally, the weights a_i sum up to 1 to accommodate bags of varying sizes, and the instances are randomly ordered within the bag to prevent overfitting.

3.2.2. Interpretable aggregation with self-attention (SA-AbMILP) [IV]

Motivation. Attention-based MIL Pooling (AbMILP), introduced by Ilse et al. [31], has found wide application in medical image analysis [44, 48, 72], particularly for whole-slide image assessment. However, AbMILP lacks the ability to model dependencies between instances in a bag, despite its effectiveness in aggregating information from varying numbers of instances.

Contributions. To overcome this limitation, we propose a method combining self-attention mechanism with Attention-based MIL Pooling. This approach captures global dependencies between instances in a bag, which have been shown to be beneficial [82], while simultaneously aggregating them into a fixed-sized vector for use in successive layers of the network. The resulting vector can be used for regression, and for binary and multi-class classification problems.

Details. Our method, called SA-AbMILP, is a pipeline combining Self-Attention with Attention-based MIL Pooling in four steps. Firstly, the bag's images are fed through a Convolutional Neural Network (CNN) to obtain their representations. Subsequently, the self-attention module is applied to these representations to integrate dependencies between the instances. This can be achieved using dot product or other kernels (details about kernels are provided in work [III]). The resulting feature vectors, which include integrated dependencies, are passed to the AbMILP module to obtain a fixed-sized vector for each bag. This vector can then be passed to successive Fully-Connected (FC) layers of the network. The entire pipeline is illustrated in Fig. 3.11.

The Self-Attention (SA) mechanism is used to identify the interdependencies between instances within a bag. It enhances the instance representations by incorporating knowledge from the entire bag, which is crucial for identifying the number of instances of the same concept and their relationships. SA transforms each instance into two feature spaces, namely, keys

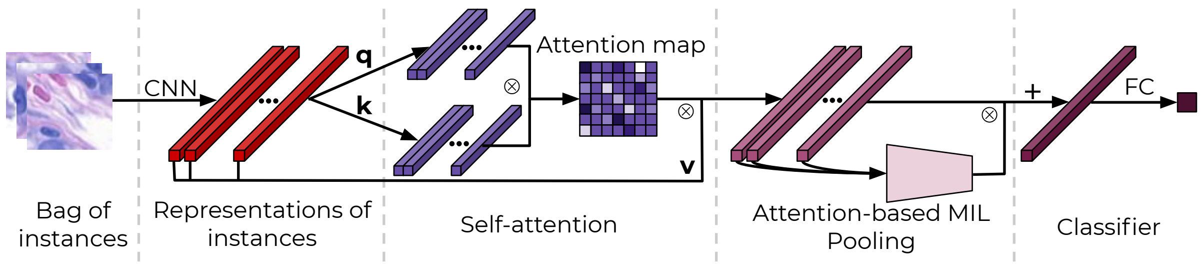


Figure 3.11: SA-AbMILP architecture. Thanks to the use of self-attention module, SA-AbMILP can model relationships between instances within a bag. Image copied from our work [IV].

$\mathbf{k_i} = \mathbf{W_k h_i}$ and queries $\mathbf{q_j} = \mathbf{W_q h_j}$, and calculates their similarity score $s_{ij} = \langle \mathbf{k}(\mathbf{h_i}), \mathbf{q}(\mathbf{h_j}) \rangle$ using dot product or other kernels. This similarity score is used to compute attention weights $\beta_{\mathbf{j},\mathbf{i}}$ for each instance, indicating how much the model focuses on the i^{th} instance when processing the j^{th} one. The attention layer's output is defined for each instance separately as:

$$\hat{\mathbf{h}}_\mathbf{j} = \gamma \mathbf{o_j} + \mathbf{h_j}, \text{ where } \mathbf{o_j} = \sum_{i=1}^{N} \beta_{j,i} \mathbf{W_v h_i}, \tag{3.14}$$

Here, $\mathbf{W_q}, \mathbf{W_k} \in \mathbb{R}^{\bar{L} \times L}$, and $\mathbf{W_v} \in \mathbb{R}^{L \times L}$ are trainable layers. The output $\mathbf{o_j}$ is obtained by weighting the instance representations with the computed attention weights $\beta_{\mathbf{j},\mathbf{i}}$ and passing them through a trainable layer $\mathbf{W_v}$. The parameter $\bar{L} = L/8$ is chosen based on the results presented in [79]. Moreover, γ is a trainable scalar initialized to 0.

Results. Table 3.4 presents the results of our method on histological datasets [25, 66]. In both datasets, our method, with or without kernel extension, outperforms the baseline methods in terms of the Area Under the ROC Curve (AUC). Furthermore, our method achieves the highest recall, which is crucial in medical applications. To shed light on why our method outperforms AbMILP, we compare the weights of patches in the average pooling. These patches contribute the most to the final score and thus should be thoroughly investigated by pathologists. Lastly, we observe that while kernels generally enhance performance, none of them is significantly superior.

Figure 3.12 illustrates a comparison between the explanations obtained from AbMILP and SA-AbMILP for a toy dataset created with MNIST digits. The dataset follows the presence-based MIL assumption, where a bag is considered positive if it contains both 9 and 7 digits. It can be observed that in SA-AbMILP, the self-attention module strengthens the representations of both 9s and 7s, resulting in higher weights in the aggregation operator as compared to AbMILP.

Table 3.4: Results for breast and colon cancer datasets (mean and standard error of the mean over 5 repetitions). It can be noticed that SA-AbMILP and its kernel variations achieved state-of-the-art results. However, there is no superior kernel in the results, meaning that it is a hyperparameter of the model. Description of acronyms in [IV]. Table copied from our work [IV]

breast cancer dataset					
method	accuracy	precision	recall	F-score	AUC
instance+max	61.4 ± 2.0	58.5 ± 3.0	47.7 ± 8.7	50.6 ± 5.4	61.2 ± 2.6
instance+mean	67.2 ± 2.6	67.2 ± 3.4	51.5 ± 5.6	57.7 ± 4.9	71.9 ± 1.9
embedding+max	60.7 ± 1.5	55.8 ± 1.3	54.6 ± 7.0	54.3 ± 4.2	65.0 ± 1.3
embedding+mean	$\mathbf{74.1 \pm 2.3}$	74.1 ± 2.3	65.4 ± 5.4	$\mathbf{68.9 \pm 3.4}$	79.6 ± 1.2
AbMILP	71.7 ± 2.7	$\mathbf{77.1 \pm 4.1}$	68.6 ± 3.9	66.5 ± 3.1	85.6 ± 2.2
SA-AbMILP	$\mathbf{75.1 \pm 2.4}$	$\mathbf{77.4 \pm 3.7}$	74.9 ± 3.7	$\mathbf{69.9 \pm 3.0}$	$\mathbf{86.2 \pm 2.2}$
GSA-AbMILP	$\mathbf{75.8 \pm 2.2}$	$\mathbf{79.3 \pm 3.3}$	74.7 ± 3.4	$\mathbf{72.5 \pm 2.5}$	$\mathbf{85.9 \pm 2.2}$
IQSA-AbMILP	$\mathbf{76.7 \pm 2.3}$	$\mathbf{78.6 \pm 4.2}$	75.1 ± 4.2	$\mathbf{66.6 \pm 4.3}$	$\mathbf{85.9 \pm 2.2}$
LSA-AbMILP	65.5 ± 2.9	62.5 ± 3.7	$\mathbf{89.5 \pm 2.6}$	$\mathbf{68.5 \pm 2.6}$	$\mathbf{86.7 \pm 2.1}$
MSA-AbMILP	$\mathbf{73.8 \pm 2.6}$	$\mathbf{78.4 \pm 3.9}$	73.8 ± 3.6	$\mathbf{69.4 \pm 3.4}$	85.8 ± 2.2

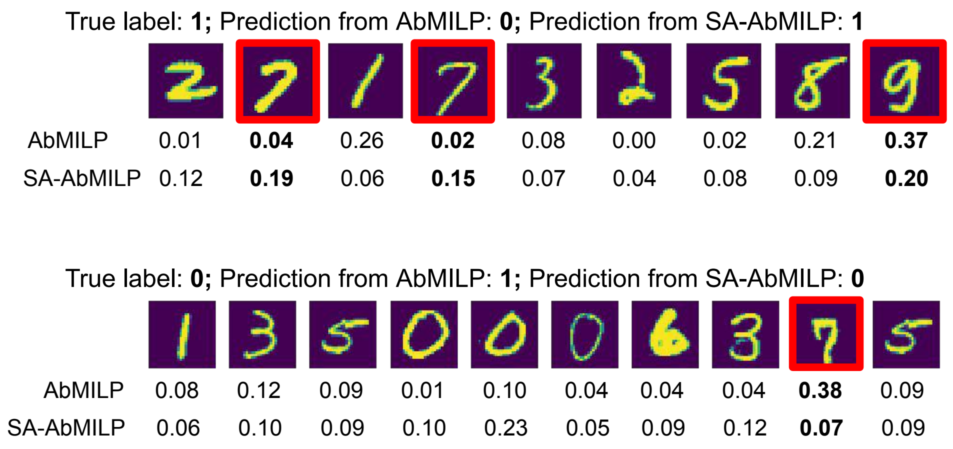
colon cancer dataset					
method	accuracy	precision	recall	F-score	AUC
instance+max	84.2 ± 2.1	86.6 ± 1.7	81.6 ± 3.1	83.9 ± 2.3	91.4 ± 1.0
instance+mean	77.2 ± 1.2	82.1 ± 1.1	71.0 ± 3.1	75.9 ± 1.7	86.6 ± 0.8
embedding+max	82.4 ± 1.5	88.4 ± 1.4	75.3 ± 2.0	81.3 ± 1.7	91.8 ± 1.0
embedding+mean	86.0 ± 1.4	81.1 ± 1.1	80.4 ± 2.7	85.3 ± 1.6	94.0 ± 1.0
AbMILP	88.4 ± 1.4	$\mathbf{95.3 \pm 1.5}$	$\mathbf{84.1 \pm 2.9}$	$\mathbf{87.2 \pm 2.1}$	97.3 ± 0.7
SA-AbMILP	$\mathbf{90.8 \pm 1.3}$	$\mathbf{93.8 \pm 2.0}$	$\mathbf{87.2 \pm 2.4}$	$\mathbf{89.0 \pm 1.9}$	$\mathbf{98.1 \pm 0.7}$
GSA-AbMILP	88.4 ± 1.7	$\mathbf{95.2 \pm 1.7}$	83.7 ± 2.8	$\mathbf{86.9 \pm 2.1}$	$\mathbf{98.5 \pm 0.6}$
IQSA-AbMILP	89.0 ± 1.9	$\mathbf{93.9 \pm 2.1}$	$\mathbf{85.5 \pm 3.0}$	$\mathbf{86.9 \pm 2.5}$	96.6 ± 1.1
LSA-AbMILP	84.8 ± 1.8	$\mathbf{92.7 \pm 2.7}$	71.1 ± 4.6	73.4 ± 4.3	95.5 ± 1.7
MSA-AbMILP	$\mathbf{89.6 \pm 1.6}$	$\mathbf{94.6 \pm 1.5}$	$\mathbf{85.7 \pm 2.7}$	$\mathbf{87.9 \pm 1.8}$	$\mathbf{98.4 \pm 0.5}$

Summary. Our work introduced the SA-AbMILP [IV] architecture, which is capable of modeling dependencies between instances within a bag. We also presented a generalization of the attention-based approach to encompass more MIL assumptions, such as threshold- and presence-based.

One of the key advantages of our approach is ability to maintain the interpretability component of attention pooling while achieving state-of-the-art results. This is a significant contribution to the field, as it enabled us to better understand the underlying mechanisms of the model's decision-making process while still achieving high performance.

3.2.3. Local and global interpertability for MIL (ProtoMIL) [V]

Motivation. In recent years, deep learning has led to the integration of MIL with several neural network-based models, including convolutional blocks, embedding all instances of a bag and aggregate their embeddings [31,42,59,74]. While some aggregation methods highlight important instances for prediction interpretation, they typically fail to explain the cause of their importance [31,42,59]. Previous attempts to explain MIL models usually introduce additional bias into the explanation or require extra input [7,9,43,57].


True label: **1; Prediction from AbMILP: 0; Prediction from SA-AbMILP: 1**

AbMILP	0.01	**0.04**	0.26	**0.02**	0.08	0.00	0.02	0.21	**0.37**
SA-AbMILP	0.12	**0.19**	0.06	**0.15**	0.07	0.04	0.08	0.09	**0.20**

True label: **0; Prediction from AbMILP: 1; Prediction from SA-AbMILP: 0**

AbMILP	0.08	0.12	0.09	0.01	0.10	0.04	0.04	0.04	**0.38**	0.09
SA-AbMILP	0.06	0.10	0.09	0.10	0.23	0.05	0.09	0.12	**0.07**	0.09

Figure 3.12: This image shows the comparison of the explanations generated by AbMILP and SA-AbMILP on a MIL dataset with a presence-based MIL assumption. The bag is considered positive if it contains both the digits 7 and 9. As the image shows, SA-AbMILP is able to correctly capture the most important instances within a bag compared to the AbMILP model. Image copied from our work [IV]

Contributions. To address these shortcomings, we introduce ProtoMIL, which builds on case-based reasoning, a natural explanation strategy used by humans [40]. It addresses the problem of gigapixel images classification, e.g. whole slide images (WSI) of biopsies. This classification can be defined under the MIL framework, where we divide each WSI into patches (to obtain instances) and analyze their similarity to trainable set prototypical parts of positive and negative data classes, as in [11]. The trainable prototypes enable ProtoMIL to automatically derive visual concepts from training set of positive and negative instances with the usage only of a bag label. We then apply an attention pooling operator to aggregate these similarities over instances to obtain a bag-level representation classified with an additional neural layer.

ProtoMIL is different from non-MIL approaches as it incorporates an aggregation layer and a novel regularization technique encouraging the model to derive prototypes from instances responsible for the positive label of a bag, which is challenging due to their concealed and underrepresented nature.

Details. To obtain a representation of an image, we first divide it into patches. However, since we do not know which patches correspond to the given data class, we employ Multiple Instance Learning (MIL) where a bag of instances (in our case, patches) has only one label for the whole bag. This bag is then passed through the four modules of ProtoMIL, as shown in Fig. 3.13: the convolutional network f_{conv}, the prototype layer f_{proto}, the attention pooling operator a, and the fully connected last layer g. The convolutional and prototype layers process single instances, while the attention pooling and the last layer work at the bag level. More precisely, given a bag of patches $x = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$, each $\mathbf{x}_i \in X$ is pushed through convolutional layers to obtain their representations $F = \{f_{conv}(\mathbf{x}_1), \ldots, f_{conv}(\mathbf{x}_k)\}$. As $f_{conv}(\mathbf{x}_i) \in H \times W \times D$, for the clarity

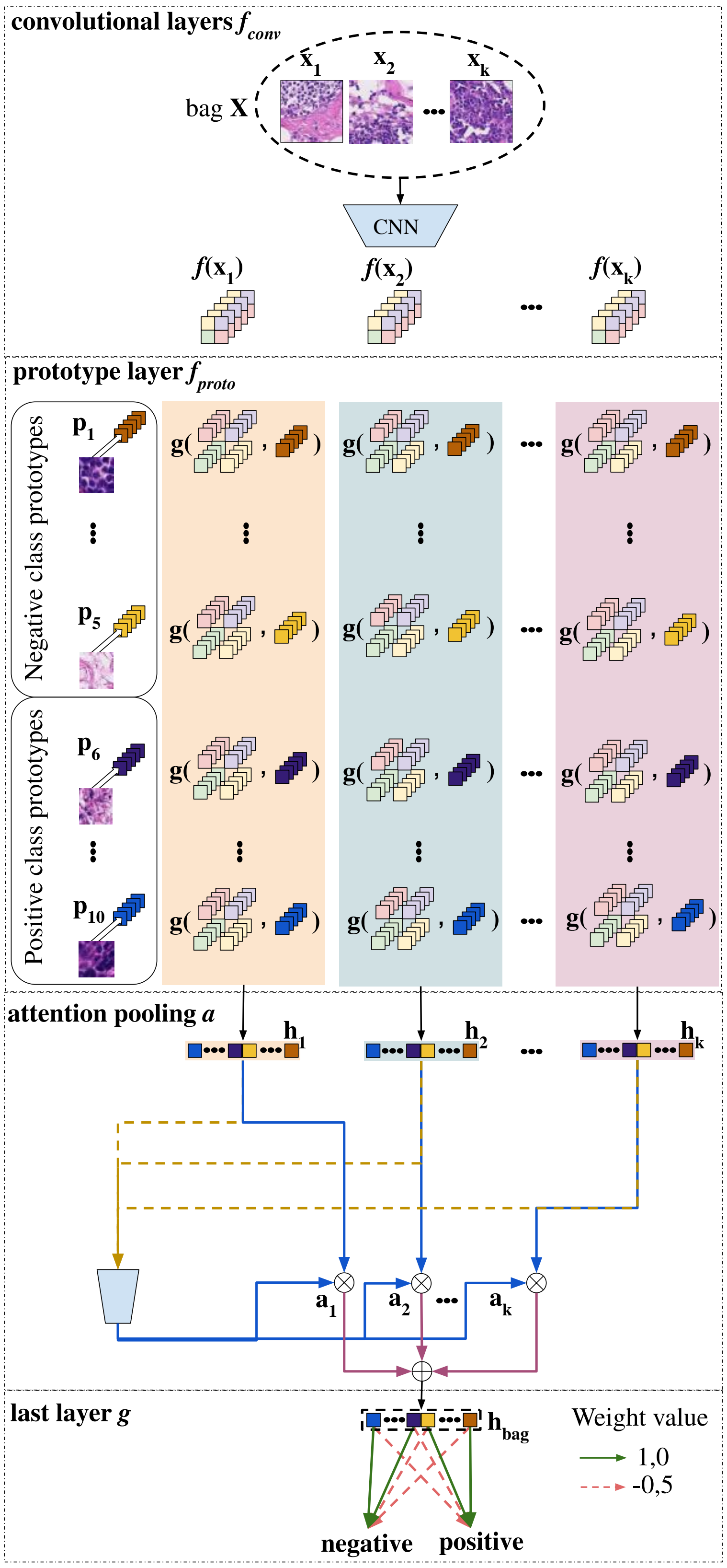


Figure 3.13: Architecture of ProtoMIL. Thanks to the use of prototypical parts, the ProtoMIL assures not only local interpretability (with attention pooling) but also the global one. Image copied from our work [V].

Table 3.5: Effectiveness of ProtoMIL on thre large-scale histopathology datasets. It can be noticed that ProtoMIL offers the best interpretability features while it still achieves comparable results to other methods, even much more complex such as transformes (TransMIL). Notice that values for comparison marked with “*” and “**” are taken from [42] and [64], respectively. Table copied from our work [V]

Method	Camelyon16		TCGA-NSCLC		TCGA-RCC		Inter.
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	
instance+mean*	79.84%	0.762	72.82%	0.840	90.54%	0.978	-
instance+max*	82.95%	0.864	85.93%	0.946	93.78%	0.988	+
MILRNN*	80.62%	0.807	86.19%	0.910	-	-	-
ABMILP*	84.50%	0.865	77.19%	0.865	89.34%	0.970	++
DSMIL*	86.82%	0.894	80.58%	0.892	92.94%	0.984	++
CLAM-SB**	87.60%	0.881	81.80%	0.881	88.16%	0.972	+
CLAM-MB**	83.72%	0.868	84.22%	0.937	89.66%	0.980	+
TransMIL**	**88.37%**	0.931	**88.35%**	**0.960**	**94.66%**	**0.988**	+
ProtoMIL (our)	87.29%	**0.935**	83.66%	0.918	92.79%	0.961	+++

of description, let $Z_{\mathbf{x}_i} = \{\mathbf{z}_j \in f_{conv}(\mathbf{x}_i) : \mathbf{z}_j \in \mathbb{R}^D, j = 1..HW\}$. Then, the prototypical part layer computes vector $\mathbf{h}_i$ of similarity scores [11] between each embedding $f_{conv}(\mathbf{x}_i)$ and all prototypes $\mathbf{p} \in P$ as

$$\mathbf{h}_i = \left(g(Z_{\mathbf{x}_i}, \mathbf{p}) = \max_{\mathbf{z} \in Z_{\mathbf{x}_i}} \log \left(\frac{\| \mathbf{z}_i - \mathbf{p} \|^2 + 1}{\| \mathbf{z}_i - \mathbf{p} \|^2 + \varepsilon} \right) \right)_{\mathbf{p} \in P} \quad \text{for } \varepsilon > 0.$$

As a results, we have a bag of similarity scores $\{\mathbf{h}_1, \ldots, \mathbf{h}_k\}$, passed to the attention pooling [31] to obtain a single similarity scores for the entire bag

$$\mathbf{h}_{bag} = \sum_{i=1}^{k} a_i \, \mathbf{h}_i, \quad \text{where } a_i = \frac{\exp\{\mathbf{w}^T (\tanh(\mathbf{V}\,\mathbf{h}_i^T) \odot \mathrm{sigm}(\mathbf{U}\,\mathbf{h}_i^T)\}}{\sum_{j=1}^{k} \exp\{\mathbf{w}^T (\tanh(\mathbf{V}\,\mathbf{h}_j^T) \odot \mathrm{sigm}(\mathbf{U}\,\mathbf{h}_j^T)\}}, \tag{3.15}$$

$\mathbf{w} \in \mathbb{R}^{L \times 1}$, $\mathbf{V} \in \mathbb{R}^{L \times M}$, and $\mathbf{U} \in \mathbb{R}^{L \times M}$ are parameters, $\tanh$ is the hyperbolic tangent, sigm is the sigmoid non-linearity and $\odot$ is an element-wise multiplication. Such representation is then sent to the last layer to obtain the predicted label $\check{y} = g(\mathbf{h_{bag}})$ as in [11].

Results. We evaluated the effectiveness of ProtoMIL on multiple datasets, but in this section, we present the results on three large-scale datasets: Camelyon16 [18], TCGA-NSCLC [6], and TCGA-RCC [2]. Table 3.5 summarizes the results. ProtoMIL achieved superior results in terms of AUC for the Camelyon dataset, while the results for the other datasets are comparable. It is worth noting that interpretable methods often trade-off interpretability for effectiveness, which can lead to a slight drop in accuracy [58]. We also marked the level of interpretability given by the models, only ProtoMIL can explain its decisionwith global and local interpretations as it is presented in Figure 2.3.

Summary. Our work [V] extends the use of prototypical parts to the multiple instance learning (MIL) problem. By doing so, we are able to not only obtain local interpretations for individual predictions but also effectively characterize the classes using prototypical parts. This provides the user with the ability to assess the trustworthiness of the model's predictions and facilitating making informed decisions.

3.2.4. Summary

In this chapter, we summarized the contributions made by the Ph.D. candidate towards advancing the field of interpretable deep learning models for Multiple-Instance Learning problems, specifically through the adoption of self-attention mechanisms and prototypical parts. Moving forward, we now discuss the other research-related activities undertaken by the candidate that are not directly related to the publication included in this dissertation.

4. Ph.D. candidate's research profile

This chapter of the dissertation provides a comprehensive account of the Ph.D. candidate's other notable accomplishments outside the scope of the publications that comprise this thesis. Specifically, we discuss the candidate's contributions to other publications, grants obtained, and participation in relevant projects. Additionally, we examine the candidate's collaborations with other researchers. Finally, we highlight the candidate's work as a reviewer and efforts to promote science.

List of publications to which Ph.D. candidate contributed that are not part of this dissertation:

- Adriana Borowa, **Dawid Rymarczyk**, Dorota Ochońska, Agnieszka Sroka-Oleksiak, Monika Brzychczy-Włoch, Bartosz Zieliński, "Identifying bacteria species on microscopic polyculture images using deep learning", IEEE Journal of Biomedical and Health Informatics, 2023, IF 7.021, MEiN points 140.
- Bartosz Zieliński, Agnieszka Sroka-Oleksiak, **Dawid Rymarczyk**, Adam Piekarczyk, Monika Brzychczy-Włoch, "Deep learning approach to describe and classify fungi microscopic images", PLoS ONE, 2020, IF 3.240, MEiN points 100.
- Adriana Borowa, **Dawid Rymarczyk**, Dorota Ochońska, Monika Brzychczy-Włoch, Bartosz Zieliński, "Deep learning classification of bacteria clones explained by persistence homology", International Joint Conference on Neural Networks (IJCNN), 2021, CORE B, MEiN points 140.
- Mikołaj Sacha, **Dawid Rymarczyk**, Łukasz Struski, Jacek Tabor, Bartosz Zieliński, "ProtoSeg: Interpretable Semantic Segmentation With Prototypical Parts", IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023, CORE A, MEiN points 140.
- Adam Pardyl, **Dawid Rymarczyk**, Zbisław Tabor, Bartosz Zielinski. "Automating Patient-Level Lung Cancer Diagnosis in Different Data Regimes." Neural Information Processing (ICONIP). 2022. CORE B. MEiN points 140.
- **Dawid, Rymarczyk**, Adriana, Borowa, Anna, Bracha, Maurycy, Chronowski, Wojciech, Ozimek, Bartosz, Zieliński. "Comparison of supervised and self-supervised deep representations trained on histological images." One World, One Health – Global Partnership for Digital Innovation (MedInfo), 2021, rank CORE B, MEiN points 70.

List of acquired grants (as principal investigator):

- NCN Preludium: Improving interpretability properties of prototypical parts-based deep neural networks.
  National Science Center, Poland
  2023 - 2024
- POB DigiWorld minigrant: Deep Neural Networks in the context of human cognitive system.
  Jagiellonian University, Poland
  2021 - 2022

List of grants to which Ph.D. candidate contributed (as co-investigator):

- Team-Net: Bio-inspired artificial neural networks
  Foundation for Polish Science, Poland
  principal investigator: prof. Jacek Tabor,
  2020 - 2023
- NCN Opus: Deep Self-Organizing Neural Graphs
  National Science Center, Poland
  principal investigator: prof. Jacek Tabor,
  2022 - 2025
- POB DigiWorld minigrant: Innovative microbiological diagnostics using machine learning
  Jagiellonian University, Poland
  principal investigator: dr Bartosz Zieliński,
  2021
- Fast track: Opracowanie platformy badań fenotypowych, opartej na technologii high-content screening, z analizą za pomocą algorytmów sztucznej inteligencji w celu odkrywania nowych leków w chorobach neurozapalnych i zwłóknieniowych
  National Center for Research and Development, Poland
  company: Selvita,
  2019 - 2022
- Fast track: X-rAI: Przeglądarka diagnostyczna dla radiologii z komputerowym wspomaganiem wykorzystującym Sztuczną Inteligencję
  National Center for Research and Development, Poland
  company: AGH University of Science and Technology,
  2021 - 2023

Established research cooperations.

- International cooperation:
  - Internship in the group of prof. Joost van de Weijer at Computer Vision Center at Autonomous University of Barcelona. Outcomes:
    - **Dawid Rymarczyk**, Joost van de Weijer, Bartosz Zieliński, Bartłomiej Twardowski. ICICLE: Interpretable Class Incremental Continual Learning. arXiv preprint arXiv:2303.07811, 2023.
    - Sumbission of a grant proposal for European Commission call HORIZON-CL4-2023-HUMAN-01-CNECT with a topic HORIZON-CL4-2023-HUMAN-01-01. The consortium consists of 18 partners worldwide, among others there are University of Oxford, and University of Piza.
- National cooperation:
  - Cooperation with prof. Zbisław Tabor from AGH University of Science and Technology in Kraków. Outcomes:
    - Adam Pardyl, **Dawid Rymarczyk**, Zbisław Tabor, Bartosz Zieliński. Automating Patient-Level Lung Cancer Diagnosis in Different Data Regimes. Neural Information Processing (ICONIP). 2022. CORE B. MNiSW points 140.
  - Cooperation with prof. Monika Brzychczy-Włoch from Colegium Medicum of Jagiellonian University. Outcomes:
    - Adriana Borowa, **Dawid Rymarczyk**, Dorota Ochońska, Agnieszka Sroka-Oleksiak, Monika Brzychczy-Włoch, Bartosz Zieliński, Identifying bacteria species on microscopic polyculture images using deep learning, IEEE Journal of Biomedical and Health Informatics, 2023, IF 7.021, MNiSW points 140.
    - Bartosz Zieliński, Agnieszka Sroka-Oleksiak, **Dawid Rymarczyk**, Adam Piekarczyk, Monika Brzychczy-Włoch, Deep learning approach to describe and classify fungi microscopic images, PLoS ONE, 2020, IF 3.240, MNiSW points 100.
    - Adriana Borowa, **Dawid Rymarczyk**, Dorota Ochońska, Monika Brzychczy-Włoch, Bartosz Zieliński, Deep learning classification of bacteria clones explained by persistence homology, International Joint Conference on Neural Networks (IJCNN), 2021, CORE B, MNiSW points 140.
    - Submission to European Patent Office of the following invention: "A method and a system for identifying polyculture bacteria on microscopic images using deep learning", Application number: EP22461550.0. 2022.
  - Cooperation with dr Koryna Lewandowska from Jagiellonian University. Outcomes:
    - **Dawid Rymarczyk**, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, Bartosz Zieliński. "Interpretable image classification with

differentiable prototypes assignment.“ European Conference on Computer Vision (ECCV), 2022, pp. 351-368, rank CORE A*, 140 MEiN points.

Acting as a reviewer for:

- The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), CORE rank A*,
- International Conference on Computer Vision (ICCV), CORE rank A*,
- European Conference on Computer Vision (ECCV), CORE rank A*,
- IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), CORE rank A,
- European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), CORE rank A.

Science promotion:

- Co-organizer of a workshop accompanying European Conference on Artificial Intelligence (ECAI, CORE rank A) on “Joint workshops on XAI methods, challenges and applications“, 2023,
- invited speaker for TestDive 2020,
- scientific poster presentation:
  Michał Koziarski, Piotr Gaiński, Krzysztof Rataj, Adriana Borowa, Konrad Wójtowicz, Jakub Gwóźdź, Magdalena Otrocka, **Dawid Rymarczyk**, Michał Warchoł, Multimodal Approach to MoA Prediction Based on Cell Painting Imaging and Chemical Structure Data, ELRIG Annual Drug Discovery Conference, 2022.

5. Conclusions

This dissertation highlights notable accomplishments of a Ph.D. candidate in the field of interpretable deep learning, particularly in the areas of prototypical parts and self-explainable Multiple-Instance Learning. The candidate's research has been accepted to the main tracks of prestigious conferences, including ECCV and KDD, as well as other reputable international events such as WACV, SDM, and ECML PKDD. In these works, new self-explainable deep learning approaches were introduced, such as ProtoPShare and ProtoPool sharing prototypical parts between classes, ProGReST tackling molecular property prediction problem in an interpretable way, and advancing field of Multiple Instance Learning with SA-AbMILP and ProtoMIL, focusing on modeling relationships between instances in a bag and providing global explanations. Moreover, he has acted as a reviewer for top-tier conferences such as CVPR, and he has actively promoted science, e.g. by organizing workshops at conferences.

The Ph.D. candidate's achievements go beyond his high-quality research work, as he has also secured research funding from a Preludium grant from NCN and Excellence Initiative from Jagiellonian University. Moreover, he established international collaborations resulting in a scientific internship in the group of prof. Joost van der Weijer at Computer Vision Center at Autonomous University of Barcelona, and a submission of a grant proposal to the European Commission with 18 international partners. He also contributed to multiple interdisciplinary national collaborations, e.g. with microbiologists. Additionally, he has shown commercialization potential of his work by submitting a patent application to the European Patent Office.

His previous works have inspired others in the field, such as ProtoPShare [I] which served as a basis for the development of data distillation techniques for interpretable models [36], and ProtoPool [II] which has led to the generalization of prototypical parts to ProtoKNN [68]. In addition, the MIL methods developed in the work [IV] have been used to develop more effective MIL classifiers [42], and the concept of ProtoMIL [V] has been further explored by [78] to combine prototypes with contrastive learning to improve the model's performance.

Looking ahead, the Ph.D. candidate plans to collaborate with international partners to develop sustainable and interpretable neural networks, continue exploring multiple-instance learning techniques for pathogen classification with microbiologists. Currently, he is involved in works on enhancing the robustness of prototypical parts against adversarial attacks and generating counterfactual examples by altering the prototypical parts.

Information on publications constituting part of collection of research articles

[I] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, Bartosz Zieliński. “Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification.“ Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021, pp. 1420-1430, rank CORE A*, 200 MEiN points.

[II] Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, Bartosz Zieliński. “Interpretable image classification with differentiable prototypes assignment.“ European Conference on Computer Vision (ECCV), 2022, pp. 351-368, rank CORE A*, 140 MEiN points.

[III] Dawid Rymarczyk, Daniel Dobrowolski, Tomasz Danel. “ProGReST: Prototypical Graph Regression Soft Trees for Molecular Property Prediction.“ SIAM Conference on Data Mining (SDM), 2023, pp. 379-387, rank CORE A, 140 MEiN points.

[IV] Dawid Rymarczyk , Adriana Borowa, Jacek Tabor, Bartosz Zieliński. “Kernel self-attention for weakly-supervised image classification using deep multiple instance learning.“ IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2021, pp. 1721-1730, rank CORE A,140 MEiN points.

[V] Dawid Rymarczyk, Adam Pardyl, Jarosław Kraus, Aneta Kaczyńska, Marek Skomorowski, Bartosz Zieliński. “ProtoMIL: Multiple Instance Learning with Prototypical Parts for Whole-Slide Image Classification.“ European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), 2022, rank CORE A, 140 MEiN points.

Bibliography

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

[2] O Akin, P Elnajjar, M Heller, R Jarosz, B Erickson, S Kirk, and J Filippini. Radiology data from the cancer genome atlas kidney renal clear cell carcinoma [tcga-kirc] collection. *The Cancer Imaging Archive*, 2016.

[3] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018.

[4] Vijay Arya, Rachel KE Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C Hoffman, Stephanie Houde, Q Vera Liao, Ronny Luss, Aleksandra Mojsilović, et al. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques. *arXiv:1909.03012*, 2019.

[5] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[6] Shaimaa Bakr, Olivier Gevaert, Sebastian Echegaray, Kelsey Ayers, Mu Zhou, Majid Shafiq, Hong Zheng, Jalen Anthony Benson, Weiruo Zhang, Ann NC Leung, et al. A radiogenomic dataset of non-small cell lung cancer. *Scientific data*, 5(1):1–9, 2018.

[7] Alina Jade Barnett, Fides Regina Schwartz, Chaofan Tao, Chaofan Chen, Yinhao Ren, Joseph Y Lo, and Cynthia Rudin. Iaia-bl: A case-based interpretable deep learning model for classification of mass lesions in digital mammography. *arXiv preprint arXiv:2103.12308*, 2021.

[8] Suman K. Bera, Jason Dent, Gill Gurbinder, Andrew Stoleman, and Bo Wu. Simgcn for tdc benchmarks. 2022.

[9] Adriana Borowa, Dawid Rymarczyk, Dorota Ochońska, Monika Brzychczy-Włoch, and Bartosz Zieliński. Classifying bacteria clones using attention-based deep multiple instance learning interpreted by persistence homology. In *International joint conference on neural networks*, 2021.

[10] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *arXiv preprint arXiv:1904.00760*, 2019.

[11] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

[12] Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018.

[13] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

[14] Li Deng and Yang Liu. *Deep learning in natural language processing*. Springer, 2018.

[15] 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.

[16] Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable protopnet: An interpretable image classifier using deformable prototypes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10265–10275, 2022.

[17] Finale Doshi-Velez and Been Kim. A roadmap for a rigorous science of interpretability. *arXiv:1702.08608*, 2, 2017.

[18] Babak Ehteshami Bejnordi, Mitko Veta, Paul Johannes van Diest, Bram van Ginneken, Nico Karssemeijer, Geert Litjens, Jeroen A. W. M. van der Laak, , and the CAMELYON16 Consortium. Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer. *JAMA*, 318(22):2199–2210, 12 2017.

[19] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2950–2958, 2019.

[20] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017.

[21] James Foulds and Eibe Frank. A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(1):1–25, 2010.

[22] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv*, 2017.

[23] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4438–4446, 2017.

[24] Erik Gawehn, Jan A Hiss, and Gisbert Schneider. Deep learning in drug discovery. *Molecular informatics*, 35(1):3–14, 2016.

[25] Elisa Drelie Gelasca, Jiyun Byun, Boguslaw Obara, and BS Manjunath. Evaluation and benchmark for biological image segmentation. In *2008 15th IEEE International Conference on Image Processing*, pages 1816–1819. IEEE, 2008.

[26] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[28] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. Interpretable image recognition with hierarchical prototypes. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7, pages 32–40, 2019.

[29] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv*, 2019.

[30] Kexin Huang, Tianfan Fu, Lucas M Glass, Marinka Zitnik, Cao Xiao, and Jimeng Sun. Deeppurpose: a deep learning library for drug–target interaction prediction. *Bioinformatics*, 36(22-23):5545–5547, 2020.

[31] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Attention-based deep multiple instance learning. *arXiv preprint arXiv:1802.04712*, 2018.

[32] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv:1611.01144*, 2016.

[33] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani Srivastava. How can i explain this to you? an empirical study of deep neural network explanation methods. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4211–4222. Curran Associates, Inc., 2020.

[34] Margot E Kaminski. The right to explanation, explained. In *Research Handbook on Information Law and Governance*, pages 278–299. Edward Elgar Publishing, 2021.

[35] Raymond Kesner. A neural system analysis of memory storage and retrieval. *Psychological Bulletin*, 80(3):177, 1973.

[36] Monish Keswani, Sriranjani Ramakrishnan, Nishant Reddy, and Vineeth N Balasubramanian. Proto2proto: Can you recognize the car, the way i do? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10233–10243, 2022.

[37] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[38] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv*, 2016.

[39] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR, 2020.

[40] Janet Kolodner. *Case-based reasoning*. Morgan Kaufmann, 2014.

[41] Wei-Han Lee, Steve Millman, Nirmit Desai, et al. Neuralfp: Out-of-distribution detection using fingerprints of neural networks. In *ICPR*, 2021.

[42] Bin Li, Yin Li, and Kevin W Eliceiri. Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2021.

[43] Guangli Li, Chuanxiu Li, Guangting Wu, Donghong Ji, and Hongbing Zhang. Multi-view attention-guided multiple instance detection network for interpretable breast cancer histopathological image diagnosis. *IEEE Access*, 2021.

[44] Jiayun Li, Wenyuan Li, Arkadiusz Gertych, Beatrice S Knudsen, William Speier, and Corey W Arnold. An attention-based multi-resolution model for prostate whole slide imageclassification and localization. *arXiv preprint arXiv:1905.13208*, 2019.

[45] Katherine Li, Richard Strauss, Colleen Marano, Linda E Greenbaum, Joshua R Friedman, Laurent Peyrin-Biroulet, Carrie Brodmerkel, and Gert De Hertogh. A simplified definition of histologic

improvement in ulcerative colitis and its association with disease outcomes up to 30 weeks from initiation of therapy: Post hoc analysis of three clinical trials. *Journal of Crohn's and Colitis*, 13(8):1025–1035, 2019.

[46] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[47] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[48] Ming Y Lu, Richard J Chen, Jingwen Wang, Debora Dillon, and Faisal Mahmood. Semi-supervised histology classification using deep multiple instance learning and contrastive predictive coding. *arXiv preprint arXiv:1910.10825*, 2019.

[49] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[50] Emanuele Marconato, Andrea Passerini, and Stefano Teso. Glancenets: Interpretabile, leak-proof concept-based models. *arXiv preprint arXiv:2205.15612*, 2022.

[51] Łukasz Maziarka, Dawid Majchrowski, Tomasz Danel, Piotr Gaiński, Jacek Tabor, Igor Podolak, Paweł Morkisz, and Stanisław Jastrzębski. Relative molecule self-attention transformer. *arXiv*, 2021.

[52] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[53] Meike Nauta et al. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14933–14943, 2021.

[54] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[55] Sylvestre-Alvise Rebuffi, Ruth Fong, Xu Ji, and Andrea Vedaldi. There and back again: Revisiting backpropagation saliency methods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8839–8848, 2020.

[56] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[57] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[58] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

[59] Dawid Rymarczyk, Adriana Borowa, Jacek Tabor, and Bartosz Zielinski. Kernel self-attention for weakly-supervised image classification using deep multiple instance learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1721–1730, 2021.

[60] Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. Interpretable image classification with differentiable prototypes assignment. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[61] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1420–1430, 2021.

[62] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[63] Ramprasaath R Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry Heck, Dhruv Batra, and Devi Parikh. Taking a hint: Leveraging explanations to make vision and language models more grounded. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2591–2600, 2019.

[64] Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, and Yongbing Zhang. Transmil: Transformer based correlated multiple instance learning for whole slide image classication. *arXiv preprint arXiv:2106.00908*, 2021.

[65] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[66] 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.

[67] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[68] Yuki Ukai, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. This looks like it rather than that: Protoknn for similarity-based classifiers. In *The Eleventh International Conference on Learning Representations*, 2023.

[69] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, Eftychios Protopapadakis, et al. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[70] Caroline Wang, Bin Han, Bhrij Patel, and Cynthia Rudin. In pursuit of interpretable, fair and accurate machine learning for criminal recidivism prediction. *Journal of Quantitative Criminology*, pages 1–63, 2022.

[71] Jiaqi Wang et al. Interpretable image recognition by constructing transparent embedding space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 895–904, 2021.

[72] Shujun Wang, Yaxi Zhu, Lequan Yu, Hao Chen, Huangjing Lin, Xiangbo Wan, Xinjuan Fan, and Pheng-Ann Heng. Rmdl: Recalibrated multi-instance deep learning for whole slide gastric image classification. *Medical image analysis*, 58:101549, 2019.

[73] Tong Wang. Gaining free or low-cost interpretability with interpretable partial substitute. In *ICML*, pages 6505–6514. PMLR, 2019.

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

[75] Zhaoping Xiong et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of Medicinal Chemistry*, 2020.

[76] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020.

[77] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[78] Jin-Gang Yu, Zihao Wu, Yu Ming, Shule Deng, Yuanqing Li, Caifeng Ou, Chunjiang He, Baiye Wang, Pusheng Zhang, and Yu Wang. Prototypical multiple instance learning for predicting lymph node metastasis of breast cancer from whole-slide pathological images. *Medical Image Analysis*, page 102748, 2023.

[79] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

[80] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9127–9135, 2022.

[81] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018.

[82] 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*, pages 1249–1256, 2009.

ProtoPShare: Prototypical Parts Sharing for Similarity Discovery in Interpretable Image Classification

Dawid Rymarczyk
dawid.rymarczyk@student.uj.edu.pl
Faculty of Mathematics and Computer Science,
Jagiellonian University and Ardigen SA
Cracow, Poland

Łukasz Struski
lukasz.struski@uj.edu.pl
Faculty of Mathematics and Computer Science,
Jagiellonian University
Cracow, Poland

Jacek Tabor
jacek.tabor@uj.edu.pl
Faculty of Mathematics and Computer Science,
Jagiellonian University
Cracow, Poland

Bartosz Zieliński
bartosz.zielinski@uj.edu.pl
Faculty of Mathematics and Computer Science,
Jagiellonian University and Ardigen SA
Cracow, Poland

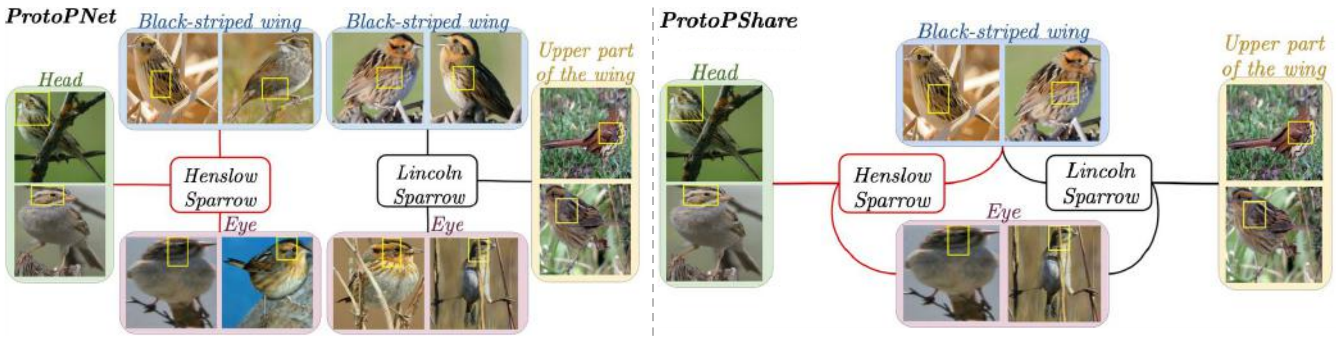


Figure 1: Comparison between ProtoPNet with exclusive prototypes and our ProtoPShare that shares prototypes between classes. The number of prototypes in ProtoPNet is large because each of them is assigned to only one class. E.g. in this example, classes *Henslow sparrow* and *Lincoln sparrow* have separate prototypes corresponding to *Eye* in ProtoPNet. In ProtoPShare, the similar prototypes are merged into one, decreasing model complexity and discovering similarities between classes.

ABSTRACT

In this work, we introduce an extension to ProtoPNet called ProtoPShare which shares prototypical parts between classes. To obtain prototype sharing we prune prototypical parts using a novel data-dependent similarity. Our approach substantially reduces the number of prototypes needed to preserve baseline accuracy and finds prototypical similarities between classes. We show the effectiveness of ProtoPShare on the CUB-200-2011 and the Stanford Cars datasets and confirm the semantic consistency of its prototypical parts in user-study.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; **Neural networks**.

KEYWORDS

prototypical parts; interpretability; explainability; neural networks

ACM Reference Format:
Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. 2021. ProtoPShare: Prototypical Parts Sharing for Similarity Discovery in Interpretable Image Classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467245

1 INTRODUCTION

Broad application of deep learning in domains like medical diagnosis and autonomous systems enforces models to explain their decisions. Ergo, more and more methods provide human-understandable justifications for their output [2–5, 8, 10, 13, 30]. Some of them are inspired by the human brain and how it explains its visual judgments by pointing to prototypical features that an object possesses [32].

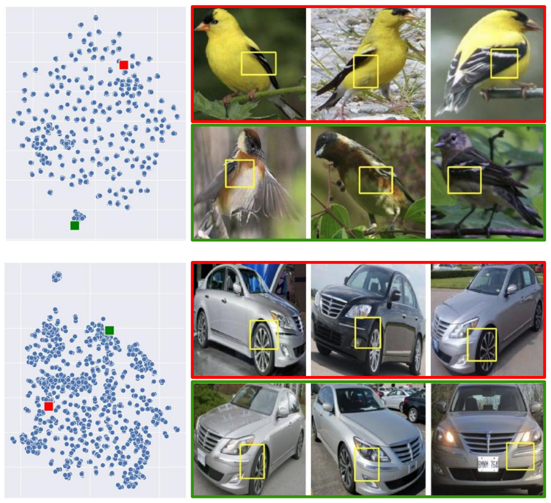


Figure 2: Example prototypes similar according to our data-dependent similarity but distant in representation space. The tSNE projections of prototypes (left plots) trained for CUB-200-2011 and Stanford Cars datasets (top and bottom part, respectively) contains two prototypes marked as red/green squares. They correspond to semantically similar prototypical parts (top representing *bright belly with grayish wings* and bottom representing *fender*) that are distant in the representation space. Such pairs can be discovered by our data-dependent similarity defined in Section 3. Notice that each example prototype is represented by one row of 3 closest images' parts (marked with yellow bounding-boxes). More examples are presented in Figure 11.

I.e., a certain object is a car because it has tires, roof, headlights, and horn.

Recently introduced *Prototypical Part Network* (*ProtoPNet*) [4] applies this paradigm by focusing on parts of an image and comparing them with *prototypical parts* of a given class. This comparison is achieved by pushing parts of an image and prototypical parts through the convolutional layers, obtaining their representation, and computing similarity between them. We will refer to the representations of prototypical parts as *prototypes*.

ProtoPNet is a self-explaining model that generates intuitive explanations and achieves accuracy comparable with its analogous non-interpretable counterparts. However, it has limited applicability due to two main reasons. Firstly, the number of prototypes is large because each of them is assigned to only one class. It negatively influences the interpretability whose much-desirable properties are small size and low complexity [7, 39]. Secondly, due to the training that pushes away the prototypes of different classes in the representation space, the prototypes with similar semantic can be distant (see Figure 2). Therefore, the predictions obtained from the network can be unstable.

In this work, we address those limitations by introducing the *Prototypical Part Shared* network (*ProtoPShare*)[1] that shares the prototypes between the classes, like presented in Figure 1. As a

[1]The code is available here: https://github.com/gmum/ProtoPShare.

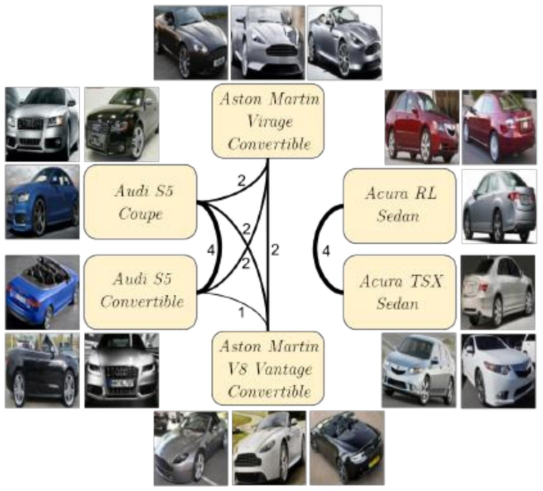


Figure 3: Inter class similarity visualization as a graph generated based on the prototypes shared in ProtoPShare trained on Stanford Cars dataset. Each node corresponds to a class, and the strength of the edge between two nodes corresponds to the number of shared prototypes. E.g. *Audi S5 Coupe* shares 4 prototypes with *Audi S5 Convertible* but does not share prototypes with *Acura RL Sedan*. This graph can be used to find similarities between the classes or to cluster them into groups. Notice that each class is represented by three images located around the node.

result, the number of prototypes is relatively small and prototypes with similar semantic are close to each other. Additionally, it is possible to discover similarities between the classes, as presented in Figure 3. ProtoPShare method consists of two phases, initial training and prototypes' pruning. In the first phase, the model is trained using exclusive prototypes and a loss function described in [4]. In the second phase, the iterative pruning merges successive portions of the most similar prototypes. For this purpose, we introduce the data-dependent similarity described in Section 3 that finds prototypes with similar semantics, even if they are distant in representation space. We demonstrate the superiority of the ProtoPShare approach by comparing it to the other methods based on the prototypes' paradigm. Our contribution can be therefore summarized as follows:

- We construct ProtoPShare, a self-explained method built on the paradigm of prototypical parts that shares prototypes between the classes.
- We introduce data-dependent similarity that can find semantically similar prototypes even if they are distant in the representation space.
- We significantly reduce the number of prototypes comparing to state of the art, reducing model complexity and discovering similarities between classes.

This paper is organized as follows. Section 2 investigates related works on interpretability and pruning. In Section 3, we introduce our ProtoPShare method together with its theoretical understanding. Section 4 illustrates experimental results on CUB-200-2011 and

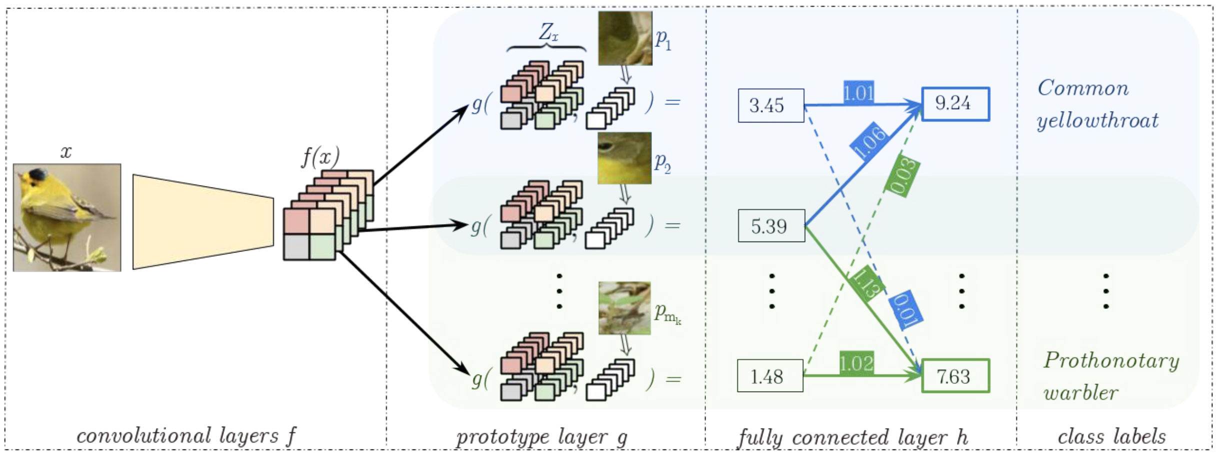


Figure 4: The architecture of our ProtoPShare. It consists of convolutional layers f, followed by a prototype layer g, and a fully connected layer h. Layer g contains prototypes that are shared between the classes. E.g. in this example, prototype p_2 is assigned both to class *Common yellowthroat* and *Prothonotary warbler* which corresponds to stronger connections between p_2 and both classes in layer h. As a result of sharing the prototypes, we reduce their number and discover similarities between classes.

Stanford Cars datasets, while Section 5 concentrates on ProtoPShare interpretability. Finally, we conclude the work in Section 6.

2 RELATED WORKS

ProtoPShare is a self-explained method with a strong focus on the prototypes' pruning. Therefore, in the related works, we consider the articles about interpretability and pruning.

Interpretability. Interpretability approaches can be divided into post hoc and self-explaining methods [2]. *Post hoc* techniques include saliency maps, showing how important each pixel of a particular image is for its classification [30, 33–35]. Another technique called concept activation vectors provides an interpretation of the internal network state in terms of human-friendly concepts [5, 12, 19, 42]. Other methods analyze how the network's response changes for perturbed images [8, 9, 31]. Post hoc methods are easy to use in practice as they do not require changes in the architecture. However, they can produce unfaithful and fragile explanations [1]. Therefore, as a remedy, *self-explainable* models were introduced by applying the attention mechanism [10, 36, 40, 44, 45] or the bag of local features [3]. Other works [13, 23, 29] focus on exploiting the latent feature space obtained, e.g. with adversarial autoencoder. The interesting self-explaining methods are prototype-based models represented, like Prototypical Part Network [4] with a hidden layer of prototypes representing the activation patterns. A similar approach for hierarchically organized prototypes is presented in [14] to classify objects at every level of a predefined taxonomy. Some works concentrate on transforming prototypes from the latent space to data space [14]. The others try to apply prototypes to other domains like sequence learning [27] or time series analysis [11].

Pruning. Pruning is mostly used to accelerate deep neural networks by removing unnecessary weights or filters [25]. The latter can be roughly divided into data-dependent and data-independent filter pruning [16], depending on whether the data is used or not to determine the pruned filters. The basic type of *data-independent* pruning removes filters with the smallest sum of absolute values of weights [22]. More complicated approaches designate irrelevant or redundant filters using scaling parameters of batch normalization layers [41] or the concept of geometric median [16]. *Data-dependent* pruning can be solved as an optimization problem on the statistics computed from its subsequent layer [26], by minimizing the reconstruction error on the subsequent feature map using Lasso method [17], or by using a criterion based on Taylor expansion that approximates the change in the cost function induced by pruning [28]. Filters' importance can also be propagated from the final response layer to minimize its reconstruction error [43]. Another possibility is to introduce additional discrimination-aware losses and select the most discriminative channels for each layer [46]. Finally, feature maps can be clustered and replaced by the average representative from each cluster [38]. Moreover, filters can be pruned together with other structures of the network [24].

Our ProtoPShare extends the ProtoPNet [4] by the shared prototypes obtained with the data-dependent pruning based on the feature maps.

3 PROTOPSHARE

In this section, we first describe the architecture of ProtoPShare and then define our data-dependent merge-pruning algorithm. Finally, we provide a theoretical understanding of how the prototypes' merge affects classification accuracy.

Architecture. The architecture of ProtoPShare, shown in Figure 4, consists of convolutional layers f, followed by a prototype layer g, and a fully connected layer h (with weight w_h and no bias). Given an input image $x \in X$, the convolutional layers extract image representation $f(x)$ of shape $H \times W \times D$. For the clarity of description, let $Z_x = \{z_i \in f(x) : z_i \in \mathbb{R}^D, i = 1..HW\}$. Then, for

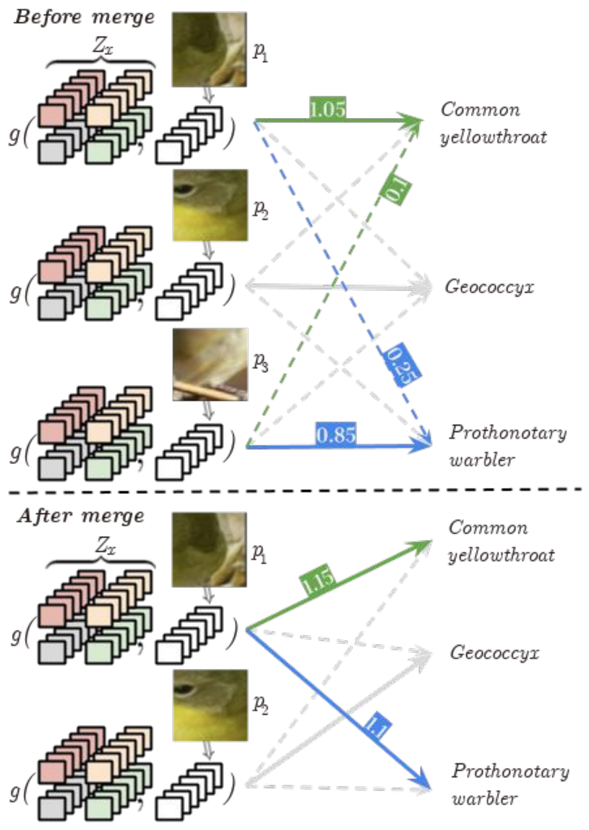


Figure 5: The process of merging two prototypes. If prototypes p_1 and p_3 are similar (e.g. they both present a *birds' leg*), we delete one of them (in this case p_3). Moreover, to recompensate this removal, we assign p_1 to p_3's class and add p_3's weights to p_1's weights in h (here done by summing weights represented as the blue and green arrows).

each class k, the network learns m_k prototypes $P_k = \{p_i\}_{i=1}^{m_k}$, where $p_i \in \mathbb{R}^D$ represents prototypical parts trained for class k. Given a convolutional output Z_x and prototype p, ProtoPShare computes the distances between them, inverts them to obtain the similarity scores, and takes the maximum among all $z \in Z_x$:

$$g(Z_x, p) = \max_{z \in Z_x} \log \left(\frac{\|z-p\|^2+1}{\|z-p\|^2+\varepsilon} \right) \quad \text{for} \quad \varepsilon > 0. \tag{1}$$

Finally, the similarity scores produced by the prototype layer (m_k values per class) are multiplied by the weight matrix w_h in the fully connected layer h. This results in the output logits that are normalized using softmax to obtain a final prediction.

After training with exclusive prototypes and a loss function adapted from [4], most representations of the image's parts are clustered around semantically similar prototypes of their true classes, and the prototypes from different classes are well-separated. As a result, the prototypes with similar semantics can be distant in representation space (see Figure 2), resulting in unstable predictions. In the next paragraph, we present how to overcome this issue with our data-dependent merge-pruning.

Data-dependent merge-pruning. Here, we describe the pruning phase of our method. As the input, it obtains the network trained with exclusive prototypes, and as the output, it returns the network with a smaller number of shared prototypes.

Let P be the set of all prototypes (after training phase), K be the number of classes, and ζ be the percentage of prototypes to merge per pruning step. Each step begins with computing the similarities between the pairs of prototypes, then for each pair $(p, \tilde{p})$ among ζ percent of the most similar pairs, the prototype p is removed together with its weights $w_h(p)$. In exchange, the class to which p was assigned reuses prototype $\tilde{p}$ whose weights $w_h(\tilde{p})$ are aggregated with $w_h(p)$. We present this procedure in Figure 5. Note that in each step, we cannot always merge exactly ζ percent of prototypes. It is because the similarity metric can have the same value for many pairs. Lastly, we finetune layer h after each step.

As described in the architecture description, training with a loss function described in [4] can result in prototypes of similar semantic that are far from each other. To overcome this problem, we introduce a special type of data-dependent similarity. It uses training images $x \in X$ to generate the representation of all training patches, and compare the difference between $g(Z_x, p)$ and $g(Z_x, \tilde{p})$. The *data-dependent similarity* for pair of prototypes $(p, \tilde{p}) \in P^2$ is defined as the compliance on the similarity scores computed for all patches $x \in X$:

$$d_{DD}(p, \tilde{p}) = \frac{1}{\sum_{x \in X} (g(Z_x, p) - g(Z_x, \tilde{p}))^2}. \tag{2}$$

As a result, prototypes are considered similar if they activate alike on the training images, even if they are far from each other in the representation space.

Theoretical results. The following theorem provides some theoretical understanding of how the prototypes' merge affects the classification accuracy. Intuitively, the theorem states that if the pruning merges similar prototypes, then the predictions after merge do not change if predictions before the merge had certain confidence. Due to the page limits, the proof is in the Appendix A.

Theorem 1. *Let $x \in X$ be an input image correctly classified by ProtoPShare $h \circ g_p \circ f$ as class c before the prototypes' merge, and let:*

(A1) some of the class k prototypes remain unchanged $P'_k \subset P_k$ while the others merge into the other classes' prototypes $S_k \subset \bigcup_{i \neq k} P'_i$,

(A2) $p \in P_k \setminus P'_k$ is merged into $\tilde{p} \in S_k$,

(A3) $z_p = \arg\min_{z \in \bigcup_{x \in X} Z_x} \|z - p\|$ is the representation of the training patch nearest to any prototype p,

(A4) there exist $\delta \in (0, 1)$ such that:

a) for $k \neq c$, $p \in P_k \setminus P'_k$ and $\tilde{p} \in S_k$, we suppose that $\|p - \tilde{p}\| \leq \theta \|z_p - p\| - \sqrt{\varepsilon}$ and $\theta = \min(\sqrt{1+\delta} - 1, 1 - \frac{1}{\sqrt{2-\delta}})$ (ε comes from function g defined in (1)),

b) for $p \in P_c \setminus P'_c$ and $\tilde{p} \in S_c$, we suppose that $\|\tilde{p} - p\| \leq (\sqrt{1+\delta} - 1)\|z_p - p\|$ and $\|z_p - p\| \leq \sqrt{1-\delta}$,

(A5) for each class k, weights connecting class with assigned prototypes equal 1, and the other weights equal 0 (i.e., $w_h(p) = 1$ for $p \in P_k \cup S_k$ and $w_h(p) = 0$ for $p \in \bigcup_{i \neq k} P_i \setminus S_k$).

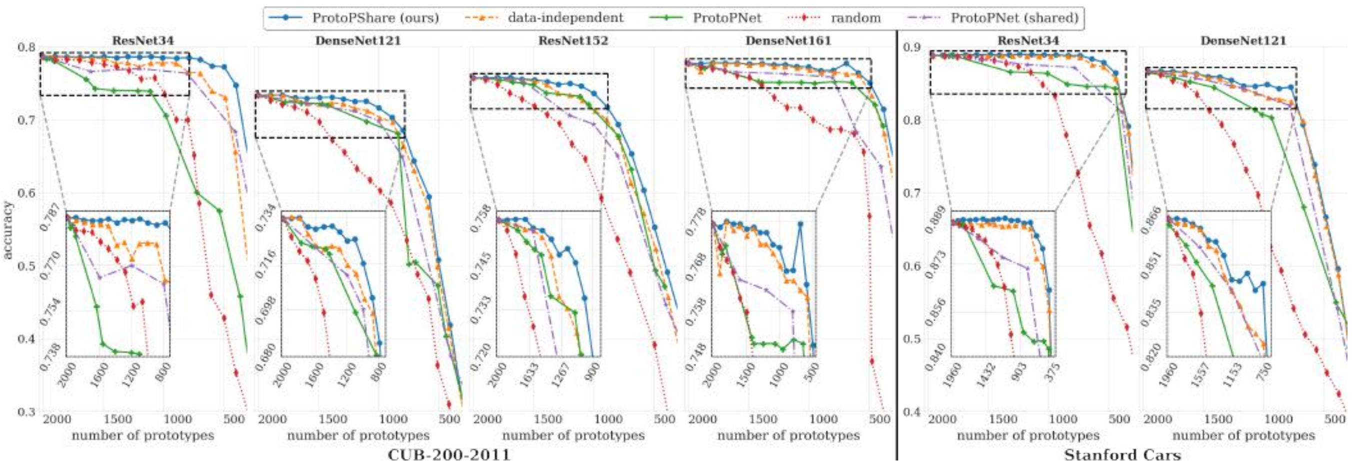


Figure 6: The accuracy of ProtoPShare and baseline methods with respect to decreasing number of prototypes. We compare the accuracy (higher is better) for various pruning rates on CUB-200-2011 and Stanford Cars datasets. As observed, our ProtoPShare achieves higher accuracy than the variations of our method (data-independent and random), ProtoPNet with the original pruning, and ProtoPNet (shared) trained with shared prototypes. Moreover, ProtoPShare maintains high accuracy even for only 25% of the initial prototypes. Notice that the accuracy for the initial pruning steps is zoomed to increase plot clarity, and that we stop pruning around 300 remaining prototypes due to the model collapse.

Then, after the prototypes' merge, the output logit for class c can decrease at most by $\Delta^c_{\max}$*, and the output logit for the other classes* $k \neq c$ *can increase at most by* $\Delta^k_{\max}$*, where:*

$$\Delta^k_{\max} := |P_k \setminus P'_k| \log\left((1+\delta)(2-\delta)\right) \quad \text{for} \;\; k \in \{1,\dots,K\}.$$

Hence, if the output logits between the top-2 classes are at least $\Delta^c_{\max} + \max_{k\neq c}\left(\Delta^k_{\max}\right)$ *apart, then the merge of prototypes does not change the prediction of* x.

4 EXPERIMENTS

In this section, we analyze the accuracy and robustness of ProtoPShare in various scenarios of the merge-pruning, and we explain our architectural choices by comparing it to other methods.

We train ProtoPShare to classify 200 bird species from the CUB-200-2011 dataset [37] and 196 car models from the Stanford Cars dataset [21]. As the convolutional layers f, we take the convolutional blocks of ResNet-34, ResNet-152 [15], DenseNet-121, and DenseNet-161 [18] pretrained on ImageNet [6]. We set the number of prototypes per class to 10, which results in 2000 prototypes for birds and 1960 prototypes for cars after the first phase of ProtoPShare. Moreover, in the second phase, we assume finetuning with 25 iterations after each pruning step.

How many prototypes are required? In Figure 6, we compare the accuracy of ProtoPShare with ProtoPNet and variations of our method (data-independent and random) for various pruning rates. Data-independent uses inverse Euclidean norm as the similarity measure instead of similarity defined in (2), while random corresponds to random joining. The results for ProtoPShare and its variations were obtained from the successive steps of pruning, while the results for ProtoPNet were obtained for different values of κ and τ from Appendix S8 in [4]. One can observe that ProtoPShare

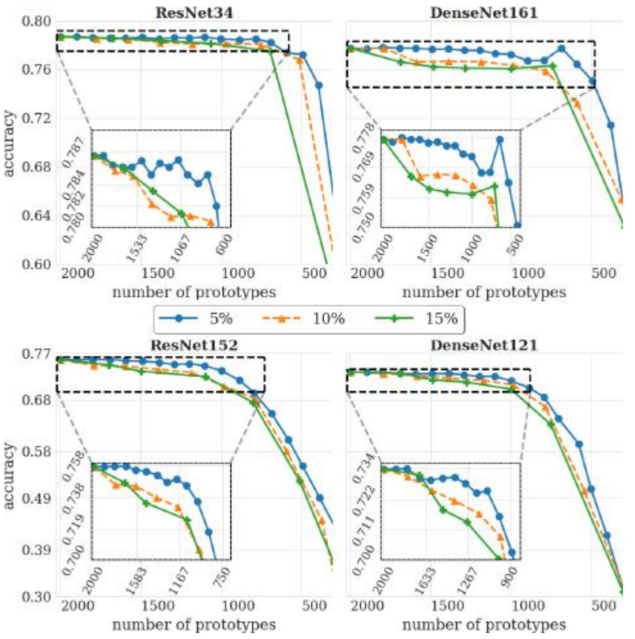


Figure 7: The accuracy of ProtoPShare for different percentages of prototypes merged per pruning step with respect to decreasing number of prototypes. Comparing the accuracy (higher is better) on the CUB-200-2011 dataset, we observe that the drop in accuracy is lower for a smaller number of prototypes (5%) merged per step. It is expected because small modifications in the set of prototypes make finetuning more effective than in the case of broader changes.

Table 1: ProtoPShare and baseline method results for the smallest number of prototypes that preserve baseline accuracy. Results show the superiority of our method achieved for various architectures trained on both datasets, which in some cases (e.g. ResNet34 trained on CUB-200-2011) obtains almost 10% higher accuracy. Additional results correspond to different finetuning strategies used after ProtoPShare pruning steps, including optimizing only the last layer (h), optimizing the prototypes and last layers (P and h), and optimizing all the layers (f, P, and h). Note that RN and DN correspond to ResNet and DenseNet architectures, respectively.

Model	Finetuning after pruning step f	P	h	$\|P\|$ before pruning	RN34 $\|P\|$ in the final model 400	RN152 1000	DN121 1000	DN161 600	$\|P\|$ before pruning	RN34 $\|P\|$ in the final model 480	DN121 980
				CUB-200-2011					Stanford Cars		
ProtoPNet (shared)	no pruning				0.6833	0.6938	0.6985	0.6823		0.8231	0.8264
ProtoPNet			✓	2000	0.5751	0.7207	0.6972	0.7509	1960	0.8428	0.8031
	✓	✓	✓	2000	0.6355	0.6821	0.6612	0.6526	1960	0.7864	0.7767
ProtoPShare (ours)		✓	✓	2000	0.6591	0.7161	0.6836	0.7047	1960	0.7902	0.7903
			✓	2000	**0.7472**	**0.7361**	**0.7472**	**0.7645**	1960	**0.8638**	**0.8481**

achieves higher accuracy for all pruning rates and works reasonably well even for only 25% of the initial prototypes in case of ResNet34. Moreover, data-independent obtains similarly good results at the initial steps of prunings, but its accuracy drops more rapidly with a higher pruning rate. At the same time, the results of ProtoPNet, either with the original pruning or with shared prototypes, are always worse than ProtoPShare. Moreover, results for those two ProtoPNet variations are between results for data-independent and random models. Finally, for each model, we observe a critical step of pruning with a significant accuracy drop. Hence, we suggest monitoring the accuracy of the validation set when applying ProtoPShare in specific domains.

Accuracy vs. step size. In Figure 7, we present how the ProtoPShare behaves for different percentages of prototypes merged per pruning step (ζ = 5%, 10%, or 15%). One can observe that the accuracy is higher for smaller ζ. It is expected because small modifications in the set of prototypes make finetuning with 25 iterations more effective than in the case of broader changes. Notice that we have not tested $\zeta < 5\%$, which would most probably further increase the accuracy at the cost of extensive computations.

Why two training phases? To demonstrate the need for two training phases in ProtoPShare, we compare it to ProtoPNet trained with shared prototypes. In this setting, ProtoPShare is first trained with plenty of prototypes (e.g. 2000 for the CUB-200-2011) and then pruned to a small number of prototypes (e.g. 400 for ResNet34). Simultaneously, we train ProtoPNet with a small number of prototypes from the beginning (e.g. 400 for ResNet34) that are shared between classes. The results presented in Figure 6 and Table 1 confirm that ProtoPShare achieves better accuracy for both datasets. The biggest gain in accuracy, between 5% to 10%, is obtained for the CUB-200-2011. However, the trend is similar in the case of the Stanford Cars. Additionally, we compare the effectiveness of different finetuning strategies in Table 1. Based on the results, we conclude that the finetuning only the last layer is the best strategy. Notice that the results in Table 1 are obtained for the smallest number of prototypes that preserve baseline accuracy, while in Figure 6 we plot the complete results.

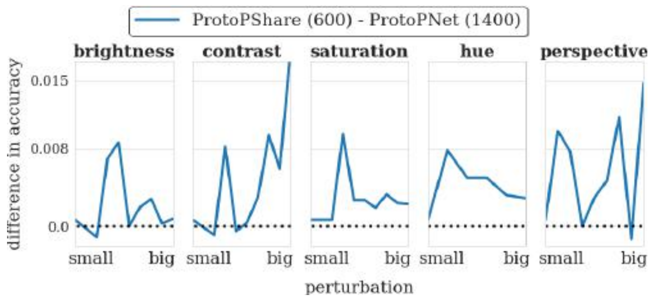


Figure 8: The difference in the accuracy between ProtoPShare with 600 prototypes and ProtoPNet with 1400 prototypes. The results obtained for DenseNet161 and various image perturbations suggest that ProtoPShare preserves robustness to the image perturbations even though it uses a smaller number of prototypes.

Resistance to perturbations. In Figure 8 we present the difference in accuracy between ProtoPShare trained for 2000 and then pruned to 600 prototypes with ProtoPNet trained for 2000 and then pruned to 1400 prototypes (using $\kappa = 6$ and $\tau = 5$ from Appendix S8 in [4]) for various image perturbations. We incorporate different perturbations and magnitudes that modify brightness, contrast, saturation, hue, and perspective using *Torchvision ColorJitter* and *Perspective* transformations with probability of perturbation equal 1 and perturbation values from range $[0; 0.5]$ for hue and $[0; 1]$ for the others. As presented, the ProtoPShare with smaller number of prototypes performs on par to the ProtoPNet with a larger number of prototypes. Therefore, we conclude that ProtoPShare preserves robustness to the image perturbations even though it uses a smaller number of prototypes.

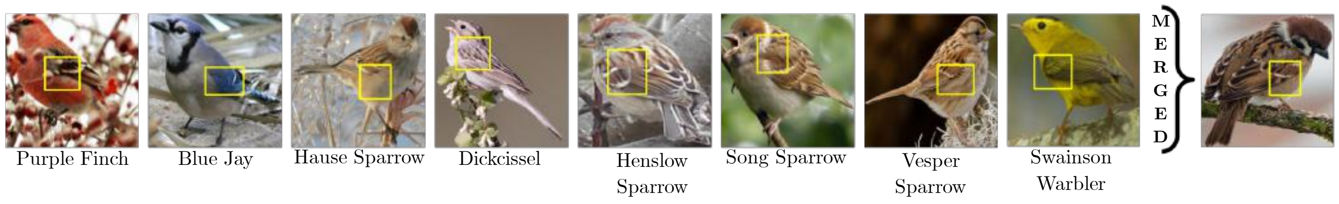


Figure 9: Eight prototypes from eight classes merged into one prototype by our ProtoPShare method. One can see that all prototypes correspond to the prototypical part of *a wing with light smudge*.

Discussion. We confirmed that ProtoPShare achieves better accuracy than other methods for almost all pruning rates. We demonstrated the need for small-steps pruning with finetuning only the last fully connected layer instead of the broader optimization. Additionally, we explained the need for two phases (training and pruning) and verified that the smaller number of prototypes does not negatively influence the model's robustness.

5 INTERPRETABILITY OF PROTOPSHARE

In the section, we focus on the interpretability of ProtoPShare using qualitative results and user study. We explain why the merge of prototypes is better than the original pruning from [4], illustrate how our model can discover the inter-class similarity, and demonstrate the superiority of data-dependent similarity over its data-independent counterpart.

Why merge-pruning instead of pruning? We decided to introduce our merge-pruning algorithm after a detailed analysis of the prototypes pruned by ProtoPNet and the conclusion that around 30% of them represent significant prototypical parts instead of a background. This observation is somehow consistent with Section 3.2 in [4], which states that "the prototypes whose nearest training patches have mixed class identities *usually* correspond to background patches, and they can be automatically pruned from the model." Nevertheless, we were surprised by the high percentage of meaningful prototypes that are pruned. To illustrate the problem, we present some of them in Figure 10. In our opinion, it negatively influences the model accuracy (what we show in Section 4) and the model explanations, as some of the important prototypes can be removed only because of its similarity to the other class. That is why we provide the merge-pruning that can join many semantically similar prototypes from different classes without significant decrease in accuracy. As an example, in Figure 9, we present eight prototypes from eight classes merged into one prototype corresponding to *a wing with light smudge.*

Inter-class similarity discovery. The positive property of ProtoPShare is its ability to discover the similarity between classes based on the prototypes they share. Such similarity can be represented, i.a., by the graph that we present in Figure 3 where each node corresponds to a class and the strength of an edge between two nodes corresponds to the number of shared prototypes. Such analysis has many applications, like finding similar prototypical parts between two classes or clustering them into groups. Moreover, such a graph can be the starting point for more advanced visualizations.

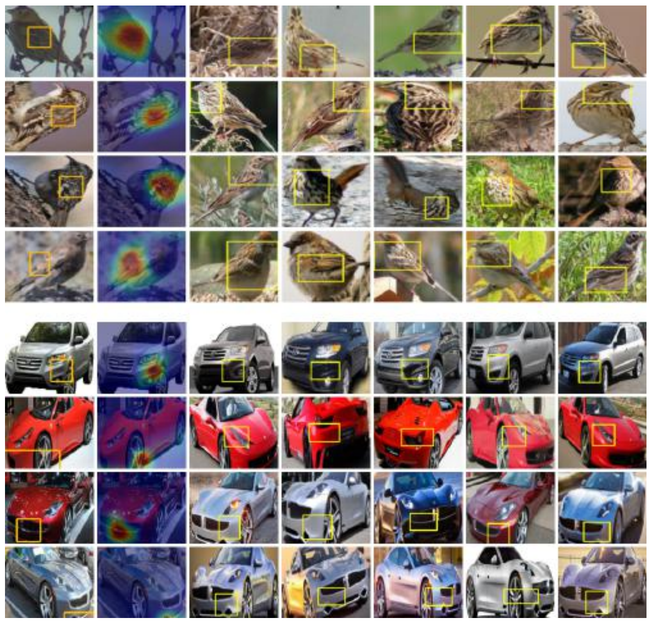
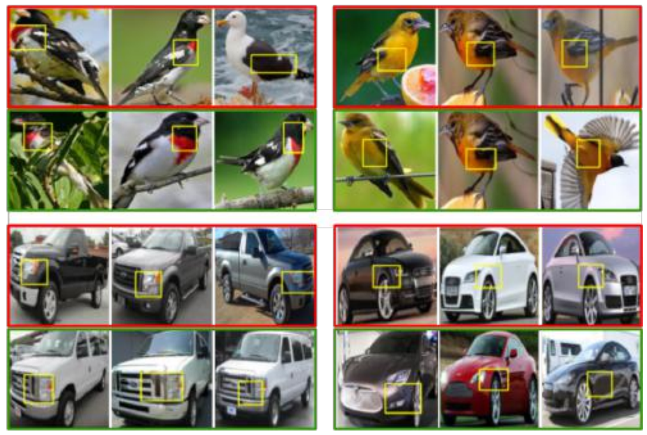


Figure 10: Example prototypes (one per row) pruned by ProtoPNet that do not represent the background. The first column corresponds to the image with the prototypical part (in a yellow bounding box). The second column shows the prototype's activation map for this image obtained as described in [4]. The remaining five columns present images' parts (in yellow bounding boxes) whose representations are closest to the considered prototype. We observe that none of the presented prototypes represent the background what illustrates that ProtoPNet removes many meaningful prototypes.

Why data-dependent similarity? To explain the superiority of our data-dependent over the data-independent similarity (an inverse of Euclidean distance), in Figure 2, we present the pairs of prototypes close according to the former, but distant according to the later. More examples are presented in Figure 11. It can be noticed that the prototypes of one pair are semantically similar, even though they sometimes differ in colors' distribution. In our opinion, the ability to find such pairs of similar prototypes is the main advantage of our data-dependent similarity. This ability is extremely important after the merge-pruning step because, as presented in Figure 12, ProtoPShare representations of the image's parts more often activate the prototypes assigned to their true class, which indirectly


(a) Pairs of semantically similar prototypes distant in the representation space but close according to our data-dependent similarity.

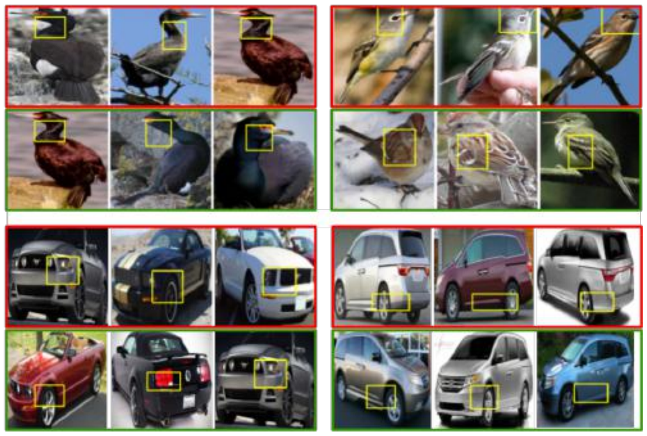

(b) Pairs of prototypes close in the representation space.

Figure 11: Similar prototypes according to data-dependent and data-independent metrics. Notice that each prototype is represented by one row of 3 closest images' parts (marked with yellow bounding-boxes), so each pair corresponds to 6 pictures.

results in higher accuracy and robustness. Notice that, to generate Figure 12, we create a dataset with 5 parts from each testing image with the highest prototype activation, and then analyze which of those activations correspond to a true class. We conclude that only ProtoPShare constantly increases this number, which means that it uses model capacity more effectively.

User study. To further verify the superiority of our data-dependent pruning, we conducted user study where we presented two pairs of prototypes to the users. One of the pair contains prototypes most similar according to our data-dependent similarity, while the other contains prototypes closest in the representation space. The pairs were presented at once but in random order to prevent the order bias, and the users were asked to indicate a more consistent pair of prototypes (with an additional option "Impossible to decide").

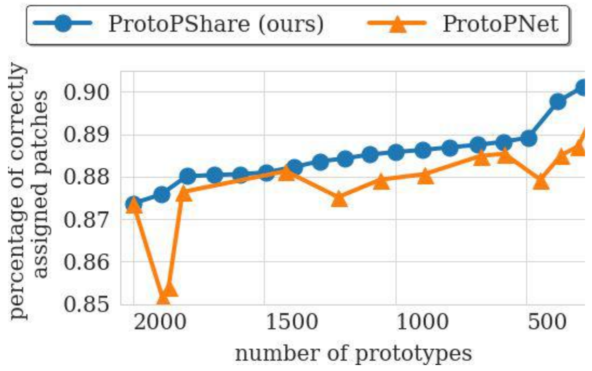


Figure 12: Percent of patches correctly assigned to their classes (higher is better) in the successive steps of the pruning. The ProtoPShare pruning method constantly increases this number, which means that it uses the model capacity more effectively than the original ProtoPNet pruning.

The study was conducted on 57 users asked 5 times for a different randomly chosen set of prototypes, resulting in 285 answers altogether. Among all the answers, 195 favored our approach, 61 preferred data-independent similarity, and in 29 cases users could not decide which similarity is better. Therefore, we conclude that our data-dependent similarity is considered as more consistent than its data-independent counterpart.

Discussion. In this section, we explained why prototype merge is preferred over background pruning and why it is crucial to use data-dependent instead of the data-independent similarity. Moreover, we showed how ProtoPShare can be used for similarity discovery.

6 CONCLUSIONS

We presented ProtoPShare, a self-explained method that incorporates the paradigm of prototypical parts to explain its predictions. The method extends the existing approaches because it can share the prototypes between classes, reducing their number up to three times and still preserving baseline accuracy. To efficiently share the prototypes, we introduced our data-dependent pruning that merges prototypes with similar semantics. As a result, we increased the model's interpretability and enabled similarity discovery while maintaining high accuracy, as we showed through theoretical results and many experiments, including user study.

ACKNOWLEDGMENTS

Dawid Rymarczyk part in this research was funded by the Priority Research Area Digiworld under the program Excellence Initiative – Research University at the Jagiellonian University in Kraków.

Łukasz Struski, Jacek Tabor and Bartosz Zieliński part was founded by the POIR.04.04.00-00-14DE/18-00 project carried out within the Team-Net programme of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund.

REFERENCES

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *NIPS*. 9505–9515.

[2] Vijay Arya, Rachel KE Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C Hoffman, Stephanie Houde, Q Vera Liao, Ronny Luss, Aleksandra Mojsilović, et al. 2019. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques. *arXiv preprint arXiv:1909.03012* (2019).

[3] Wieland Brendel and Matthias Bethge. 2019. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *ICLR* (2019).

[4] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. 2019. This looks like that: deep learning for interpretable image recognition. In *NIPS*. 8930–8941.

[5] Zhi Chen, Yijie Bei, and Cynthia Rudin. 2020. Concept Whitening for Interpretable Image Recognition. *arXiv:2002.01650* (2020).

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. Ieee, 248–255.

[7] Finale Doshi-Velez and Been Kim. 2017. A roadmap for a rigorous science of interpretability. *arXiv:1702.08608* 2 (2017).

[8] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. 2019. Understanding deep networks via extremal perturbations and smooth masks. In *ICCV*. 2950–2958.

[9] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*. 3429–3437.

[10] Jianlong Fu, Heliang Zheng, and Tao Mei. 2017. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *CVPR*. 4438–4446.

[11] Alan H Gee, Diego Garcia-Olano, Joydeep Ghosh, and David Paydarfar. 2019. Explaining deep classification of time-series data with learned prototypes. *arXiv:1904.08935* (2019).

[12] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. 2019. Towards automatic concept-based explanations. In *NIPS*. 9277–9286.

[13] Riccardo Guidotti, Anna Monreale, Stan Matwin, and Dino Pedreschi. 2019. Black Box Explanation by Learning Image Exemplars in the Latent Feature Space. In *ECML PKDD*. Springer, 189–205.

[14] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. 2019. Interpretable Image Recognition with Hierarchical Prototypes. In *AAAI*. 32–40.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[16] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. 2019. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *CVPR*. 4340–4349.

[17] Yihui He, Xiangyu Zhang, and Jian Sun. 2017. Channel pruning for accelerating very deep neural networks. In *ICCV*. 1389–1397.

[18] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *CVPR*. 4700–4708.

[19] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *ICML*. PMLR, 2668–2677.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980* (2014).

[21] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3D Object Representations for Fine-Grained Categorization. In *3dRR-13*. Sydney, Australia.

[22] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016. Pruning filters for efficient convnets. In *ICLR*.

[23] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. 2018. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. *AAAI* (2018).

[24] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. 2019. Towards optimal structured cnn pruning via generative adversarial learning. In *CVPR*. 2790–2799.

[25] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2019. Rethinking the value of network pruning. *ICLR* (2019).

[26] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. 2017. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*. 5058–5066.

[27] Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. 2019. Interpretable and steerable sequence learning via prototypes. In *KDD*. 903–913.

[28] P Molchanov, S Tyree, T Karras, T Aila, and J Kautz. 2019. Pruning convolutional neural networks for resource efficient inference. In *ICLR*.

[29] Esther Puyol-Antón, Chen Chen, James R Clough, Bram Ruijsink, Baldeep S Sidhu, Justin Gould, Bradley Porter, Marc Elliott, Vishal Mehta, Daniel Rueckert, et al. 2020. Interpretable Deep Models for Cardiac Resynchronisation Therapy Response Prediction. In *MICCAI*. Springer, 284–293.

[30] Sylvestre-Alvise Rebuffi, Ruth Fong, Xu Ji, and Andrea Vedaldi. 2020. There and Back Again: Revisiting Backpropagation Saliency Methods. In *CVPR*. 8839–8848.

[31] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should I trust you?" Explaining the predictions of any classifier. In *KDD*. 1135–1144.

[32] Ruslan Salakhutdinov, Joshua Tenenbaum, and Antonio Torralba. 2012. One-shot learning with a hierarchical nonparametric bayesian model. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. 195–206.

[33] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*. 618–626.

[34] Ramprasaath R Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry Heck, Dhruv Batra, and Devi Parikh. 2019. Taking a hint: Leveraging explanations to make vision and language models more grounded. In *ICCV*. 2591–2600.

[35] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv:1312.6034* (2013).

[36] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. *ICML* (2017), 3319–3328.

[37] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. The caltech-ucsd birds-200-2011 dataset. (2011).

[38] Dong Wang, Lei Zhou, Xueni Zhang, Xiao Bai, and Jun Zhou. 2018. Exploring linear relationship in feature map subspace for convnets compression. In *WACV*.

[39] Tong Wang. 2019. Gaining free or low-cost interpretability with interpretable partial substitute. In *ICML*. PMLR, 6505–6514.

[40] Tianjun Xiao, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, and Zheng Zhang. 2015. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *CVPR*. 842–850.

[41] Jianbo Ye, Xin Lu, Zhe Lin, and James Z Wang. 2018. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In *ICLR*.

[42] Chih-Kuan Yeh, Been Kim, Sercan O Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. 2019. On Completeness-aware Concept-Based Explanations in Deep Neural Networks. *arXiv:1910.07969* (2019).

[43] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. 2018. Nisp: Pruning networks using neuron importance score propagation. In *CVPR*. 9194–9203.

[44] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. 2017. Learning multi-attention convolutional neural network for fine-grained image recognition. In *ICCV*. 5209–5217.

[45] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. 2018. Interpretable basis decomposition for visual explanation. In *ECCV*. 119–134.

[46] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu. 2018. Discrimination-aware channel pruning for deep neural networks. In *NIPS*. 875–886.

A PROOF FOR THEOREM 1

In this section, we present proof of Theorem 1 that examines the changes in the network's predictions after the prototypes' merge.

Proof of Theorem 1. For any class k, let

$$L_k(x,P)=\sum_{p\in P} w_h(p)\cdot\log\left(\frac{\|z_p-p\|^2+1}{\|z_p-p\|^2+\varepsilon}\right)$$

denotes its output logit for an input image $x\in X$. From assumption (A5), before the merge, we have

$$L_k(x,P_k)=\sum_{p\in P_k}\log\left(\frac{\|z_p-p\|^2+1}{\|z_p-p\|^2+\varepsilon}\right),$$

and after merge

$$L_k(x,P'_k\cup S_k)=\sum_{p\in P'_k\cup S_k}\log\left(\frac{\|z_p-p\|^2+1}{\|z_p-p\|^2+\varepsilon}\right).$$

If none of the prototypes from class k is merged into the other prototype (i.e., $S_k=\emptyset$), then there are no changes at the output (i.e., $L_k(x,P'_k\cup S_k)-L_k(x,P_k)=0$). In the opposite case

$$L_k(x,P'_k\cup S_k)-L_k(x,P_k)=\sum_{p\in S_k}\log\left(\frac{\|z_p-p\|^2+1}{\|z_p-p\|^2+\varepsilon}\right)-$$

$$\sum_{p\in P_k\setminus P'_k}\log\left(\frac{\|z_p-p\|^2+1}{\|z_p-p\|^2+\varepsilon}\right)=\sum_{i=1}^{|P_k\setminus P'_k|}\log\left(\frac{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\cdot\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+\varepsilon}\right).$$

For each class $k\in\{1..K\}$ and its prototypes $i\in\{1..|P_k\setminus P'_k|\}$, let

$$\vartheta_i:=\frac{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\cdot\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+\varepsilon}.$$

To obtain the lower bound of ϑ_i for $k=c$ (correct class of x), we use the second inequality in (A4b), receiving

$$\frac{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\geq\frac{1}{\|z_{p_i}-p_i\|^2+1}\overset{by\ (A4b)}{\geq}\frac{1}{2-\delta}.$$

By the assumption (A3) and the triangle inequality we obtain $\|z_{\tilde{p}_i}-\tilde{p}_i\|\leq\|z_{p_i}-\tilde{p}_i\|\leq\|z_{p_i}-p_i\|+\|\tilde{p}_i-p_i\|$. From the assumption (A4b) we obtain $\|\tilde{p}_i-p_i\|\leq(\sqrt{1+\delta}-1)\|z_{p_i}-p_i\|$, what implicate $\|\tilde{p}_i-p_i\|+\|z_{p_i}-p_i\|\leq\sqrt{1+\delta}\|z_{p_i}-p_i\|$. Hence

$$\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+\varepsilon}\geq\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{(\|z_{p_i}-p_i\|+\|\tilde{p}_i-p_i\|)^2+\varepsilon}\geq\frac{1}{1+\delta}.$$

Combining the above inequalities, we have

$$\vartheta_i=\frac{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\cdot\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+\varepsilon}\geq\frac{1}{(1+\delta)(2-\delta)}.$$

It means that the output logit change of class c as a result of the prototype merge satisfies

$$L_c(x,P'_c\cup S_c)-L_c(x,P_c)=\sum_{i=1}^{|P_c\setminus P'_c|}\log\vartheta_i\geq-|P_c\setminus P'_c|\log((1+\delta)(2-\delta))$$

or equivalently, $L_c(x,P_c)-L_c(x,P'_c\cup S_c)\leq|P_c\setminus P'_c|\log((1+\delta)(2-\delta))$. Hence, the worst decrease of the class c output logit as a result of prototype merge is $\Delta^c_{\max}$.

To obtain the lower bound of ϑ_i for $k\neq c$ (incorrect class of x), we use the triangle inequality together with assumptions (A3) and (A4a), receiving

$$\|z_{\tilde{p}_i}-\tilde{p}_i\|^2\overset{by\ (A3)}{\leq}\|z_{p_i}-\tilde{p}_i\|^2\leq(\|z_{p_i}-p_i\|+\|\tilde{p}_i-p_i\|)^2$$
$$\overset{by\ (A4a)}{\leq}(\|z_{p_i}-p_i\|+(\sqrt{1+\delta}-1)\|z_{p_i}-p_i\|)^2=(1+\delta)\|z_{p_i}-p_i\|^2.$$

Hence

$$\frac{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\leq\frac{(1+\delta)\|z_{p_i}-p_i\|^2+1}{\|z_{p_i}-p_i\|^2+1}\leq1+\delta. \tag{3}$$

To derive an upper bound for the second part of ϑ_i, we use the assumption (A3) and (A4a)

$$\|p_i-\tilde{p}_i\|\overset{by\ (A4a)}{\leq}\theta\|z_{p_i}-p_i\|-\sqrt{\varepsilon}\overset{by\ (A3)}{\leq}\theta\|z_{\tilde{p}_i}-p_i\|-\sqrt{\varepsilon}$$
$$\overset{by\ (A4a)}{\leq}\left(1-\frac{1}{\sqrt{2-\delta}}\right)\|z_{\tilde{p}_i}-p_i\|-\sqrt{\varepsilon}.$$

From which we know that $\|z_{\tilde{p}_i}-p_i\|-\|\tilde{p}_i-p_i\|>0$. Again, we use the triangle inequality $\|z_{\tilde{p}_i}-p_i\|\leq\|z_{\tilde{p}_i}-\tilde{p}_i\|+\|\tilde{p}_i-p_i\|$, what implicate $\|z_{\tilde{p}_i}-\tilde{p}\|\geq\|z_{\tilde{p}_i}-p_i\|-\|\tilde{p}_i-p_i\|$. By reusing assumptions (A3) and (A4a), we have

$$\frac{\|z_{p_i}-p_i\|+\sqrt{\varepsilon}}{\sqrt{2-\delta}}\leq\frac{1}{\sqrt{2-\delta}}\|z_{p_i}-p_i\|+\sqrt{\varepsilon}\overset{\text{by (A4a)}}{\leq}\|z_{p_i}-p_i\|-\|\tilde{p}_i-p_i\|$$
$$\overset{\text{by (A3)}}{\leq}\|z_{\tilde{p}_i}-p_i\|-\|\tilde{p}_i-p_i\|.$$

Thanks to the above, we get

$$\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{\|z_{\tilde{p}_i}-\tilde{p}_i\|^2+\varepsilon}\leq\frac{\|z_{p_i}-p_i\|^2+\varepsilon}{(\|z_{\tilde{p}_i}-p_i\|-\|\tilde{p}_i-p_i\|)^2+\varepsilon}\leq\left(\frac{\|z_{p_i}-p_i\|+\varepsilon}{\|z_{\tilde{p}_i}-p_i\|-\|\tilde{p}_i-p_i\|}\right)^2\leq2-\delta. \tag{4}$$

By inequalities (3), (4) get that the output logit change of class k as a result of the prototype merge satisfies

$$L_k(x,P'_k\cup S_k)-L_k(x,P_k)=\sum_{i=1}^{|P_k\setminus P'_k|}\log\vartheta_i\leq|P_k\setminus P'_k|\log((1+\delta)(2-\delta)).$$

Hence, the worst increase of the class k output logit as a result of prototype merge is $\Delta^k_{\max}$.

To prove the last thesis, let us assume that the output logit $L_c(x,P_k)$ of the correct class c before prototype merge is at least $\Delta^c_{\max}+\max_{k\neq c}(\Delta^k_{\max})$ higher than the output logit $L_k(x,P_k)$ of any other class $k\neq c$, i.e.,

$$L_c(x,P_c)\geq L_k(x,P_k)+\Delta^c_{\max}+\max_{k\neq c}(\Delta^k_{\max})\quad\text{for}\quad k\neq c. \tag{5}$$

Since the output logit of the correct class c satisfies

$$L_c(x,P'_c\cup S_c)\geq L_c(x,P_c)-\Delta^c_{\max} \tag{6}$$

and the output logit of any class $k\neq c$ satisfies

$$L_k(x,P'_k\cup S_k)\leq L_k(x,P_k)+\Delta^k_{\max}, \tag{7}$$

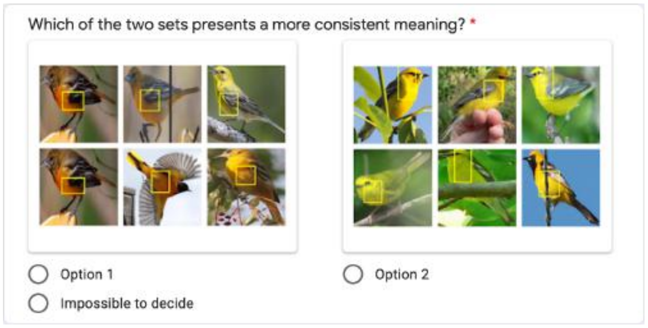
for any $k\neq c$ we have $L_c(x,P'_c\cup S_c)\geq L_c(x,P_c)-\Delta^c_{\max}\geq L_k(x,P_k)+\max_{k\neq c}(\Delta^k_{\max})\geq L_k(x,P'_k\cup S_k)$ (inequalities (6), (5), (7) give the individual inequalities, respectively). Hence, the input image x will still be correctly classified as class c after prototype merge. □

B TRAINING DETAILS

We decided to omit the training details in the paper for its clarity. However, to make the paper reproducible, we provide them in this section.

First of all, we perform exhaustive offline data augmentation using rotation, skewing, flipping, and shearing with the probability of 0.5 for each operation. When it comes to the architecture, f is followed by two additional 1×1 convolutional layers before the prototypes' layer, and we use ReLU as the activation function for all convolutional layers except the last one (with the sigmoid function). The training bases on the images cropped (using the bounding-boxes provided with the datasets) and resized to 224×224 pixels. As a result, we obtain a convolutional feature map of size $7 \times 7 \times 256$. Hence, prototypes have size $1 \times 1 \times 256$. During training, we use Adam optimizer [20] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, batch size 80, learning rates 10^{-4} for f and h, and $3 \cdot 10^{-3}$ for additional 1×1 convolutions and prototypes' layer. For the loss function defined in [4], we use weights $\lambda_1 = 0.8$, $\lambda_2 = 0.08$, and $\lambda_l = 10^{-4}$. Moreover, the learning rates for convolutional parts and prototype layer are multiplied by 0.1 every 5 epochs of the training.

C USER STUDY QUESTIONNAIRE


Which of the two sets presents a more consistent meaning? *

Option 1

Option 2

Impossible to decide

Figure 13: Example question from our user study.

To further verify the superiority of our data-dependent pruning, we conduct a user study where we present five questions to the user. There were four versions of the questionnaire with different sets of questions. Each questionnaire had the following initial instruction: "In this form, you will see two sets of images with yellow bounding-boxes. All bounding-boxes of one set should present one semantic meaning that corresponds to a particular characteristic of a bird's part (e.g. white head, dark neck, striped wing, or black legs). Please, decide which of the two sets presents more consistent bounding-boxes. This task can be difficult, as two sets you compare can have a different meaning. Nevertheless, try to decide which of them is more consistent, and if it is impossible, please choose the option Impossible to decide." After the initial instruction, we asked five times, "Which of the two sets presents a more consistent meaning?". An example question is presented in Figure 13. We present the detailed results of the study in the paper.

Interpretable Image Classification with Differentiable Prototypes Assignment

Dawid Rymarczyk[1,2], Łukasz Struski[1], Michał Górszczak[1], Koryna Lewandowska[3], Jacek Tabor[1], and Bartosz Zieliński[1,2]

[1] Faculty of Mathematics and Computer Science, Jagiellonian University
[2] Ardigen SA
[3] Department of Cognitive Neuroscience and Neuroergonomics, Institute of Applied Psychology, Jagiellonian University

Abstract. Existing prototypical-based models address the black-box nature of deep learning. However, they are sub-optimal as they often assume separate prototypes for each class, require multi-step optimization, make decisions based on prototype absence (so-called negative reasoning process), and derive vague prototypes. To address those shortcomings, we introduce ProtoPool, an interpretable prototype-based model with positive reasoning and three main novelties. Firstly, we reuse prototypes in classes, which significantly decreases their number. Secondly, we allow automatic, fully differentiable assignment of prototypes to classes, which substantially simplifies the training process. Finally, we propose a new focal similarity function that contrasts the prototype from the background and consequently concentrates on more salient visual features. We show that ProtoPool obtains state-of-the-art accuracy on the CUB-200-2011 and the Stanford Cars datasets, substantially reducing the number of prototypes. We provide a theoretical analysis of the method and a user study to show that our prototypes capture more salient features than those obtained with competitive methods. We made the code available at `https://github.com/gmum/ProtoPool`.

Keywords: deep learning; interpretability; case-based reasoning

1 Introduction

The broad application of deep learning in fields like medical diagnosis [3] and autonomous driving [53], together with current law requirements (such as GDPR in EU [21]), enforces models to explain the rationale behind their decisions. That is why explainers [6,23,29,39,44] and self-explainable [4,7,58] models are developed to justify neural network predictions. Some of them are inspired by mechanisms used by humans to explain their decisions, like matching image parts with memorized prototypical features that an object can poses [8,27,34,43].

Recently, a self-explainable model called Prototypical Part Network (ProtoPNet) [8] was introduced, employing feature matching learning theory [40,41]. It focuses on crucial image parts and compares them with reference patterns

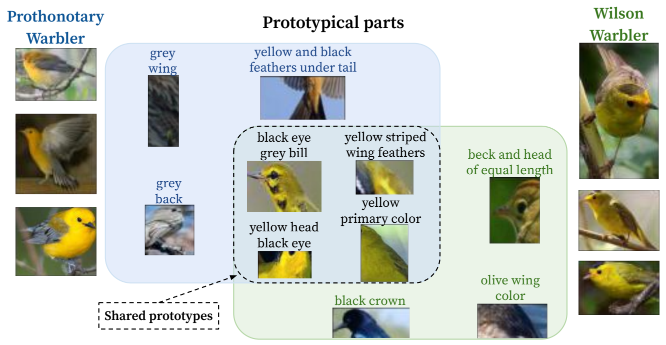


Fig. 1: Automatically discovered prototypes[4] for two classes, *Prothonotary Warbler* and *Wilson Warbler* (each class represented by three images on left and right side). Three prototypical parts on the blue and green background are specific for a *Prothonotary Warbler* and *Wilson Warbler*, respectively (they correspond to heads and wings feathers). At the same time, four prototypes shared between those classes (related to yellow feathers) are presented in the intersection. Prototypes sharing reduces their amount, leads to a more interpretable model, and discovers classes similarities.

(prototypical parts) assigned to classes. The comparison is based on a similarity metric between the image activation map and representations of prototypical parts (later called prototypes). The maximum value of similarity is pooled to the classification layer. As a result, ProtoPNet explains each prediction with a list of reference patterns and their similarity to the input image. Moreover, a global explanation can be obtained for each class by analyzing prototypical parts assigned to particular classes.

However, ProtoPNet assumes that each class has its own separate set of prototypes, which is problematic because many visual features occur in many classes. For instance, both *Prothonotary Warbler* and *Wilson Warbler* have yellow as a primary color (see Figure 1). Such limitation of ProtoPNet hinders the scalability because the number of prototypes grows linearly growing number of classes. Moreover, a large number of prototypes makes ProtoPNet hard to interpret by the users and results in many background prototypes [43].

To address these limitations, ProtoPShare [43] and ProtoTree [34] were introduced. They share the prototypes between classes but suffer from other drawbacks. ProtoPShare requires previously trained ProtoPNet to perform the merge-pruning step, which extends the training time. At the same time, ProtoTree builds a decision tree and exploits the negative reasoning process that may result in explanations based only on prototype absence. For example, a model can predict a *sparrow* because an image does not contain red feathers, a long beak,

[4] Names of prototypical parts were generated based on the annotations from CUB-200-2011 dataset (see details in Supplementary Materials).

and wide wings. While this characteristic is true in the case of a *sparrow*, it also matches many other species.

To deal with the above shortcomings, we introduce ProtoPool, a self-explainable prototype model for fine-grained images classification. ProtoPool introduces significantly novel mechanisms that substantially reduce the number of prototypes and obtain higher interpretability and easier training. Instead of using hard assignment of prototypes to classes, we implement the soft assignment represented by a distribution over the set of prototypes. This distribution is randomly initialized and binarized during training using the Gumbel-Softmax trick. Such a mechanism simplifies the training process by removing the pruning step required in ProtoPNet, ProtoPShare, and ProtoTree. The second novelty is a focal similarity function that focuses the model on the salient features. For this purpose, instead of maximizing the global activation, we widen the gap between the maximal and average similarity between the image activation map and prototypes (see Figure 4). As a result, we reduce the number of prototypes and use the positive reasoning process on salient features, as presented in Figure 2 and Figure 10.

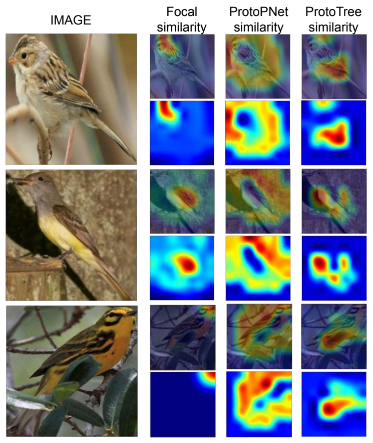


Fig. 2: Focal similarity focuses the prototype on a salient visual feature. While the other similarity metrics are more distributed through the image, making the interpretation harder to comprehend. It is shown with three input images, the prototype activation map, and its overlay.

We confirm the effectiveness of ProtoPool with theoretical analysis and exhaustive experiments, showing that it achieves the highest accuracy among models with a reduced number of prototypes. What is more, we discuss interpretability, perform a user study, and discuss the cognitive aspects of the ProtoPool over existing methods.

The main achievements of the paper can be summarized as follows:

- We construct ProtoPool, a case-based self-explainable method that shares prototypes between data classes without any predefined concept dictionary.
- We introduce fully differentiable assignments of prototypes to classes, allowing the end-to-end training.
- We define a novel similarity function, called focal similarity, that focuses the model on the salient features.
- We increase interpretability by reducing prototypes number and providing explanations in a positive reasoning process.

2 Related works

Attempts to explain deep learning models can be divided into the post hoc and self-explainable [42] methods. The former approaches assume that the reasoning process is hidden in a black box model and a new explainer model has to be created to reveal it. Post hoc methods include a saliency map [31,38,44,45,46] generating a heatmap of crucial image parts, or Concept Activation Vectors (CAV) explaining the internal network state as user-friendly concepts [9,14,23,25,55]. Other methods provide counterfactual examples [1,15,33,36,52] or analyze the networks' reaction to the image perturbation [6,11,12,39]. Post hoc methods are easy to implement because they do not interfere with the architecture, but they can produce biased and unreliable explanations [2]. That is why more focus is recently put on designing self-explainable models [4,7] that make the decision process directly visible. Many interpretable solutions are based on the attention [28,48,54,57,58,59] or exploit the activation space [16,37], e.g. with adversarial autoencoder. However, most recent approaches built on an interpretable method introduced in [8] (ProtoPNet) with a hidden layer of prototypes representing the activation patterns.

ProtoPNet inspired the design of many self-explainable models, such as TesNet [51] that constructs the latent space on a Grassman manifold without prototypes reduction. Other models like ProtoPShare [43] and ProtoTree [34] reduce the number of prototypes used in the classification. The former introduces data-dependent merge-pruning that discovers prototypes of similar semantics and joins them. The latter uses a soft neural decision tree that may depend on the negative reasoning process. Alternative approaches organize the prototypes hierarchically [17] to classify input at every level of a predefined taxonomy or transform prototypes from the latent space to data space [27]. Moreover, prototype-based solutions are widely adopted in various fields such as medical imaging [3,5,24,47], time-series analysis [13], graphs classification [56], and sequence learning [32].

3 ProtoPool

In this section, we describe the overall architecture of ProtoPool presented in Figure 3 and the main novelties of ProtoPool compared to the existing models, including the mechanism of assigning prototypes to slots and the focal similarity. Moreover, we provide a theoretical analysis of the approach.

Overall architecture The architecture of ProtoPool, shown in Figure 3, is generally inspired by ProtoNet [8]. It consists of convolutional layers f, a prototype pool layer g, and a fully connected layer h. Layer g contains a pool of M trainable prototypes $P = \{p_i \in \mathbb{R}^D\}_{i=1}^M$ and K slots for each class. Each slot is implemented as a distribution $q_k \in \mathbb{R}^M$ of prototypes available in the pool, where successive values of q_k correspond to the probability of assigning successive prototypes to slot k ($\|q_k\| = 1$). Layer h is linear and initialized to enforce

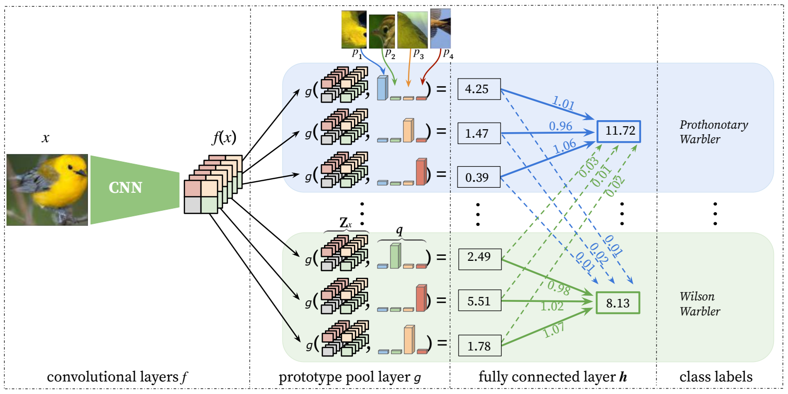


Fig. 3: The architecture of our ProtoPool with a prototype pool layer g. Layer g contains a pool of prototypes $p_1 - p_4$ and three slots per class. Each slot is implemented as a distribution $q \in \mathbb{R}^4$ of prototypes from the pool, where successive values of q correspond to the probability of assigning successive prototypes to the slot. In this example, p_1 and p_2 are assigned to the first slot of *Prothonotary Warbler* and *Wilson Warbler*, respectively. At the same time, the shared prototypes p_3 and p_4 are assigned to the second and third slots of both classes.

the positive reasoning process, i.e. weights between each class c and its slots are initialized to 1 while remaining weights of h are set to 0.

Given an input image $x \in X$, the convolutional layers first extract image representation $f(x)$ of shape $H \times W \times D$, where H and W are the height and width of representation obtained at the last convolutional layer for image x, and D is the number of channels in this layer. Intuitively, $f(x)$ can be considered as a set of $H \cdot W$ vectors of dimension D, each corresponding to a specific location of the image (as presented in Figure 3). For the clarity of description, we will denote this set as $Z_x = \{z_i \in f(x) : z_i \in \mathbb{R}^D, i = 1, ..., H \cdot W\}$. Then, the prototype pool layer is used on each k-th slot to compute the aggregated similarity $g_k = \sum_{i=1}^{M} q_k^i g_{p_i}$ between Z_x and all prototypes considering the distribution q_k of this slot, where g_p is defined below. Finally, the similarity scores (K values per class) are multiplied by the weight matrix w_h in the fully connected layer h. This results in the output logits, further normalized using softmax to obtain a final prediction.

Focal similarity In ProtoPNet [8] and other models using prototypical parts, the similarity of point z to prototype p is defined as[5] $g_p(z) = \log(1 + \frac{1}{\|z-p\|^2})$, and the final activation of the prototype p with respect to image x is given by $g_p = \max_{z \in Z_x} g_p(z)$. One can observe that such an approach has two possible disadvantages. First, high activation can be obtained when all the elements in Z_x are similar to a prototype. It is undesirable because the prototypes can then

[5] The following regularization is used to avoid numerical instability in the experiments: $g_p(z) = \log(\frac{\|z-p\|^2+1}{\|z-p\|^2+\varepsilon})$, with a small $\varepsilon > 0$.

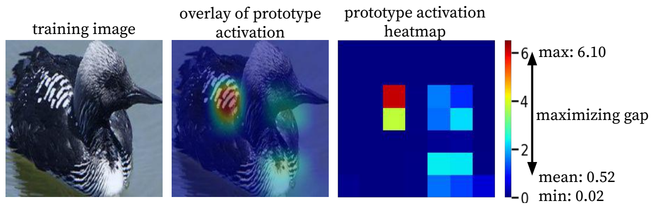


Fig. 4: Our focal similarity limits high prototype activation to a narrow area (corresponding to white and black striped wings). It is obtained by widening the gap between the maximal and average activation (equal 6.10 and 0.52, respectively). As a result, our prototypes correspond to more salient features (according to our user studies described in Section 5).

concentrate on the background. The other negative aspect concerns the training process, as the gradient is passed only through the most active part of the image.

To prevent those behaviors, in ProtoPool, we introduce a novel focal similarity function that widens the gap between maximal and average activation

$$g_p = \max_{z \in Z_x} g_p(z) - \operatorname*{mean}_{z \in Z_x} g_p(z), \tag{1}$$

as presented in Figure 4. The maximal activation of focal similarity is obtained if a prototype is similar to only a narrow area of the image x (see Figure 2). Consequently, the constructed prototypes correspond to more salient features (according to our user studies described in Section 5), and the gradient passes through all elements of Z_x.

Assigning one prototype per slot Previous prototypical methods use the hard predefined assignment of the prototypes to classes [8,43,51] or nodes of a tree [34]. Therefore, no gradient propagation is needed to model the prototypes assignment. In contrast, our ProtoPool employs a soft assignment based on prototypes distributions to use prototypes from the pool optimally. To generate prototype distribution q, one could apply softmax on the vector of size $\mathbb{R}^M$. However, this could result in assigning many prototypes to one slot and consequently could decrease the interpretability. Therefore, to obtain distributions with exactly one probability close to 1, we require a differentiable arg max function. A perfect match, in this case is the Gumbel-Softmax estimator [20], where for $q = (q^1, \ldots, q^M) \in \mathbb{R}^M$ and $\tau \in (0, \infty)$

$$\text{Gumbel-softmax}(q, \tau) = (y^1, \ldots, y^M) \in \mathbb{R}^M,$$

where $y^i = \frac{\exp\left((q^i + \eta_i)/\tau\right)}{\sum_{m=1}^{M} \exp((q^m + \eta_m)/\tau)}$ and η_m for $m \in 1, .., M$ are samples drawn from standard Gumbel distribution. The Gumbel-Softmax distribution interpolates between continuous categorical densities and discrete one-hot-encoded categorical distributions, approaching the latter for low temperatures $\tau \in [0.1, 0.5]$ (see Figure 5).

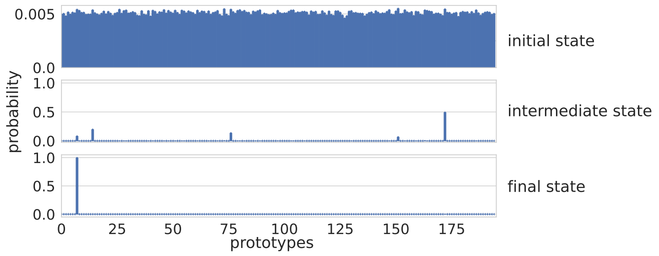


Fig. 5: A sample distribution (slot) at the initial, middle, and final step of training. In the beginning, all prototypes are assigned with a probability of 0.005. Then, the distribution binarizes, and finally, one prototype is assigned to this slot with a probability close to 1.

Slots orthogonality Without any additional constraints, the same prototype could be assigned to many slots of one class, wasting the capacity of the prototype pool layer and consequently returning poor results. Therefore, we extend the loss function with

$$\mathcal{L}_{orth} = \sum_{i<j}^{K} \frac{\langle q_i, q_j \rangle}{\|q_i\|_2 \cdot \|q_j\|_2}, \tag{2}$$

where $q_1, .., q_K$ are the distributions of a particular class. As a result, successive slots of a class are assigned to different prototypes.

Prototypes projection Prototypes projection is a step in the training process that allows prototypes visualization. It replaces each abstract prototype learned by the model with the representation of the nearest training patch. For prototype p, it can be expressed by the following formula

$$p \leftarrow \arg\min_{z \in Z_C} \|z - p\|_2, \tag{3}$$

where $Z_C = \{z : z \in Z_x \text{ for all } (x, y) : y \in C\}$. In contrast to [8], set C is not a single class but the set of classes assigned to prototype p.

Theoretical analysis Here, we theoretically analyze why ProtoPool assigns one prototype per slot and why each prototype does not repeat in a class. For this purpose, we provide two observations.

Observation 1 *Let $q \in [0,1]^M$, $\sum q_i = 1$ be a distribution (slot) of a particular class. Then, the limit of* Gumbel-softmax(q, τ)*, as τ approaches zero, is the canonical vector $e_i \in \mathbb{R}^M$, i.e. for q there exists $i = 1, .., M$ such that* $\lim_{\tau \to 0}$ Gumbel-softmax$(q, \tau) = e_i$.

The temperature parameter $\tau > 0$ controls how closely the new samples approximate discrete one-hot vectors (the canonical vector). From paper [20] we know that as $\tau \to 0$, the softmax computation smoothly approaches the $\arg\max$, and the sample vectors approach one-hot q distribution (see Figure 5).

Table 1: Comparison of ProtoPool with other prototypical methods trained on the CUB-200-2011 and Stanford Cars datasets, which considers a various number of prototypes and types of convolutional layers f. In the case of the CUB-200-2011 dataset, ProtoPool achieves the highest accuracy than other models, even those containing ten times more prototypes. Moreover, the ensemble of three ProtoPools surpasses the ensemble of five TesNets with 17 times more prototypes. On the other hand, in the case of Stanford Cars, ProtoPool achieves competitive results with significantly fewer prototypes. Please note that the results are first sorted by backbone network and then by the number of prototypes, R stands for ResNet, iN means pretrained on iNaturalist, and Ex is an ensemble of three or five models.

CUB-200-2011			
Model	Arch.	Proto. #	Acc [%]
ProtoPool (ours)	R34	202	80.3±0.2
ProtoPShare [43]	R34	400	74.7
ProtoPNet [8]	R34	1655	79.5
TesNet [51]	R34	2000	82.7±0.2
ProtoPool (ours)	R152	202	81.5±0.1
ProtoPShare [43]	R152	1000	73.6
ProtoPNet [8]	R152	1734	78.6
TesNet [51]	R152	2000	82.8±0.2
ProtoPool (ours)	iNR50	202	85.5±0.1
ProtoTree [34]	iNR50	202	82.2±0.7
ProtoPool (ours)	Ex3	202×3	87.5
ProtoTree [34]	Ex3	202×3	86.6
ProtoPool (ours)	Ex5	202×5	**87.6**
ProtoTree [34]	Ex5	202×5	87.2
ProtoPNet [8]	Ex5	2000×5	84.8
TesNet [51]	Ex5	2000×5	86.2

Stanford Cars			
Model	Arch.	Proto. #	Acc [%]
ProtoPool (ours)	R34	195	89.3±0.1
ProtoPShare [43]	R34	480	86.4
ProtoPNet [8]	R34	1960	86.1±0.2
TesNet [51]	R34	1960	92.6±0.3
ProtoPool (ours)	R50	195	88.9±0.1
ProtoTree [34]	R50	195	86.6±0.2
ProtoPool (ours)	Ex3	195×3	91.1
ProtoTree [34]	Ex3	195×3	90.5
ProtoPool (ours)	Ex5	195×5	91.6
ProtoTree [34]	Ex5	195×5	91.5
ProtoPNet [8]	Ex5	1960×5	91.4
TesNet [51]	Ex5	1960×5	**93.1**

Observation 2 *Let $K \in \mathbb{N}$ and $q_1, .., q_K$ be the distributions (slots) of a particular class. If $\mathcal{L}_{orth}$ defined in Eq. (2) is zero, then each prototype from a pool is assigned to only one slot of the class.*

It follows the fact that $\mathcal{L}_{orth} = 0$ only if $\langle q_i, q_j \rangle = 0$ for all $i < j \leq K$, i.e. only if q_i, q_j have non-zero values for different prototypes.

4 Experiments

We train our model on CUB-200-2011 [50] and Stanford Cars [26] datasets to classify 200 bird species and 196 car models, respectively. As the convolutional layers f of the model, we take ResNet-34, ResNet-50, ResNet-121 [18], DenseNet-121, and DenseNet-161 [19] without the last layer, pretrained on ImageNet [10].

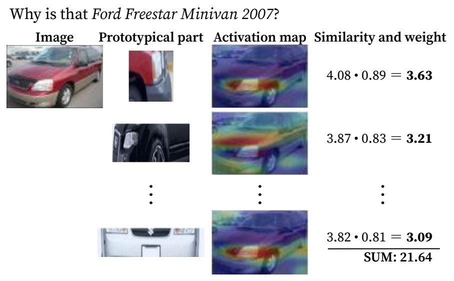


Fig. 6: Sample explanation of *Ford Freestar Minivan 2007* predictions. Except for an image, we present a few prototypical parts of this class, their activation maps, similarity function values, and the last layer weights. Moreover, we provide the sum of the similarities multiplied by the weights. ProtoPool returns the class with the largest sum as a prediction.

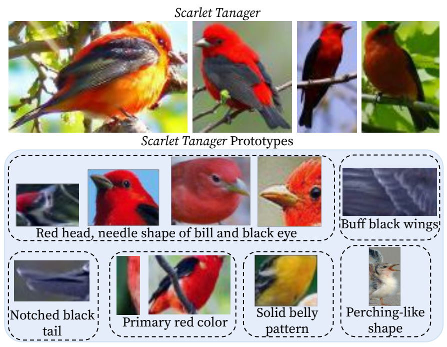


Fig. 7: Samples of *Scarlet Tanager* and prototypical parts assigned to this class by our ProtoPool model. Prototypes correspond, among others, to primary red color of feathers, black eye, perching-like shape, black notched tail, and black buff wings[6].

The one exception is ResNet-50 used with CUB-200-2011 dataset, which we pretrain on iNaturalist2017 [49] for fair comparison with ProtoTree model [34]. In the testing scenario, we make the prototype assignment hard, i.e. we set all values of a distribution q higher than 0.5 to 1, and the remaining values to 0 otherwise. We set the number of prototypes assigned to each class to be at most 10 and use the pool of 202 and 195 prototypical parts for CUB-200-2011 and Stanford Cars, respectively. Details on experimental setup and results for other backbone networks are provided in the Supplementary Materials.

Comparison with other prototypical models In Table 1 we compare the efficiency of our ProtoPool with other models based on prototypical parts. We report the mean accuracy and standard error of the mean for 5 repetitions. Additionally, we present the number of prototypes used by the models, and we use this parameter to sort the results. We compare ProtoPool with ProtoPNet [8], ProtoPShare [43], ProtoTree [34], and TesNet [51].

One can observe that ProtoPool achieves the highest accuracy for the CUB-200-2011 dataset, surpassing even the models with a much larger number of prototypical parts (TesNet and ProtoPNet). For Stanford Cars, our model still performs better than other models with a similarly low number of prototypes, like ProtoTree and ProtoPShare, and slightly worse than TesNet, which uses ten times more prototypes. The higher accuracy of the latter might be caused by prototype orthogonality enforced in training. Overall, our method achieves competitive results with significantly fewer prototypes. However, ensemble ProtoPool or TesNet should be used if higher accuracy is preferred at the expense of interpretability.

Table 2: Characteristics of prototypical methods for fine-grained image classification that considers the number of prototypes, reasoning type, and prototype sharing between classes. ProtoPool uses 10% of ProtoPNet's prototypes but only with positive reasoning. It shares the prototypes between classes but, in contrast to ProtoPShare, is trained in an end-to-end, fully differentiable manner. Please notice that 100% corresponds to 2000 and 1960 of prototypes for CUB-200-2011 and Stanford Cars datasets, respectively.

Model	ProtoPool	ProtoTree	ProtoPShare	ProtoPNet	TesNet
Portion of prototypes	∼10%	∼10%	[20%;50%]	100%	100%
Reasoning type	+	+/−	+	+	+
Prototype sharing	direct	indirect	direct	none	none

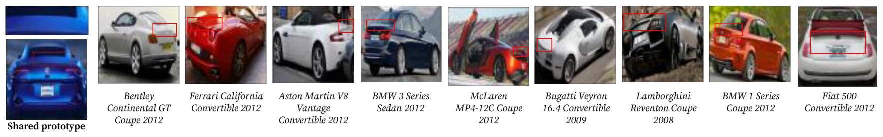

Shared prototype
Bentley Continental GT Coupe 2012
Ferrari California Convertible 2012
Aston Martin V8 Vantage Convertible 2012
BMW 3 Series Sedan 2012
McLaren MP4-12C Coupe 2012
Bugatti Veyron 16.4 Convertible 2009
Lamborghini Reventon Coupe 2008
BMW 1 Series Coupe 2012
Fiat 500 Convertible 2012

Fig. 8: Sample prototype of a *convex tailgate* (left top corner) shared by nine classes. Most of the classes correspond to luxury cars, but some exceptions exist, such as *Fiat 500*.

5 Interpretability

In this section, we analyze the interpretability of the ProtoPool model. Firstly, we show that our model can be used for local and global explanations. Then, we discuss the differences between ProtoPool and other prototypical approaches, and investigate its stability. Then, we perform a user study on the similarity functions used by the ProtoPNet, ProtoTree, and ProtoPool to assess the saliency of the obtained prototypes. Lastly, we consider ProtoPool from the cognitive psychology perspective.

Local and global interpretations Except for local explanations that are similar to those provided by the existing methods (see Figure 6), ProtoPool can provide a global characteristic of a class. It is presented in Figure 7, where we show the prototypical parts of *Scarlet Tanager* that correspond to the visual features of this species, such as red feathers, a puffy belly, and a short beak. Moreover, similarly to ProtoPShare, ProtoPool shares the prototypical parts between data classes. Therefore, it can describe the relations between classes relying only on the positive reasoning process, as presented in Figure 1 (in contrast, ProtoTree also uses negative reasoning). In Figure 8, we further provide visualization of the prototypical part shared by nine classes. More examples are provided in Supplementary Materials.

Differences between prototypical methods In Table 2, we compare the characteristics of various prototypical-based methods. Firstly, ProtoPool and

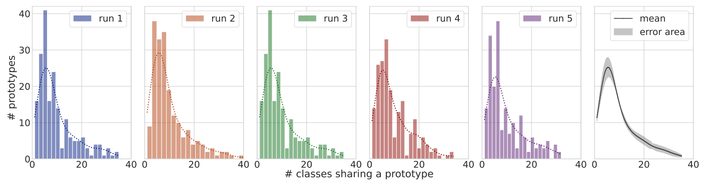


Fig. 9: Distribution presenting how many prototypes are shared by the specific number of classes (an estimation plot is represented with a dashed line). Each color corresponds to a single ProtoPool training on Stanford Cars dataset with ResNet50 as a backbone network. The right plot corresponds to the mean and standard deviation for five training runs. One can observe that the distribution behaves stable between runs.

ProtoTree utilize fewer prototypical parts than ProtoPNet and TesNet (around 10%). ProtoPShare also uses fewer prototypes (up to 20%), but it requires a trained ProtoPNet model before performing merge-pruning. Regarding class similarity, it is directly obtained from ProtoPool slots, in contrast to ProtoTree, which requires traversing through the decision tree. Moreover, ProtoPNet and TesNet have no mechanism to detect inter-class similarities. Finally, ProtoTree depends, among others, on *negative* reasoning process, while in the case of ProtoPool, it relies only on the positive reasoning process, which is a desirable feature according to [8].

Stability of shared prototypes The natural question that appears when analyzing the assignment of the prototypes is: *Does the similarity between two classes hold for many runs of ProtoPool training?* To analyze this behavior, in Figure 9 we show five distributions for five training runs. They present how many prototypes are shared by the specific number of classes. One can observe that difference between runs is negligible. In all runs, most prototypes are shared by five classes, but there exist prototypes shared by more than thirty classes. Moreover, on average, a prototype is shared by 2.73 ± 0.51 classes. A sample inter-class similarity graph is presented in the Supplementary Materials.

User study on focal similarity To validate if using focal similarity results in more salient prototypical parts, we performed a user study where we asked the participants to answer the question: *"How salient is the feature pointed out by the AI system?"*. The task was to assign a score from 1 to 5 where 1 meant *"Least salient"* and 5 meant *"Most salient"*. Images were generated using prototypes obtained for ProtoPool with ProtoPNets similarity or with focal similarity and from a trained ProtoTree[7]. To perform the user study, we used Amazon Mechan-

[6] Names of prototypical parts were generated based on the annotations from CUB-200-2011 dataset (see details in Supplementary Materials).

[7] ProtoTree was trained using code from `https://github.com/M-Nauta/ProtoTree` and obtained accuracy similar to [34]. For ProtoPNet similarity, we used code from `https://github.com/cfchen-duke/ProtoPNet`.

ical Turk (AMT) system[8]. To assure the reliability of the answers, we required the users to be masters according to AMT. 40 workers participated in our study and answered 60 questions (30 per dataset) presented in a random order, which resulted in 2400 answers. Each question contained an original training image and the same image with overlayed activation map, as presented in Figure 2.

Results presented in Figure 10 show that ProtoPool obtains mostly scores from 3 to 5, while other methods often obtain lower scores. We obtained a mean value of scores equal to 3.66, 2.87, and 2.85 for ProtoPool, ProtoTree, and ProtoPool without focal similarity, respectively. Hence, we conclude that ProtoPool with focal similarity generated more salient prototypes than the reference models, including ProtoTree. See Supplementary Materials for more information about a user study, detailed results, and a sample questionnaire.

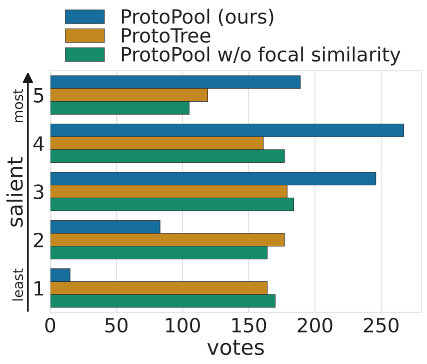


Fig. 10: Distribution of scores from user study on prototypes obtained for ProtoPool without and with focal similarity and for ProtoTree. One can observe that ProtoPool with focal similarity generates more salient prototypes than the other models.

ProtoPool in the context of cognitive psychology ProtoPool can be described in terms of parallel or simultaneous information processing, while ProtoTree may be characterized by serial or successive processing, which takes more time [22,30,35]. More specifically, human cognition is marked with the speed-accuracy trade-off. Depending on the perceptual situation and the goal of a task, the human mind can apply a categorization process (simultaneous or successive) that is the most appropriate in a given context, i.e. the fastest or the most accurate. Both models have their advantages. However, ProtoTree has a specific shortcoming because it allows for a categorization process to rely on an absence of features. In other words, an object characterized by none of the enlisted features is labeled as a member of a specific category. This type of reasoning is useful when the amount of information to be processed (i.e. number of features and categories) is fixed and relatively small. However, the time of object categorization profoundly elongates if the number of categories (and therefore the number of features to be crossed out) is high. Also, the chance of miscategorizing completely new information is increased.

6 Ablation study

In this section, we analyze how the novel architectural choices, the prototype projection, and the number of prototypes influence the model performance.

[8] `https://www.mturk.com`

Table 3: The influence of prototype projection on ProtoPool performance for CUB-200-2011 and Stanford Cars datasets is negligible. Note that for CUB-200-2011, we used ResNet50 pretrained on iNaturalist.

	CUB-200-2011		**Stanford Cars**	
Architecture	Acc [%] before	Acc [%] after	Acc [%] before	Acc [%] after
ResNet34	80.8±0.2	80.3±0.2	89.1±0.2	89.3±0.1
ResNet50	85.9±0.1	85.5±0.1	88.4±0.1	88.9±0.1
ResNet152	81.2±0.2	81.5±0.1	—	—

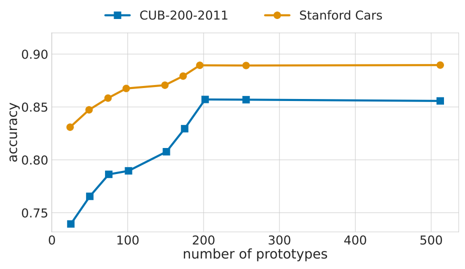
Table 4: The influence of novel architectural choices on ProtoPool performance for CUB-200-2011 and Stanford Cars datasets is significant. We consider training without orthogonalization loss, with softmax instead of Gumbel-Softmax, and with similarity from ProtoPNet instead of focal similarity. One can observe that the mix of the proposed mechanisms (i.e. ProtoPool) obtains the best accuracy.

	CUB-200-2011	**Stanford Cars**
Model	Acc [%]	Acc [%]
ProtoPool	**85.5**	**88.9**
w/o $\mathcal{L}_{orth}$	82.4	86.8
w/o Gumbel-Softmax trick	80.3	64.5
w/o Gumbel-Softmax trick and $\mathcal{L}_{orth}$	65.1	30.8
w/o focal similarity	85.3	88.8

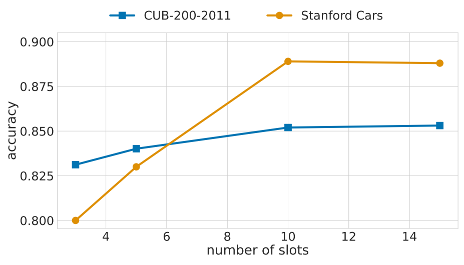
Influence of the novel architectural choices Additionally, we analyze the influence of the novel components we introduce on the final results. For this purpose, we train ProtoPool without orthogonalization loss, with softmax instead of Gumbel-Softmax trick, and with similarity from ProtoPNet instead of focal similarity. Results are presented in Table 4 and in Supplementary Materials. We observe that the Gumbel-Softmax trick has a significant influence on the model performance, especially for the Stanford Cars dataset, probably due to lower inter-class similarity than in CUB-200-2011 dataset [34]. On the other hand, the focal similarity does not influence model accuracy, although as presented in Section 5, it has a positive impact on the interpretability. When it comes to orthogonality, it slightly increases the model accuracy by forcing diversity in slots of each class. Finally, the mix of the proposed mechanisms gets the best results.

Before and after prototype projection Since ProtoPool has much fewer prototypical parts than other models based on a positive reasoning process, applying projection could result in insignificant prototypes and reduced model performance. Therefore, we decided to test model accuracy before and after the projection (see Table 3), and we concluded that differences are negligible.

Number of prototypes and slots vs accuracy Finally, in Figure 11 we investigate how the number of prototypical parts or slots influences accuracy for



(a) Accuracy depending on the number of prototypes. One can observe that the model reaches a plateau for around 200 prototypical parts, and there is no gain in further increase of prototype number.



(b) Accuracy depending on the number of slots. One can observe that the model reaches a plateau for around 10 slots per class.

Fig. 11: ProtoPool accuracy with ResNet50 backbone depending on the number of prototypes and slots for CUB-200-2011 (blue square) and Stanford Cars (orange circle) datasets.

the CUB-200-2011 and Stanford Cars datasets. We observe that up to around 200 prototypical parts, the accuracy increases and reaches the plateau. Therefore, we conclude that the amount of prototypes optimal for ProtoTree is also optimal for ProtoPool. Similarly, in the case of slots, ProtoPool accuracy increases till the 10 slots and then reaches the plateau.

7 Conclusions

We presented ProtoPool, a self-explainable method that incorporates the paradigm of prototypical parts to explain its predictions. This model shares the prototypes between classes without pruning operations, reducing their number up to ten times. Moreover, it is fully differentiable. To efficiently assign the prototypes to classes, we apply the Gumbel-Softmax trick together with orthogonalization loss. Additionally, we introduced focal similarity that focuses on salient features. As a result, we increased the interpretability while maintaining high accuracy, as we showed through theoretical analysis, multiple experiments, and user study.

Acknowledgments

The research of J. Tabor, K. Lewandowska and D. Rymarczyk were carried out within the research project "Bio-inspired artificial neural network" (grant no. POIR.04.04.00-00-14DE/18-00) within the Team-Net program of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund. The work of Ł. Struski and B. Zieliński were supported by the National Centre of Science (Poland) Grant No. 2020/39/D/ST6/01332, and 2021/41/B/ST6/01370, respectively.

References

1. Abbasnejad, E., Teney, D., Parvaneh, A., Shi, J., Hengel, A.v.d.: Counterfactual vision and language learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10044–10054 (2020)
2. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), `https://proceedings.neurips.cc/paper/2018/file/294a8ed24b1ad22ec2e7efea049b8737-Paper.pdf`
3. Afnan, M.A.M., Liu, Y., Conitzer, V., Rudin, C., Mishra, A., Savulescu, J., Afnan, M.: Interpretable, not black-box, artificial intelligence should be used for embryo selection. Human Reproduction Open (2021)
4. Alvarez Melis, D., Jaakkola, T.: Towards robust interpretability with self-explaining neural networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), `https://proceedings.neurips.cc/paper/2018/file/3e9f0fc9b2f89e043bc6233994dfcf76-Paper.pdf`
5. Barnett, A.J., Schwartz, F.R., Tao, C., Chen, C., Ren, Y., Lo, J.Y., Rudin, C.: Iaia-bl: A case-based interpretable deep learning model for classification of mass lesions in digital mammography. arXiv preprint arXiv:2103.12308 (2021)
6. Basaj, D., Oleszkiewicz, W., Sieradzki, I., Górszczak, M., Rychalska, B., Trzcinski, T., Zielinski, B.: Explaining self-supervised image representations with visual probing. In: International Joint Conference on Artificial Intelligence (2021)
7. Brendel, W., Bethge, M.: Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In: International Conference on Learning Representations (2019), `https://openreview.net/forum?id=SkfMWhAqYQ`
8. Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., Su, J.K.: This looks like that: deep learning for interpretable image recognition. In: NeurIPS. pp. 8930–8941 (2019)
9. Chen, Z., Bei, Y., Rudin, C.: Concept whitening for interpretable image recognition. Nature Machine Intelligence **2**(12), 772–782 (2020)
10. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
11. Fong, R., Patrick, M., Vedaldi, A.: Understanding deep networks via extremal perturbations and smooth masks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2950–2958 (2019)
12. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: Proceedings of the IEEE international conference on computer vision. pp. 3429–3437 (2017)
13. Gee, A.H., Garcia-Olano, D., Ghosh, J., Paydarfar, D.: Explaining deep classification of time-series data with learned prototypes. In: CEUR workshop proceedings. vol. 2429, p. 15. NIH Public Access (2019)
14. Ghorbani, A., Wexler, J., Zou, J.Y., Kim, B.: Towards automatic concept-based explanations. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), `https://proceedings.neurips.cc/paper/2019/file/77d2afcb31f6493e350fca61764efb9a-Paper.pdf`
15. Goyal, Y., Wu, Z., Ernst, J., Batra, D., Parikh, D., Lee, S.: Counterfactual visual explanations. In: International Conference on Machine Learning. pp. 2376–2384. PMLR (2019)

16. Guidotti, R., Monreale, A., Matwin, S., Pedreschi, D.: Explaining image classifiers generating exemplars and counter-exemplars from latent representations. Proceedings of the AAAI Conference on Artificial Intelligence **34**(09), 13665–13668 (Apr 2020). https://doi.org/10.1609/aaai.v34i09.7116, `https://ojs.aaai.org/index.php/AAAI/article/view/7116`
17. Hase, P., Chen, C., Li, O., Rudin, C.: Interpretable image recognition with hierarchical prototypes. In: Proceedings of the AAAI Conference on Human Computation and Crowdsourcing. vol. 7, pp. 32–40 (2019)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
19. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
20. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv:1611.01144 (2016)
21. Kaminski, M.E.: The right to explanation, explained. In: Research Handbook on Information Law and Governance. Edward Elgar Publishing (2021)
22. Kesner, R.: A neural system analysis of memory storage and retrieval. Psychological Bulletin **80**(3), 177 (1973)
23. Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In: International conference on machine learning. pp. 2668–2677. PMLR (2018)
24. Kim, E., Kim, S., Seo, M., Yoon, S.: Xprotonet: Diagnosis in chest radiography with global and local explanations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15719–15728 (2021)
25. Koh, P.W., Nguyen, T., Tang, Y.S., Mussmann, S., Pierson, E., Kim, B., Liang, P.: Concept bottleneck models. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 5338–5348. PMLR (13–18 Jul 2020), `https://proceedings.mlr.press/v119/koh20a.html`
26. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 554–561 (2013)
27. Li, O., Liu, H., Chen, C., Rudin, C.: Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
28. Liu, N., Zhang, N., Wan, K., Shao, L., Han, J.: Visual saliency transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4722–4732 (2021)
29. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Proceedings of the 31st international conference on neural information processing systems. pp. 4768–4777 (2017)
30. Luria, A.: The origin and cerebral organization of man's conscious action. Children with learning problems: Readings in a developmental-interaction. New York, Brunner/Mazel pp. 109–130 (1973)
31. Marcos, D., Lobry, S., Tuia, D.: Semantically interpretable activation maps: what-where-how explanations within cnns. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW). pp. 4207–4215. IEEE (2019)

32. Ming, Y., Xu, P., Qu, H., Ren, L.: Interpretable and steerable sequence learning via prototypes. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 903–913 (2019)
33. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency. pp. 607–617 (2020)
34. Nauta, M., et al.: Neural prototype trees for interpretable fine-grained image recognition. In: CVPR. pp. 14933–14943 (2021)
35. Neisser, U.: Cognitive psychology (new york: Appleton). Century, Crofts (1967)
36. Niu, Y., Tang, K., Zhang, H., Lu, Z., Hua, X.S., Wen, J.R.: Counterfactual vqa: A cause-effect look at language bias. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12700–12710 (2021)
37. Puyol-Antón, E., Chen, C., Clough, J.R., Ruijsink, B., Sidhu, B.S., Gould, J., Porter, B., Elliott, M., Mehta, V., Rueckert, D., et al.: Interpretable deep models for cardiac resynchronisation therapy response prediction. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 284–293. Springer (2020)
38. Rebuffi, S.A., Fong, R., Ji, X., Vedaldi, A.: There and back again: Revisiting backpropagation saliency methods. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8839–8848 (2020)
39. Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1135–1144 (2016)
40. Rosch, E.: Cognitive representations of semantic categories. Journal of experimental psychology: General **104**(3), 192 (1975)
41. Rosch, E.H.: Natural categories. Cognitive psychology **4**(3), 328–350 (1973)
42. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**(5), 206–215 (2019)
43. Rymarczyk, D., et al.: Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification. In: SIGKDD. pp. 1420–1430 (2021)
44. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)
45. Selvaraju, R.R., Lee, S., Shen, Y., Jin, H., Ghosh, S., Heck, L., Batra, D., Parikh, D.: Taking a hint: Leveraging explanations to make vision and language models more grounded. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2591–2600 (2019)
46. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. In: In Workshop at International Conference on Learning Representations. Citeseer (2014)
47. Singh, G., Yow, K.C.: These do not look like those: An interpretable deep learning model for image recognition. IEEE Access **9**, 41482–41493 (2021)
48. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning. pp. 3319–3328. PMLR (2017)
49. Van Horn, G., Mac Aodha, O., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P., Belongie, S.: The inaturalist species classification and detection dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8769–8778 (2018)

50. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)
51. Wang, J., et al.: Interpretable image recognition by constructing transparent embedding space. In: ICCV. pp. 895–904 (2021)
52. Wang, P., Vasconcelos, N.: Scout: Self-aware discriminant counterfactual explanations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8981–8990 (2020)
53. Wiegand, G., Schmidmaier, M., Weber, T., Liu, Y., Hussmann, H.: I drive-you trust: Explaining driving behavior of autonomous cars. In: Extended abstracts of the 2019 chi conference on human factors in computing systems. pp. 1–6 (2019)
54. Xiao, T., Xu, Y., Yang, K., Zhang, J., Peng, Y., Zhang, Z.: The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 842–850 (2015)
55. Yeh, C.K., Kim, B., Arik, S., Li, C.L., Pfister, T., Ravikumar, P.: On completeness-aware concept-based explanations in deep neural networks. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 20554–20565. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/ecb287ff763c169694f682af52c1f309-Paper.pdf
56. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.: Protgnn: Towards self-explaining graph neural networks (2022)
57. Zheng, H., Fu, J., Mei, T., Luo, J.: Learning multi-attention convolutional neural network for fine-grained image recognition. In: Proceedings of the IEEE international conference on computer vision. pp. 5209–5217 (2017)
58. Zheng, H., Fu, J., Zha, Z.J., Luo, J.: Looking for the devil in the details: Learning trilinear attention sampling network for fine-grained image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5012–5021 (2019)
59. Zhou, B., Sun, Y., Bau, D., Torralba, A.: Interpretable basis decomposition for visual explanation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 119–134 (2018)

ProGReST: Prototypical Graph Regression Soft Trees for Molecular Property Prediction

Dawid Rymarczyk*† Daniel Dobrowolski* Tomasz Danel*

Abstract

In this work, we propose the novel Prototypical Graph Regression Self-explainable Trees (ProGReST) model, which combines prototype learning, soft decision trees, and Graph Neural Networks. In contrast to other works, our model can be used to address various challenging tasks, including compound property prediction. In ProGReST, the rationale is obtained along with prediction due to the model's built-in interpretability. Additionally, we introduce a new graph prototype projection to accelerate model training. Finally, we evaluate PRoGReST on a wide range of chemical datasets for molecular property prediction and perform in-depth analysis with chemical experts to evaluate obtained interpretations. Our method achieves competitive results against state-of-the-art methods.

Key words. Drug design; Graph Neural Networks; Interpretability; Deep Learning

1 Introduction

In chemistry, the accurate and rapid examination of the compounds is often the key to a successful drug discovery. Searching through millions of compounds, synthesizing them, and evaluating their properties consumes astounding amounts of money and does not guarantee any success at the end of the discovery process. That is why currently *in silico* molecular property prediction is indispensable in modern drug discovery, material design, synthesis planning, etc. Computer methods can accelerate compound screening and mitigate the risk of selecting futile compounds for the *in vitro* examination.

Recent advancements in deep learning, especially in Graph Neural Networks (GNNs), raised the usability of *in vitro* cheminformatics tools to the next level [7]. Tasks such as molecular property prediction, detection of active small molecules, hit identification, and optimization can be accelerated with models such as Molecule Attention Transformer (MAT) [23], DeepGLSTM [26], and Junction Tree Variational Autoencoder (JT-VAE) [15]. Despite the early adoption of artificial intelligence (AI) methods in the drug design process, the initial results are encouraging [22]. Unfortunately, most AI methods do not offer insight into the reasoning behind the decision process.

Due to the complexity of biological systems and drug design processes, insights into the knowledge gathered by the deep learning model are highly sought. Even if the model fails to achieve its goals, the explainability component can hint at the medicinal chemist, e.g. by showing a mechanistic interpretation of the drug action [14]. Most of the current eXplainable Artificial Intelligence (XAI) approaches are post-hoc methods and are applied to already trained models [40]. However, the reliability of those methods is questionable [31]. It assumes that the second model is built to explain an existing trained model. It may result in an unnecessarily increased bias in the explanations, which come from the trained model and the post-hoc model. That is why

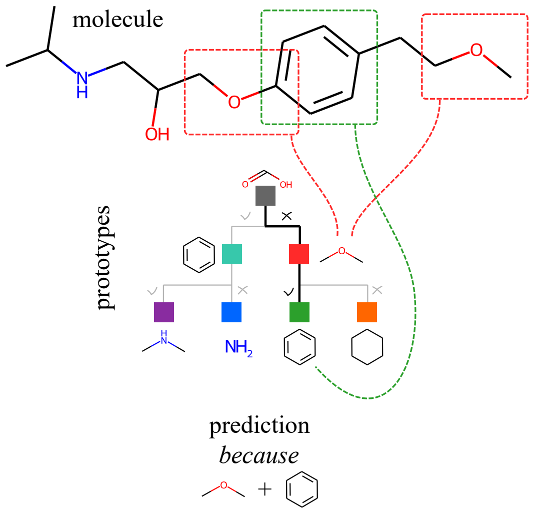


Figure 1: Overview of the ProGReST approach. Molecular substructures are matched against the trained prototypical parts, and the prediction is based on the combination of these features.

*Faculty of Mathematics and Computer Science, Jagiellonian University, Krakow, Poland. (dawid.rymarczyk@student.uj.edu.pl, daniel.dobrowolski@student.uj.edu.pl, tomasz.danel@doctoral.uj.edu.pl)
†Ardigen SA, Krakow, Poland.

self-interpretable models are being developed, such as self-explainable neural networks (SENN) [2] and Prototype Graph Neural Network (ProtGNN) [42]. Only the latter can be applied to the graph prediction problem.

However, ProtGNN is designed for classification problems only since it requires a fixed assignment of prototypes to the classes. While for a regression problem, the model predicts a single label making such an assignment impossible. To overcome the limited applicability of ProtGNN, we introduce the Prototypical Graph Regression Soft Trees (ProGReST) model that is suitable for a graph regression problem, common in the molecular property prediction [38]. It employs prototypical parts (in the paper, we use the terms "prototypical parts" and "prototypes" interchangeably.) [6] that preserve information about activation patterns and ensure intrinsic interpretability (see Fig. 1). Prototypes are derived from the training examples and used to explain the model's decision. To build a model with prototypes, we use Soft Neural Trees [8].

Hence the regression task is more challenging than the classification, it also requires more training epochs for a model to converge. And, prototypical-part-based methods use projection operation periodically [6, 42] to enforce the closeness of prototypes to the training data. In ProtGNN, projection is based on an MCTS algorithm that requires lots of computational time to find meaningful prototypes. In ProGReST, we propose proxy projection to reduce the training time and perform MCTS-based at the end to ensure the full interpretability of the derived prototypes.

The ProGReST achieves state-of-the-art results on five cheminformatics datasets for molecular property prediction and provides intuitive explanations of its prediction in the form of a tree. Also, we confronted the findings of the ProGReST with chemists to validate the usability of our model.

Our contributions can be summarized as follows:

- we introduce ProGReST, a self-explainable prototype-based model for regression of molecular properties,
- we employ a tree-based model to derive meaningful prototypes,
- we define a novel proxy projection function that substantially accelerates the training process.

2 Related Works

2.1 Molecular property prediction The accurate prediction of molecular properties is critical in chemical modeling. In machine learning, chemical compounds can be described using calculated molecular descriptors, which are computed as a function of the compound structure [34]. Many successful applications of machine learning in drug discovery utilize chemical structures directly by employing molecular fingerprints [5] or molecular graphs as an input to the model [9].

Currently, molecular graphs are a preferable representation in cheminformatics because they can capture nonlinear structure of the data. In a molecular graph, atoms are represented as nodes, and the chemical bonds are graph edges. Each atom is attributed with atomic features that encode chemical symbols of the atom and other relevant features [30]. This graphical representation can be processed by graph neural networks that learn the molecule-level vector representation of the compound and use it for property prediction. Graph neural networks usually implement the message passing scheme [10], in which information is passed between nodes along the edges, and the atom features are updated [41]. However, more recent architectures focus on modeling long-range dependencies between atoms, e.g. by implementing graph transformers [24].

2.2 Interpretability of deep learning Methods explaining deep learning models can be divided into the post-hoc and interpretable [31]. The first one creates explainer that reveals the reasoning process of a black box model. Post-hoc methods include: a saliency map [3] that highlights crucial input parts. Another one is Concept Activation Vectors (CAV), that uses concepts to explain the neural network predictions [16]. Implementation of post hoc methods is straightforward since there is no intervention into its architecture. However, they can produce biased and unreliable explanations [1]. That is why more focus is recently on designing self-explainable models [2] to make the decision process directly visible. Recently, a widely used self-explainable model introduced in [6] (ProtoPNet) has a hidden layer of prototypes representing the activation patterns.

Many of the works extended the ProtoPNet, such as TesNet [35] employing Grassman manifold to find prototypes. Also, methods like ProtoPShare [33], ProtoPool [32] and ProtoTree [27] reduce the number of used prototypes. Lastly, those solutions are widely adopted in various fields such as medical imaging [17] and graph classification [42]. Yet, none of these do not consider regression.

3 ProGReST

3.1 Architecture The architecture of ProGReST, depicted in Fig. 2, consists of a graph representation network f, a prototypical regression soft tree layer t and the last layer h. We consider a regression problem

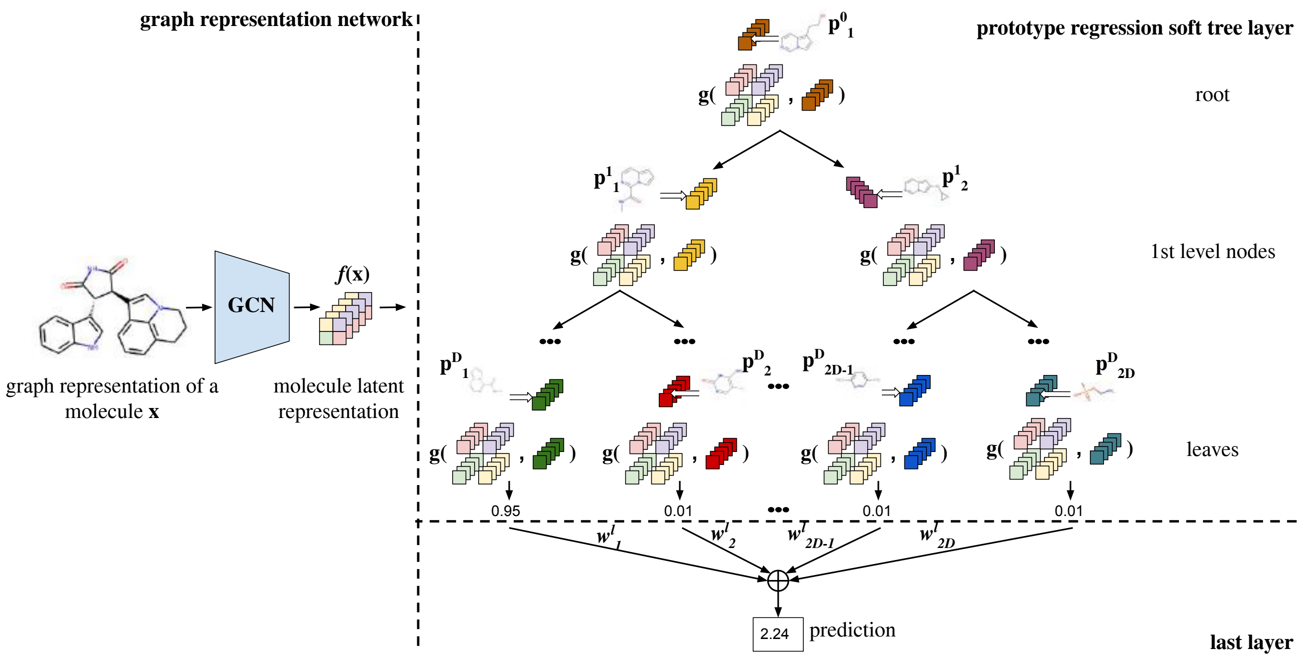

graph representation network
GCN
$f(\mathbf{x})$
graph representation of a molecule $\mathbf{x}$
molecule latent representation
prototype regression soft tree layer
root
1st level nodes
leaves
0.95
0.01
0.01
0.01
2.24
prediction
last layer

Figure 2: ProGReST architecture. It consists of a graph convolutional neural network (GCN) that generates the latent representation of the molecular graph. Later on in the prototype regression, the soft tree layer computes the similarity of each node (prototypical part) to each latent vector of molecular representation. Then the last layer is used to obtain the prediction.

with a dataset consisted of K graphs $x_i \in \mathcal{G}$ with corresponding labels $y_i \in \mathcal{Y} \subset \mathbb{R}$. Graph $\mathcal{G} \subset \mathbb{R}^{\mathcal{N} \times \mathcal{E}}$ contains a set of possible nodes $\mathcal{N}$ and a set of graph edges $\mathcal{E}$. Given an input graph $x_i \in \mathcal{G}$, the model returns its prediction $\hat{y}_i \in \mathcal{Y}$.

Before a molecule is processed by the model, it is encoded as an array of shape $N \times P$, where N is a number of nodes and P is the size of the vector encoding each node. Then, the graph representation network is used to calculate the input embedding $z = f(x)$ where $z \in \mathbb{R}^{N \times C}$, C is the prototype depth, and x is an input graph. The graph representation network is a graph convolutional network (GCN) [19] followed by a node-wise convolution with the sigmoid activation at the end, used to map the input features to the prototype space. The additional node-wise layers reduce the dimensionality of the latent representation and facilitate the learning of meaningful prototypical parts.

The prototype regression soft tree layer contains $2^{\mathcal{D}-1}$ prototypes $p_j \in \mathbb{R}^C$, where $\mathcal{D}$ is the depth of the tree. The number of prototypes is the same as the number of nodes of the tree because there is only one prototypical part in each node. All nodes have two children and the tree contains $2^{\mathcal{D}}$ leaves. Each leaf l_i calculates only one value $y_{l_i} \in \mathbb{R}$. The prototypes are trainable parameters.

For each input $x \in \mathcal{G}$, ProGReST calculates the prototype activation as the similarity between the prototypical part p and the latent graph representation $z_i \in \mathbb{R}^C$ for each graph node i. Then, it calculates the maximum activation as a *presence* of a given prototype in the input graph:

$$\hat{z} = \max_i e^{-||z_i - p||_2} \tag{3.1}$$

Unlike decision trees with nodes routing to only one child, Soft Decision Trees distribute the signal to both children simultaneously, with the probability adding up to 1. To assure that in each node there is a single prototypical part with a similarity value from Equation 3.1 used as a probability of routing to a right node as it is in [27]. The probability of routing to the left node is a complement to 1 and equals $1 - \hat{z}$.

To determine the probability in a leaf l_k, we need to traverse through the path $\mathcal{P}_i$ consisted of its parents:

$$l_k(x) = \prod_{\hat{n} \in \mathcal{P}_k} \hat{z}_{\hat{n}}(x), \tag{3.2}$$

where $\hat{z}_{\hat{n}}$ is the similarity value in node $\hat{n}$. Then, the final prediction is made by summing up probabilities from the leaves multiplied by weights w^l_k of the last layer h:

$$\hat{y}(x) = \sum_{l_k \in \ell} w^l_k \cdot l_k(x). \tag{3.3}$$

3.2 Regularizers Additional regularizers in the loss function of ProGReST are added to ensure that the prototypes and the soft decision tree are effectively learnt.

First of all, we want to minimize the chances of the tree routing only to the left side of the tree. The origin of this problem is in the initialization of the model. It is difficult to activate equally prototypes due to their large distance from the prototypical parts. To mitigate this we implement regularization from [8] that encourages each node to use the left and right subtrees equally. The penalty is the cross entropy between distribution $[0.5, 0.5]$ and the actual average distribution $[\alpha_i, 1 - \alpha_i]$ where α_i for node i is given by:

$$\alpha_i = \frac{\sum_x \Pi_i(x)\hat{z}_i(x)}{\sum_x \Pi_i(x)}, \tag{3.4}$$

where $\Pi_i(x)$ is the path probability from the root to node i. Next, we calculate the weighted sum of the cross entropy, weighted by 2^{-d_i} between each node and balanced distribution. d_i is the depth of node i.

$$L_p = -\sum_{i=1}^{2^{D-1}} 2^{-d_i} \left[0.5 \log(\alpha_i) + 0.5 \log(1 - \alpha_i)\right]. \tag{3.5}$$

As in other prototypical-parts-based models, such as [6, 33, 42], the regularization of a latent space is needed to derive meaningful prototypes. For that purpose we adapt the cluster cost from [17] to assure that the prototypes are close to the parts of the training data points with a batch of B examples.

$$L_c = \frac{1}{|B|} \sum_{x \in B} \min_{p_j \in \mathcal{P}} \min_{z \in f(x)} ||z - p_j||_2. \tag{3.6}$$

The penalty function given by Eq. 3.6 may enforce the prototypical parts to be identical. To prevent that we propose a novel orthogonal loss between parents of each leaf. We penalize the similarity between prototypical parts between parents and not all nodes because some of the prototypes can be reused on a different path. For each leaf we have $\mathcal{D}$ parents. The goal here is to have a have different prototype in each of the node on the given path to maximize the capacity of the model. We propose to minimize the Frobenius norm between the leaf parents.

$$L_d = \frac{1}{|\ell|} \sum_{l \in \ell} \sqrt{\sum_{p_i \in \mathcal{P}_l} \sum_{j > i, p_j \in \mathcal{P}_l} |S_c(p_i, p_j)|^2}, \tag{3.7}$$

where S_c is a cosine similarity. To ensure the majority of probability mass in a single leaf, we use a modified mean squared error (MSE_W) in the training phase. It is needed to assure ease to understand interpretation focused on a single path to a leaf. Given the label y for input x, we define the loss function as follows:

$$MSE_W(y, w^l) = \frac{1}{|\ell|} \sum_{l_k \in \ell} l_k(x) \cdot (y - w^l_k)^2, \tag{3.8}$$

where w^l_k is the weight of a given leaf in the last layer. When the probability in a given node is 1, the prediction itself is w^l_k. So the single path is used to derive the prediction. As a results the final cost function is as follows:

$$L = MSE_W(y, w^l) + \lambda_p L_p + \lambda_c + L_x + \lambda_d L_d, \tag{3.9}$$

where λ_p, λ_c, λ_d are hyper-parameters.

3.3 Proxy projection To assure that prototypes are from the training dataset distribution, we need to use a projection that swaps the prototypical parts with a vector from graph latent representation. In [42], a projection is based on the MCTS algorithm. However, it is done periodically during the training and is very computationally expensive. That is why we use a proxy projection for periodic prototype assignments while at the end of the training we perform the MTCS-based one. Our proxy projection is much faster than MCTS, but also not-interpretable. However, it can be used as an approximation of the exact one.

Our proxy projection tries to find the closest vector from latent graph representation to a given prototypical part and replace it. However, such an assignment makes it non-interpretable since we do not know which nodes with what level of importance contributed to a given representation vector. The proxy projection is defined as follows:

$$p \leftarrow \arg\min_{z \in Z_j} ||z - p_j||_2, \tag{3.10}$$

$$\text{where } Z = \{\hat{z} : \forall i, \hat{z} \in f(x_i)\}.$$

The MCTS-based projection looks for a subgraph of the input graph, which is the closest to the prototype in the latent space [42]. This method can be easily interpreted because two subgraphs shall similarly activate the same prototype. The MCTS-based projection:

$$p_j \leftarrow \arg\min_{z \in Z_j} ||z - p_j||_2, \tag{3.11}$$

$$\text{where } Z_j = \{\hat{z} : \forall i, \text{pool}(f(\hat{x})), \hat{x} \in \text{Subgraph}(x_i)\}.$$

In addition, we reduce the time needed for MCTS-based projection to find the closest subgraph by limiting the search to a single graph. Firstly, with our proxy projection, we identify the graph with the most similar latent vector to a prototype, and then we apply an MCTS-based search on the found graph to identify significant vertices.

3.4 Training schema Due to the more challenging nature of regression comparing to classification [41], we introduce an additional training part called a warmup. During warmup, we firstly cluster the molecules using the K-Means algorithm based on the molecular property values. Then we assign each molecule to a given cluster and we treat it as a new label of a given molecule. Using those pseudo-labels we pre-train full ProGReST using all regularizers with a CrossEntropy Loss L_{CE} using training schema from [27].

At the end of the warmup, each leaf weight w^l is translated into scalars using K-Means centroids. For each leaf l we have:

$$l = \text{Softmax}(l) \cdot \mathcal{K}_{centroids}. \tag{3.12}$$

Similarly to [6], warmup allows the model to derive initial prototypical parts that can be reused in a regression task. After warmup we train ProGReST using loss function from Eq. 3.9. Starting from a given epoch, we periodically perform a proxy projection Eq. 3.10. After the last projection (MCTS-based one) we only train leaves to refine the model. The detailed algorithm for training can be found in Supplementary Materials[1].

4 Experimental Setup

To evaluate the proposed model, we used five datasets from PyTDC [12] that are dedicated to predicting molecular properties. Depending on the dataset, we followed the fold splitting strategy recommended for a given set (scaffold-based split or random split). Each of the datasets was divided into training, validation, and testing sets with the following proportions 80%, 10%, and 10% respectively.

All experiments were implemented in Python 3 and the model is implemented in PyTorch library [29][2]. As a graph representation generation network f, we use graph convolutional network [25] containing 3-5 layers with 128-512 kernels. We used prototypes of depth $C \in \{64, 128, 256\}$. For each dataset, we performed a grid search of hyperparameters of the model. Among the tested parameters were: tree depth $\mathcal{D} \in [4; 7]$, warmup pseudo labels number from 1 to $2^{\mathcal{D}-1}$ so that each class can be in at least one leaf. The warmup lasts up to 80 epochs with an early stopping period of 5 epochs. In the next phase of training, we learned ProGReST for 250 epochs and again used an early stopping mechanism with a window of 10 epochs. Weights of the regularizers from the loss functions were: $\lambda_c \in [0.05, 0.8]$, $\lambda_p \in [0.05, 0.3]$ and $\lambda_d \in [0.0001, 0.01]$. For the MCTS-based projection, we limited the number of iterations to 32. A minimum number of atoms in MCTS-based projection is set to 3 and a maximum to 12. Moreover, MCTS can expand to 12 children. Periodical projection starts at 80^{th} epoch and is performed every 20 epoch. Each model was run 5 times with different seeds. As an optimizer, we use ADAM [18] with a learning rate η different for each node p_j including leaves, $\eta_{p_j} = \eta \cdot 2^{-(\mathcal{D}-d_j)}$. η for encoder e and add-on layer a was calculated by $\eta \cdot 2^{-\mathcal{D}}$. Base η for training was $\eta \in \{0.01, 0.005, 0.001\}$. Such sophisticated learning schedule is caused by the exponential increase of the computational complexity of gradients for the higher prototypes. It is common practice for Soft Decision Tree [8]. Our experiments were performed with NVIDIA RTX 2080 Ti.

5 Experiments

We evaluate ProGReST on five datasets from the PyTDC repository [12]: Caco-2 [36], PPBR [37], LD50 [43], VDss [21], and Half-Life (HL) [28]. In the Supplementary Materials, we provide a short characteristic of the datasets.

Results As Tab. 1 shows, our ProGReST model not only outperforms its baseline model (GCN) but also achieves the best results on four of the five datasets. The prototypical-part-based approach for molecular activity prediction not only brings the interpretability of the predictions into the process, but also achieves superior results. This is in contrast to other prototypical-part-based methods such as [27] in computer vision where the introduction of interpretability reduces the model accuracy.

6 Interpretability

To corroborate the interpretability of the prototypical parts learned by our model, we performed a qualitative study in which a chemist assessed the usefulness of the

[1] `https://arxiv.org/pdf/2210.03745.pdf`

[2] The code is available at `https://github.com/gmum/ProGReST`

Table 1: Results of molecular property prediction. Notice that ProGReST achieves better results than the baseline model (GCN) for each of the datasets. On 4 out of 5 datasets our model achieves the highest effectiveness. We conclude that interpretable molecular prediction can be done without a sacrifice of model performance.

Method	**Caco-2**$\downarrow$	**PPBR**$\downarrow$	**LD50**$\downarrow$	**VDss**$\uparrow$	**HL**$\uparrow$
RDKit2D + MLP [13]	0.393 ± 0.024	9.994 ± 0.319	0.678 ± 0.003	0.561 ± 0.025	0.184 ± 0.111
AttrMasking [11]	0.546 ± 0.052	10.075 ± 0.202	0.685 ± 0.025	0.559 ± 0.019	0.151 ± 0.068
Morgan + MLP [13]	–	12.848 ± 0.362	0.649 ± 0.019	0.493 ± 0.011	0.329 ± 0.083
ContextPred [11]	0.502 ± 0.036	9.445 ± 0.224	0.669 ± 0.030	0.485 ± 0.092	0.129 ± 0.114
NeuralFP [20]	0.530 ± 0.102	9.292 ± 0.384	0.667 ± 0.020	0.258 ± 0.162	0.177 ± 0.165
AttentiveFP [39]	0.401 ± 0.032	9.373 ± 0.335	0.678 ± 0.012	0.241 ± 0.145	0.085 ± 0.068
CNN [13]	0.446 ± 0.036	11.106 ± 0.358	0.675 ± 0.011	0.226 ± 0.114	0.038 ± 0.138
SimGCN [4]	–	–	–	0.582 ± 0.031	$\mathbf{0.392 \pm 0.065}$
ProGReST+GCN (Our)	0.367 ± 0.022	9.722 ± 0.200	0.611 ± 0.009	0.586 ± 0.012	0.295 ± 0.058
GCN (Baseline) [19]	0.599 ± 0.104	10.194 ± 0.373	0.649 ± 0.026	0.457 ± 0.050	0.239 ± 0.100
ProGReST+RMAT (Our)	$\mathbf{0.360 \pm 0.069}$	$\mathbf{9.256 \pm 0.287}$	0.597 ± 0.072	$\mathbf{0.620 \pm 0.069}$	0.337 ± 0.049
RMAT [24]	0.363 ± 0.030	9.909 ± 0.388	$\mathbf{0.569 \pm 0.092}$	0.487 ± 0.083	0.360 ± 0.063

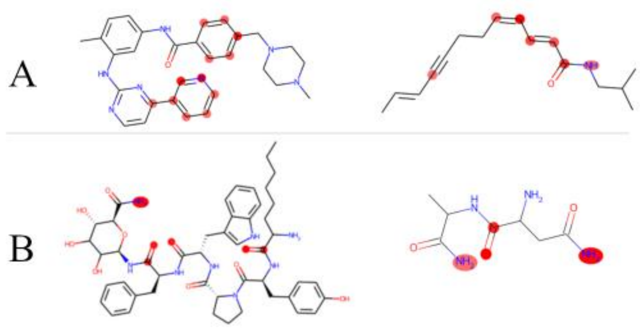


Figure 3: Two examples of the learned prototypical parts. The atoms that are marked with a red circle are a part of the prototype. The compounds on the left are the reference compounds, and the ones on the right are matched by the similarity of the prototypical parts. In prototype A, we see aromatic rings and conjugated bonds (alternating single and double bonds). Prototype B consists of ketones (=O) and amides (-C(=O)NH_2).

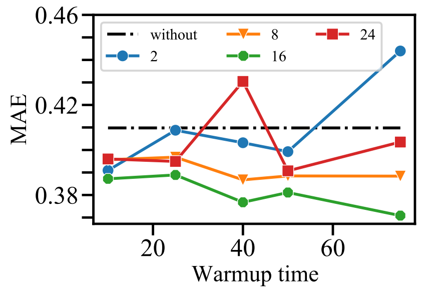
discovered molecular features. First, a small subset of compounds from the Caco-2 dataset was presented to the human expert, and the atoms constituting the learned prototypical parts were highlighted. Next, the compounds that contained the same prototypes were shown in order to confirm the agreement between the similarity function used in the model and the human knowledge-based intuition.

As a result of the visual inspection, a number of useful prototypical parts were identified. Caco-2 is a permeability assay, and there are several prominent molecular features that correlate well with the compound ability to penetrate the epithelial barrier. One of the features detected by the model is a set of ketone and amine groups that impact the hydrophilicity of the compound and can form hydrogen bonds with the lipid layers, which may hugely alter the permeability. In other prototypical parts we see aromatic rings or aliphatic side chains which are also related to the hydrophobicity and can be correlated with the compound bulkiness, decreasing the compound ability to pass the barrier. The described structures are depicted in Fig. 3.

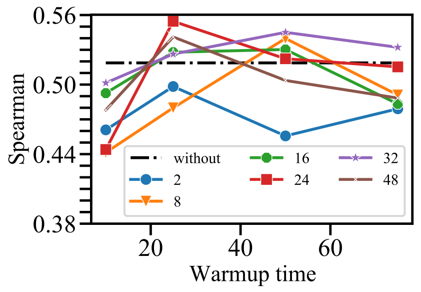
The examination of the prototype similarity between different compounds showed that the same functional groups are correctly matched. What is more interesting, some more subtle similarities are also discovered by the model, e.g. conjugated bonds are found similar to the aromatic rings, indicating that the model captures the electronic nature of these structures. On the right side of Fig. 3, an exemplary matching structure is shown.

7 Ablation study

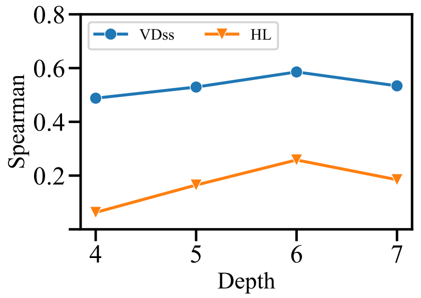
7.1 Warmup In this part, we present the influence of a warmup training stage on the effectiveness of our model. For two of the datasets (Caco2 and VDss), we show how the number of epochs and number of pseudo labels in the warmup phase influence the results, shown in Fig. 4b and Fid. 4a. The results show that too long training is not beneficial for a warmup because the model overfits to labels derived from clustering and forgets features related to a regression task. Also, the number of pseudo-labels varies between datasets, for Caco2 the best one is 16 while for VDss the best is 24. It shows that these parameters should be chosen carefully, most probably due to the relatively small number of



(a) Effect of warming up for the Caco datase. Lower is better



(b) Effect of warming up for the VDss dataset. Higher is better.



(c) Effect of tree depth on model performance. Higher is better.

Figure 4: The plots a) and b) shows how the number of clusters in K-Means in a warmup phase influences the model accuracy. We observe that, $K = 24$ is the most optimal for VDss and $K = 16$ is the most optimal for Caco. The plot c) show effect of tree depth on model performance. We observe that the model performance saturates when the depth of the tree is increased over 6.

Table 2: Comparison of the time needed to perform the projection. We conclude that Proxy Projection is much faster than MCTS-based projection.

Depth	4	5
Proxy Projection	**10.1s**	**16.2s**
MCTS-based Projection	1760.0s	4371.5s

examples in each of the datasets.

7.2 Proxy Projection vs MCTS-based projection In this part of ablation, we want to show a difference in computational time between Proxy and MCTS-based projections. As an MCTS-based one, we take the implementation from [42]. In contrast to ProtGNN, ProGReST can find the important vector of graph latent representation using a similarity function, because we do not perform pooling of a latent representation. Based on Tab. 2, we conclude that MCTS-based projection works much slower than our novel Proxy Projection. That is why we recommend using it in periodical projections. To preserve the interpretability component, we encourage to use MCTS projection at the end of the training.

7.3 Depth of the tree Also, we checked how the depth of a tree influences the model performance. Too small trees don't have enough capacity to correctly model the task. While, too big ones tend to overfit and lost the ability to generalize well. We observed it for each dataset, as it is visible in Fig. 4c.

7.4 Latent distance We checked the latent distance loss proposed in ProtGNN [42]. The authors penalize the model if the distance between all prototypes assigned to a single data class is too small. However, in a regression task there are no classes. We decided to try to apply the proposed loss on all prototypes and it is inconclusive if it helps or not. However, the influence of the loss correlates with the depth of tree. For smaller trees it is beneficial to use this regularizers (for example Caco2 in Tab. 4 has 5 levels), while for deeper trees, the number of nodes grows exponentially. This makes it difficult to ensure orthogonality between all prototypes (for example LD50 in Tab. 4 has 6 levels and the score was worse than the one without the extra loss). That is why we used this loss only for parents of leaves in our tree. We can argue that some structures can exist independently of each other. For example the absence of an aromatic ring does not ensure the absence of any functional structure.

7.5 Influence of regularizers As the training schema of the model is complicated, we tested the influence of each part of the loss function on the final performance of the model. The results are shown in Tab. 3. We notice that cluster cost does not increase the model performance, but it requires less epochs to train. For tested datasets, Caco2 with cluster cost needed 114 ± 8 epochs to get best result, but without it 137 ± 10 epochs, LD50 with needed 184 ± 28, but without 215 ± 19. On the other hand, path cost with λ_p greater than zero improves the model by encouraging the model to use all leaves during the training. The latent distance cost also results in better effectiveness of the model. We conclude that all of those regularizers are complementary and result in a lower error rate of the model.

Table 3: Influence of the different loss components on the model performance. Notice that the regularizers are crucial to training the model. A combination of all regularizers results in the best-performing model.

Dataset	**no regularizer**	**with cluster cost**	**with path cost**	**with latent distance cost**	**all**
Caco2 ↓	0.565 ± 0.003	0.558 ± 0.013	0.454 ± 0.086	0.432 ± 0.034	$\mathbf{0.367 \pm 0.022}$
LD50 ↓	0.660 ± 0.025	0.675 ± 0.004	0.639 ± 0.016	0.643 ± 0.014	$\mathbf{0.611 \pm 0.009}$
PPBR ↓	11.628 ± 0.291	11.272 ± 0.701	11.092 ± 0.771	9.846 ± 0.407	$\mathbf{9.722 \pm 0.200}$
VDss ↑	0.130 ± 0.125	0.180 ± 0.231	0.180 ± 0.151	0.531 ± 0.064	$\mathbf{0.586 \pm 0.012}$
HL ↑	-0031 ± 0.051	0.033 ± 0.134	0.029 ± 0.098	0.193 ± 0.099	$\mathbf{0.295 \pm 0.058}$

Table 4: Different strategies for distance loss. One can observe, that regularizing only parent nodes is the most effective to obtain the best model.

Dataset	**without**	**parents**	**all[42]**
Caco2 ↓	0.402 ± 0.032	$\mathbf{0.367 \pm 0.022}$	0.387 ± 0.030
LD50 ↓	0.625 ± 0.228	$\mathbf{0.611 \pm 0.009}$	0.632 ± 0.025
VDss ↑	0.544 ± 0.051	$\mathbf{0.586 \pm 0.012}$	0.558 ± 0.044
HL ↑	0.239 ± 0.025	$\mathbf{0.295 \pm 0.058}$	0.212 ± 0.053

7.6 Pretrained model As most prototype-based models use pretrained models in their backbones, especially those in the computer vision domain. We investigated how the usage of a pretrained model on a large chemical database behaves in a prototype-based learning scenario. We decided to use R-MAT [23], the current state-of-the-art model for chemical compound representation. We observe the improvement of the results for all datasets, as Tab. 1 shows.

8 Conclusions

In this work, we introduce ProGReST, which is an interpretable model for regression of molecular properties. We show that it not only brings the interpretability into the prediction but also outperforms GCN architecture, as well as achieves state-of-the-art results on 4 out of 5 datasets that we tested. Additionally, we introduce a proxy projection which accelerates the training time and reduces the power consumption needed for training which is important from an environmental point of view. Finally, we show that the ProGReST explanations are valid and show the influence of the novelties and their hyperparameters on the models' performance.

In future works, we want to analyze the possibility of pruning for ProGReST and generalize it to other data types such as text, because in NLP there are a lot of problems represented as graphs and regression tasks.

Acknowledgments

The work of D. Dobrowolski is supported by the National Centre of Science (Poland) Grant No. 2021/41/B/ST6/01370. The work of D. Rymarczyk was funded by National Centre of Science (Poland) Grant No. 2022/45/N/ST6/04147. The work of T. Danel is supported by National Centre of Science (Poland) Grant No. 2020/37/N/ST6/02728.

References

[1] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, *Sanity checks for saliency maps*, in NeurIPS, 2018.

[2] D. Alvarez Melis and T. Jaakkola, *Towards robust interpretability with self-explaining neural networks*, in NeurIPS, 2018.

[3] F. Baldassarre and H. Azizpour, *Explainability techniques for graph convolutional networks*, arXiv, (2019).

[4] S. K. Bera, J. Dent, G. Gurbinder, A. Stoleman, and B. Wu, *Simgcn for tdc benchmarks*, (2022).

[5] A. Capecchi, D. Probst, and J.-L. Reymond, *One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome*, Journal of cheminformatics, 12 (2020), pp. 1–15.

[6] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, *This looks like that: deep learning for interpretable image recognition*, NeurIPS, (2019).

[7] L. David, A. Thakkar, R. Mercado, and O. Engkvist, *Molecular representations in ai-driven drug discovery: a review and practical guide*, Journal of Cheminformatics, 12 (2020), pp. 1–22.

[8] N. Frosst and G. Hinton, *Distilling a neural network into a soft decision tree*, arXiv, (2017).

[9] T. Gaudelet, B. Day, A. R. Jamasb, et al., *Utilizing graph machine learning within drug discovery and development*, Briefings in bioinformatics, 22 (2021).

[10] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Neural message passing for quantum chemistry*, in International conference on machine learning, 2017.

[11] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, *Strategies for pre-training graph neural networks*, arXiv, (2019).

[12] K. Huang, T. Fu, W. Gao, Y. Zhao, Y. Roohani, J. Leskovec, C. W. Coley, C. Xiao, J. Sun, and M. Zitnik, *Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development*, NeurIPS Datasets and Benchmarks, (2021).

[13] K. Huang, T. Fu, L. M. Glass, M. Zitnik, C. Xiao, and J. Sun, *Deeppurpose: a deep learning library for drug–target interaction prediction*, Bioinformatics, 36 (2020), pp. 5545–5547.

[14] J. Jiménez-Luna, F. Grisoni, and G. Schneider, *Drug discovery with explainable artificial intelligence*, Nature Machine Intelligence, 2 (2020), pp. 573–584.

[15] W. Jin, R. Barzilay, and T. Jaakkola, *Junction tree variational autoencoder for molecular graph generation*, in ICML, 2018.

[16] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al., *Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)*, in ICML, 2018.

[17] E. Kim, S. Kim, M. Seo, and S. Yoon, *Xprotonet: diagnosis in chest radiography with global and local explanations*, in CVPR, 2021.

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

[19] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, arXiv, (2016).

[20] W.-H. Lee, S. Millman, N. Desai, et al., *Neuralfp: Out-of-distribution detection using fingerprints of neural networks*, in ICPR, 2021.

[21] F. Lombardo and Y. Jing, *In silico prediction of volume of distribution in humans. extensive data set and the exploration of linear and nonlinear methods coupled with molecular interaction fields descriptors*, Journal of Chemical Information and Modeling, 56 (2016), pp. 2042–2052.

[22] K.-K. Mak, M. K. Balijepalli, and M. R. Pichika, *Success stories of ai in drug discovery-where do things stand?*, Expert opinion on drug discovery, 17 (2022).

[23] Ł. Maziarka, T. Danel, S. Mucha, K. Rataj, J. Tabor, and S. Jastrzebski, *Molecule attention transformer*, arXiv, (2020).

[24] Ł. Maziarka, D. Majchrowski, T. Danel, P. Gaiński, J. Tabor, I. Podolak, P. Morkisz, and S. Jastrzebski, *Relative molecule self-attention transformer*, arXiv, (2021).

[25] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, *Weisfeiler and leman go neural: Higher-order graph neural networks*, in AAAI, 2019.

[26] S. Mukherjee, M. Ghosh, and P. Basuchowdhuri, *Deepglstm: Deep graph convolutional network and lstm based approach for predicting drug-target binding affinity*, in SDM, 2022.

[27] M. Nauta, R. van Bree, and C. Seifert, *Neural prototype trees for interpretable fine-grained image recognition*, in CVPR, 2021.

[28] R. S. Obach, F. Lombardo, and N. J. Waters, *Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 670 drug compounds*, Drug Metabolism and Disposition, (2008).

[29] A. Paszke, S. Gross, F. Massa, et al., *Pytorch: An imperative style, high-performance deep learning library*, NeurIPS, (2019).

[30] A. Pocha, T. Danel, S. Podlewska, J. Tabor, and Ł. Maziarka, *Comparison of atom representations in graph neural networks for molecular property prediction*, in IJCNN, 2021.

[31] C. Rudin, *Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead*, Nature Machine Intelligence, (2019).

[32] D. Rymarczyk, Ł. Struski, M. Górszczak, K. Lewandowska, J. Tabor, and B. Zieliński, *Interpretable image classification with differentiable prototypes assignment*, in ECCV, 2022.

[33] D. Rymarczyk, Ł. Struski, J. Tabor, and B. Zieliński, *Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification*, in SIGKDD, 2021.

[34] R. Todeschini and V. Consonni, *Handbook of molecular descriptors*, John Wiley & Sons, 2008.

[35] J. Wang, H. Liu, X. Wang, and L. Jing, *Interpretable image recognition by constructing transparent embedding space*, in ICCV, 2021.

[36] N.-N. Wang, J. Dong, Y.-H. Deng, et al., *Adme properties evaluation in drug discovery: Prediction of caco-2 cell permeability using a combination of nsga-ii and boosting*, Journal of Chemical Information and Modeling, (2016).

[37] M. Wenlock and N. Tomkinson, *Experimental in vitro dmpk and physicochemical data on a set of publicly disclosed compounds*.

[38] O. Wieder, S. Kohlbacher, M. Kuenemann, et al., *A compact review of molecular property prediction with graph neural networks*, Drug Discovery Today: Technologies, (2020).

[39] Z. Xiong et al., *Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism*, Journal of Medicinal Chemistry, (2020).

[40] Z. Ying et al., *Gnnexplainer: Generating explanations for graph neural networks*, NeurIPS, (2019).

[41] S. Zhang et al., *Graph convolutional networks: a comprehensive review*, Computational Social Networks, (2019).

[42] Z. Zhang et al., *Protgnn: Towards self-explaining graph neural networks*, AAAI, (2022).

[43] H. Zhu et al., *Quantitative structure-activity relationship modeling of rat acute toxicity by oral exposure*, Chemical Research in Toxicology, (2009).

Kernel Self-Attention for Weakly-supervised Image Classification using Deep Multiple Instance Learning

Dawid Rymarczyk [1,2,*], Adriana Borowa[1, 2,*], Jacek Tabor[1,**], and Bartosz Zieliński[1, 2,**]

[1]Faculty of Mathematics and Computer Science, Jagiellonian University, 6 Łojasiewicza Street, 30-348 Kraków, Poland,
[2]Ardigen SA, 76 Podole Street, 30-394 Kraków, Poland,

[*]{dawid.rymarczyk,ada.borowa}@student.uj.edu.pl
[**]{jacek.tabor,bartosz.zielinski}@uj.edu.pl

Abstract

Not all supervised learning problems are described by a pair of a fixed-size input tensor and a label. In some cases, especially in medical image analysis, a label corresponds to a bag of instances (e.g. image patches), and to classify such bag, aggregation of information from all of the instances is needed. There have been several attempts to create a model working with a bag of instances, however, they are assuming that there are no dependencies within the bag and the label is connected to at least one instance. In this work, we introduce Self-Attention Attention-based MIL Pooling (SA-AbMILP) aggregation operation to account for the dependencies between instances. We conduct several experiments on MNIST, histological, microbiological, and retinal databases to show that SA-AbMILP performs better than other models. Additionally, we investigate kernel variations of Self-Attention and their influence on the results.

1. Introduction

Classification methods typically assume that there exists a separate label for each example from a dataset. However, in many real-life applications, there exists only one label for a bag of instances because it is too laborious to label all of them separately. This type of problem, called Multiple Instance Learning (MIL) [7], assumes that there is only one label provided for the entire bag and that some of the instances associate to this label [9].

MIL problems are common in medical image analysis due to the vast resolution of images and the weakly-labeled small datasets [4, 21]. Among others, they appear in the whole slide-image classification of biopsies [2, 3, 34], classification of dementia based on brain MRI [28], or the diabetic retinopathy screening [22, 23]. They are also used in computer-aided drug design to identify which conformers are responsible for the molecule activity [26, 36].

Recently, Ilse *et al*. [13] introduced the Attention-based MIL Pooling (AbMILP), a trainable operator that aggregates information from multiple instances of a bag. It is based on a two-layered neural network with the attention weights, which allows finding essential instances. Since the publication, this mechanism was widely adopted in the medical image analysis [18, 20, 32], especially for the assessment of whole-slide images. However, the Attention-based MIL Pooling is significantly different from the Self-Attention (SA) mechanism [35]. It perfectly aggregates information from a varying number of instances, but it does not model dependencies between them. Additionally, the SA-AbMILP is distinct from other MIL approaches developed recently because it is not modeling the bag as a graph like in [30, 33], it is not using the pre-computed image descriptors as features as in [30], and it models the dependencies between instances in contrast to [8].

In this work, we introduce a method that combines self-attention with Attention-based MIL Pooling. It simultaneously catches the global dependencies between the instances in the bag (which are beneficial [37]) and aggregates them into a fixed-sized vector required for the successive layers of the network, which then can be used in regression, binary, and multi-class classification problems. Moreover, we investigate a broad spectrum of kernels replacing dot product when generating an attention map. According to the experiments' results, using our method with various kernels is beneficial compared to the baseline approach, especially in the case of more challenging MIL assumptions. Our code is publicly available at `https:`

//github.com/gmum/Kernel_SA-AbMILP.

2. Multiple Instance Learning

Multiple Instance Learning (MIL) is a variant of inductive machine learning belonging to the supervised learning paradigm [9]. In a typical supervised problem, a separate feature vector, e.g. ResNet representation after Global Max Pooling, exists for each sample: $\mathbf{x} = \mathbf{h}, \mathbf{h} \in \mathbb{R}^{L \times 1}$. In MIL, each example is represented by a bag of feature vectors of length L called instances: $\mathbf{x} = \{\mathbf{h_i}\}_{i=1}^{n}, \mathbf{h_i} \in \mathbb{R}^{L \times 1}$, and the bag is of variable size n. Moreover, in *standard MIL assumption*, label of the bag $\mathbf{y} \in \{0, 1\}$, each instance h_i has a hidden binary label $y_i \in \{0, 1\}$, and the bag is positive if at least one of its instances is positive:

$$\mathbf{y} = \begin{cases} 0, & \text{iff } \sum_{i=1}^{n} y_i = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

This standard assumption (considered by AbMILP) is stringent and hence does not fit to numerous real-world problems. As an example, let us consider the digestive track assessment using the NHI scoring system [19], where the score 2 is assigned to a biopsy if the neutrophils infiltrate more than 50% of crypts and there is no epithelium damage or ulceration. Such a task obviously requires more challenging types of MIL [9], which operate on many assumptions (below defined as concepts) and classes.

Let $\hat{C} \subseteq C$ be the set of required instance-level concepts, and let $p : X \times C \to K$ be the function that counts how often the concept $c \in C$ occurs in the bag $\mathbf{x} \in X$. Then, in *presence-based assumption*, the bag is positive if each concept occurs at least once:

$$\mathbf{y} = \begin{cases} 1, & \text{iff for each } c \in \hat{C} : p(\mathbf{x}, c) \geq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In the case of *threshold-based assumptions*, the bag is positive if concept $c_i \in C$ occurs at least $t_i \in \mathbb{N}$ times:

$$\mathbf{y} = \begin{cases} 1, & \text{iff for each } c_i \in \hat{C} : p(\mathbf{x}, c_i) \geq t_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

In this paper, we introduce methods suitable not only for the standard assumption (like it was in the case of AbMILP) but also for presence-based and threshold-based assumptions.

3. Methods

3.1. Attention-based Multiple Instance Learning Pooling

Attention-based MIL Pooling (AbMILP) [13] is a type of weighted average pooling, where the neural network determines the weights of instances. More formally, if the bag $\mathbf{x} = \{\mathbf{h_i}\}_{i=1}^{n}, \mathbf{h_i} \in \mathbb{R}^{L \times 1}$, then the output of the operator is defined as:

$$\mathbf{z} = \sum_{i=1}^{n} a_i \mathbf{h_i}, \text{ where } a_i = \frac{\exp\left(\mathbf{w}^T tanh(\mathbf{V}\mathbf{h_i})\right)}{\sum_{j}^{N} \exp\left(\mathbf{w}^T tanh(\mathbf{V}\mathbf{h_j})\right)}, \tag{4}$$

$\mathbf{w} \in \mathbb{R}^{M \times 1}$ and $\mathbf{V} \in \mathbb{R}^{M \times L}$ are trainable layers of neural networks, and the hyperbolic tangent prevents the exploding gradient. Moreover, the weights a_i sum up to 1 to wean from various sizes of the bags and the instances are in the random order within the bag to prevent the overfitting.

The most important limitation of AbMILP is the assumption that all instances of the bag are independent. To overcome this limitation, we extend it by introducing the Self-Attention (SA) mechanism [35] which models dependencies between instances of the bag.

3.2. Self-Attention in Multiple Instance Learning

The pipeline of our method, which applies Self-Attention into Attention-based MIL Pooling (SA-AbMILP), consists of four steps. First, the bag's images are passed through the Convolutional Neural Network (CNN) to obtain their representations. Those representations are used by the self-attention module (with dot product or the other kernels) to integrate dependencies of the instances into the process. Feature vectors with integrated dependencies are used as the input for the AbMILP module to obtain one fixed-sized vector for each bag. Such a vector can be passed to successive Fully-Connected (FC) layers of the network. The whole pipeline is presented in Fig. 1. In order to make this work self-contained, below, we describe self-attention and particular kernels.

Self-Attention (SA). SA is responsible for finding the dependencies between instances within one bag. Instance representation after SA is enriched with the knowledge from the entire bag, this is important for the detection of the number of instances of the same concept and their relation. SA transforms all the instances into two feature spaces of keys $\mathbf{k_i} = \mathbf{W_k}\mathbf{h_i}$ and queries $\mathbf{q_j} = \mathbf{W_q}\mathbf{h_j}$, and calculates:

$$\beta_{\mathbf{j},\mathbf{i}} = \frac{\exp\left(s_{ij}\right)}{\sum_{i=1}^{N} \exp\left(s_{ij}\right)}, \text{ where } s_{ij} = \langle \mathbf{k}(\mathbf{h_i}), \mathbf{q}(\mathbf{h_j}) \rangle, \tag{5}$$

to indicate the extent to which the model attends to the i^{th} instance when synthesizing the j^{th} one. The output of the attention layer is defined separately for each instance as:

$$\hat{\mathbf{h}}_{\mathbf{j}} = \gamma \mathbf{o_j} + \mathbf{h_j}, \text{ where } \mathbf{o_j} = \sum_{i=1}^{N} \beta_{j,i} \mathbf{W_v} \mathbf{h_i}, \tag{6}$$

$\mathbf{W_q}, \mathbf{W_k} \in \mathbb{R}^{\bar{L} \times L}$, $\mathbf{W_v} \in \mathbb{R}^{L \times L}$ are trainable layers, $\bar{L} = L/8$, and γ is a trainable scalar initialized to 0. Parameters $\bar{L}$ and γ were chosen based on the results presented in [35].

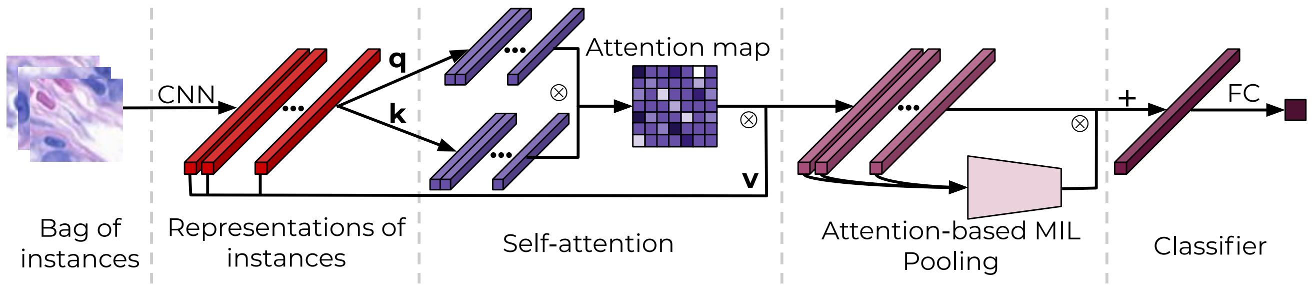


Figure 1: The pipeline of self-attention in deep MIL starts with obtaining feature space representation for each of the instances from the bag using features block of Convolutional Neural Network (CNN). In order to model dependencies between the instances, their representations pass trough the self-attention layer and then aggregate using AbMILP operator. The obtained fixed-size vector goes trough the Fully Connected (FC) classification layer.

Kernels in self-attention. In order to indicate to which extent one instance attends on synthesizing the other one, the self-attention mechanism typically employs a dot product (see s_{ij} in Eq. 5). However, dot product can be replaced by a various kernel with positive results observed in Support Vectors Machine (SVM) [1] or Convolutional Neural Networks (CNN) [31], especially in the case of small training sets.

The Radial Basis Function (RBF) and Laplace kernels were already successfully adopted to self-attention [14, 29]. Hence, in this study, we additionally extend our approach with the following standard choice of kernels (with α as a trainable parameter):

- Radial Basis Function (GSA-AbMILP):
  $k(x, y) = \exp\left(-\alpha\|x - y\|_2^2\right)$,
- Inverse quadratic (IQSA-AbMILP):
  $k(x, y) = \frac{1}{\alpha\|x-y\|_2^2+1}$,
- Laplace (LSA-AbMILP):
  $k(x, y) = -\|x - y\|_1$,
- Module (MSA-AbMILP):
  $k(x, y) = \|x - y\|^\alpha - \|x\|^\alpha - \|y\|^\alpha$.

We decided to limit to those kernels because they are complementary regarding the shape of tails in their distributions.

4. Experiments

We adopt five datasets, from which four were adopted to MIL algorithms [13, 30], to investigate the performance of our method: MNIST [17], two histological databases of colon [25] and breast [10] cancer, a microbiological dataset DIFaS [38], and a diabetic retinopathy screening data set called “Messidor” [5]. For MNIST, we adapt LeNet5 [17] architecture, for both histological datasets SC-CNN [25] is applied as it was in [13], for microbiological dataset we use convolutional parts of ResNet-18 [12] or AlexNet [16] followed by 1×1 convolution as those were the best feature extractor in [38], and for the "Messidor" dataset we used the ResNet-18 [12] as most of the approaches that we compare here are based on the handcrafted image features like in [30]. The experiments, for MNiST, histological and "Messidor" datasets, are repeated 5 times using 10 fold cross-validation with 1 validation fold and 1 test fold. In the case of the microbiological dataset, we use the original 2 fold cross-validation which divides the images by the preparation, as using images from the same preparation in both training and test set can result in overstated accuracy [38]. Due to the dataset complexity, we use the early stopping mechanism with different windows: 5, 25, 50 and 70 epochs for MNIST, histological datasets, microbiological, and "Messidor" datasets, respectively. We compare the performance of our method (SA-AbMILP) and its kernel variations with instance-level approaches (instance+max, instance+mean, and instance+voting), embedding-level approaches (embedding+max and embedding+mean), and Attention-based MIL Pooling, AbMILP [13]. The instance-level approaches in the case of MNIST and histological database compute the maximum or mean value of the instance scores. For the microbiological database, instance scores are aggregated by voting due to multiclassification. The embedding-level approaches calculate the maximum or mean for feature vector of the instances. For "Messidor" dataset we are comparing to the results obtained by [30, 8]. We run a Wilcoxon signed-rank test on the results to identify which ones significantly differ from each other, and which ones do not (and thus can be considered equally good). The comparison is performed between the best method (the one with the best mean score) and all the other methods, for each experiment separately. The mean accuracy is obtained as average over 5 repetitions with the same train/test divisions used by all compared methods. The number of repetitions is relatively small for statistical tests. Therefore we set the p-value to 0.1. For computations, we use Nvidia GeForce RTX 2080.

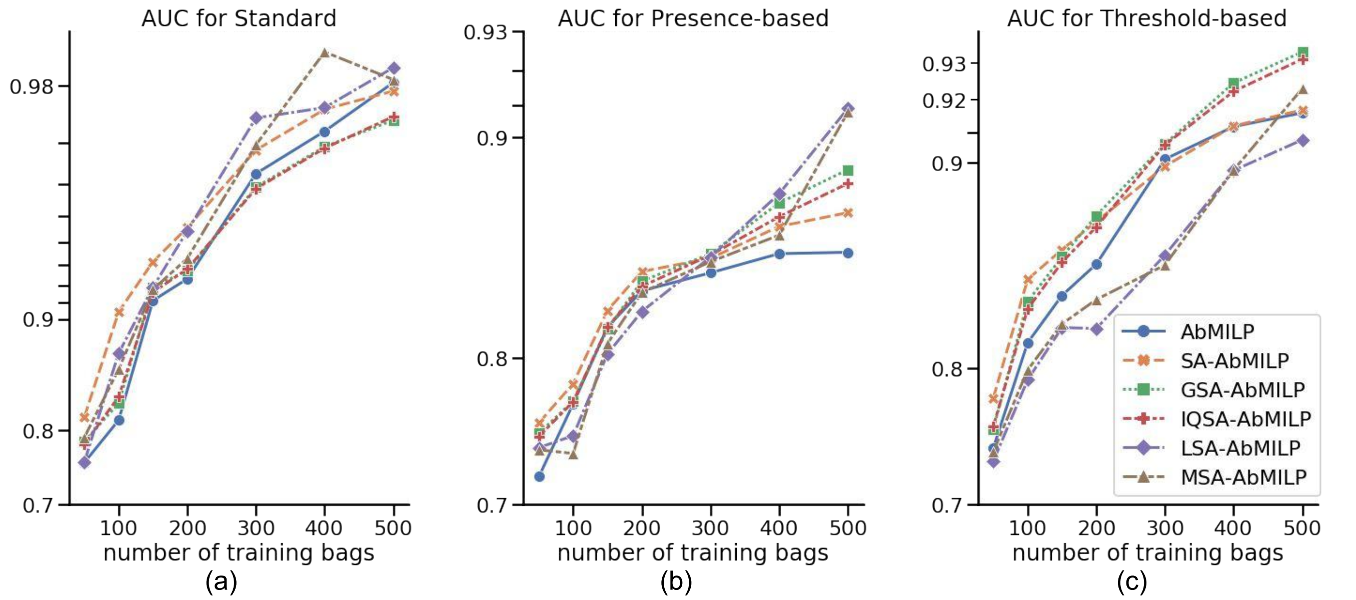


Figure 2: Results for MNIST dataset with bags generated using standard (a), presence-based (b), and threshold-based (c) assumption. In all cases, our approach, either with dot product (SA-AbMILP) or the other kernels (GSA-AbMILP, IQSA-AbMILP, LSA-AbMILP, and MSA-AbMILP) obtains statistically better results than the baseline method (AbMILP). See Section 3 for description of the shortcuts.

4.1. MNIST dataset

Experiment details. As in [13], we first construct various types of bags based on the MNIST dataset. Each bag contains a random number of MNIST images (drawn from Gaussian distributions $\mathcal{N}(10, 2)$). We adopt three types of bag labels referring to three types of MIL assumptions:

- Standard assumptions: $\mathbf{y} = 1$ if there is at least one occurrence of “9”,
- Presence-based assumptions: $\mathbf{y} = 1$ if there is at least one occurrence of “9” and at least one occurrence of “7”,
- Threshold-based assumptions: $\mathbf{y} = 1$ if there are at least two occurrences of “9”.

We decided to use “9” and “7” because they are often confused with each other, making the task more challenging.

We investigated how the performance of the model depends on the number of bags used in training (we consider 50, 100, 150, 200, 300, 400, and 500 training bags). For all experiments, we use LeNet5 [17] initialized according to [11] with the bias set to 0. We use Adam optimizer [15] with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate 10^{-5}, and batch size 1.

Results. AUC values for considered MIL assumptions are visualized in Fig. 2. One can observe that our method with a dot product (SA-AbMILP) always outperforms other methods in case of small datasets. However, when the number of training examples reaches 300, its kernel extensions work better (Laplace in presence-based and inverse quadratic in threshold-based assumption). Hence, we conclude that for small datasets, no kernel extensions should be applied, while in the case of a larger dataset, a kernel should be optimized together with the other hyperparameters. Additionally, we analyze differences between the weights of instances in AbMILP and our method. As presented in Fig. 3, for our method, “9”s and “7”s strengthen each other in the self-attention module, resulting in higher weights in the aggregation operator than for AbMILP, which returns high weight for only one digit (either “9” or “7”).

4.2. Histological datasets

Experiment details. In the second experiment, we consider two histological datasets of *breast* and *colon* cancer (described below). For both of them, we generate instance representations using SC-CNN [25] initialized according to [11] with the bias set to 0. We use Adam [15] optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate 10^{-4}, and batch size 1. We also apply extensive data augmentation, including random rotations, horizontal and vertical flipping, random staining augmentation [13], staining normalization [27], and instance normalization.

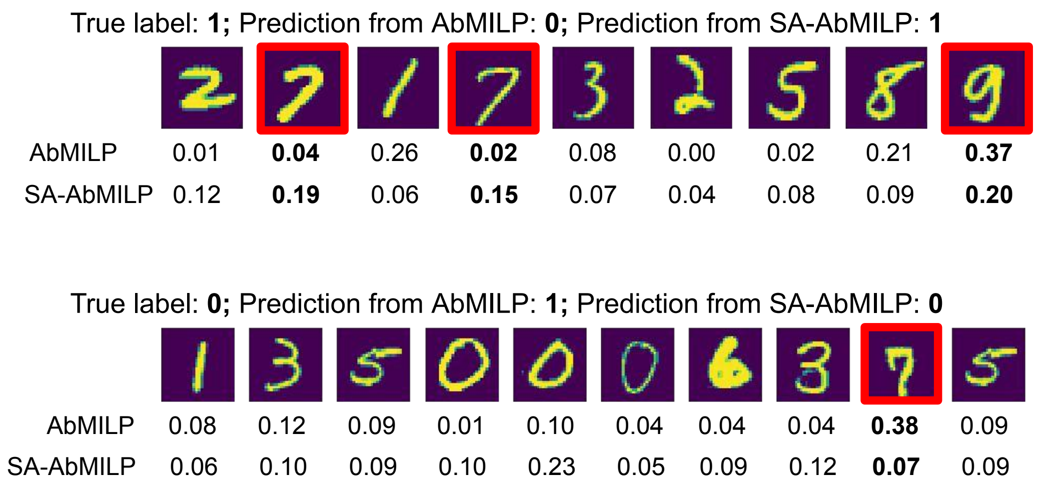


Figure 3: Example of instances' weights for a positive (top) and negative (bottom) bag in a presence-based assumption (where positive is at least one occurrence of "9" and "7") for AbMILP and our method. One can observe that for SA-AbMILP, "9"s and "7"s strengthen each other in the self-attention module, resulting in higher weights in the aggregation operator than for AbMILP.

Breast cancer dataset. Dataset from [10] contains 58 weakly labeled H&E biopsy images of resolution 896×768. The image is labeled as malignant if it contains at least one cancer cell. Otherwise, it is labeled as benign. Each image is divided into patches of resolution 32×32, resulting in 672 patches per image. Patches with at least 75% of the white pixels are discarded, generating 58 bags of various sizes.

Colon cancer dataset. Dataset from [25] contains 100 images with 22444 nuclei manually assigned to one of the following classes: epithelial, inflammatory, fibroblast, and miscellaneous. We construct bags of 27×27 patches with centers located in the middle of the nuclei. The bag has a positive label if there is at least one epithelium nucleus in the bag. Tagging epithelium nuclei is essential in the case of colon cancer because the disease originates from them [24].

Results. Results for histological datasets are presented in Table 1. For both of them, our method (with or without kernel extension) improves the Area Under the ROC Curve (AUC) comparing to the baseline methods. Moreover, our method obtains the highest recall, which is of importance for reducing the number of false negatives. To explain why our method surpasses the AbMILP, we compare the weights of patches in the average pooling. Those patches contribute the most to the final score and should be investigated by the pathologists. One can observe in Fig. 4 that our method highlights fewer patches than AbMILP, which simplifies their analysis. Additionally, SA dependencies obtained for the most relevant patch of our method are justified histologically, as they mostly focus on nuclei located in the neighborhood of crypts. Moreover, in the case of the colon cancer dataset, we further observe the positive aspect of our method, as it strengthens epithelium nuclei and weakens nuclei in the lamina propria at the same time,. Finally, we notice that kernels often improve overall performance but none of them is significantly superior.

4.3. Microbiological dataset

Experiment details. In the final experiment, we consider the microbiological DIFaS database [38] of *fungi species*. It contains 180 images for 9 fungi species (there are 2 preparations with 10 images for each species and it is a multi-class classification). As the size of images is $5760 \times 3600 \times 3$ pixels, it is difficult to process the entire image through the network so we generate patches following method used in [38]. Hence, unlike the pipeline from Section 3.2, we use two separate networks. The first network generates the representations of the instances, while the second network (consisting of self-attention, attention-based MIL pooling, and classifier) uses those representations to recognize fungi species. Due to the separation, it is possible to use deep architectures like ResNet-18 [12] and AlexNet [16] pre-trained on ImageNet database [6] to generate the representations. The second network is trained using Adam [15] optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate 10^{-5},

Table 1: Results for breast and colon cancer datasets (mean and standard error of the mean over 5 repetitions). See Section 3 for description of the acronyms.

breast cancer dataset					
method	accuracy	precision	recall	F-score	AUC
instance+max	61.4 ± 2.0	58.5 ± 3.0	47.7 ± 8.7	50.6 ± 5.4	61.2 ± 2.6
instance+mean	67.2 ± 2.6	67.2 ± 3.4	51.5 ± 5.6	57.7 ± 4.9	71.9 ± 1.9
embedding+max	60.7 ± 1.5	55.8 ± 1.3	54.6 ± 7.0	54.3 ± 4.2	65.0 ± 1.3
embedding+mean	$\mathbf{74.1 \pm 2.3}$	74.1 ± 2.3	65.4 ± 5.4	$\mathbf{68.9 \pm 3.4}$	79.6 ± 1.2
AbMILP	71.7 ± 2.7	$\mathbf{77.1 \pm 4.1}$	68.6 ± 3.9	66.5 ± 3.1	85.6 ± 2.2
SA-AbMILP	$\mathbf{75.1 \pm 2.4}$	$\mathbf{77.4 \pm 3.7}$	74.9 ± 3.7	$\mathbf{69.9 \pm 3.0}$	$\mathbf{86.2 \pm 2.2}$
GSA-AbMILP	$\mathbf{75.8 \pm 2.2}$	$\mathbf{79.3 \pm 3.3}$	74.7 ± 3.4	$\mathbf{72.5 \pm 2.5}$	$\mathbf{85.9 \pm 2.2}$
IQSA-AbMILP	$\mathbf{76.7 \pm 2.3}$	$\mathbf{78.6 \pm 4.2}$	75.1 ± 4.2	$\mathbf{66.6 \pm 4.3}$	$\mathbf{85.9 \pm 2.2}$
LSA-AbMILP	65.5 ± 2.9	62.5 ± 3.7	$\mathbf{89.5 \pm 2.6}$	$\mathbf{68.5 \pm 2.6}$	$\mathbf{86.7 \pm 2.1}$
MSA-AbMILP	$\mathbf{73.8 \pm 2.6}$	$\mathbf{78.4 \pm 3.9}$	73.8 ± 3.6	$\mathbf{69.4 \pm 3.4}$	85.8 ± 2.2
colon cancer dataset					
method	accuracy	precision	recall	F-score	AUC
instance+max	84.2 ± 2.1	86.6 ± 1.7	81.6 ± 3.1	83.9 ± 2.3	91.4 ± 1.0
instance+mean	77.2 ± 1.2	82.1 ± 1.1	71.0 ± 3.1	75.9 ± 1.7	86.6 ± 0.8
embedding+max	82.4 ± 1.5	88.4 ± 1.4	75.3 ± 2.0	81.3 ± 1.7	91.8 ± 1.0
embedding+mean	86.0 ± 1.4	81.1 ± 1.1	80.4 ± 2.7	85.3 ± 1.6	94.0 ± 1.0
AbMILP	88.4 ± 1.4	$\mathbf{95.3 \pm 1.5}$	$\mathbf{84.1 \pm 2.9}$	$\mathbf{87.2 \pm 2.1}$	97.3 ± 0.7
SA-AbMILP	$\mathbf{90.8 \pm 1.3}$	$\mathbf{93.8 \pm 2.0}$	$\mathbf{87.2 \pm 2.4}$	$\mathbf{89.0 \pm 1.9}$	$\mathbf{98.1 \pm 0.7}$
GSA-AbMILP	88.4 ± 1.7	$\mathbf{95.2 \pm 1.7}$	83.7 ± 2.8	$\mathbf{86.9 \pm 2.1}$	$\mathbf{98.5 \pm 0.6}$
IQSA-AbMILP	89.0 ± 1.9	$\mathbf{93.9 \pm 2.1}$	$\mathbf{85.5 \pm 3.0}$	$\mathbf{86.9 \pm 2.5}$	96.6 ± 1.1
LSA-AbMILP	84.8 ± 1.8	$\mathbf{92.7 \pm 2.7}$	71.1 ± 4.6	73.4 ± 4.3	95.5 ± 1.7
MSA-AbMILP	$\mathbf{89.6 \pm 1.6}$	$\mathbf{94.6 \pm 1.5}$	$\mathbf{85.7 \pm 2.7}$	$\mathbf{87.9 \pm 1.8}$	$\mathbf{98.4 \pm 0.5}$

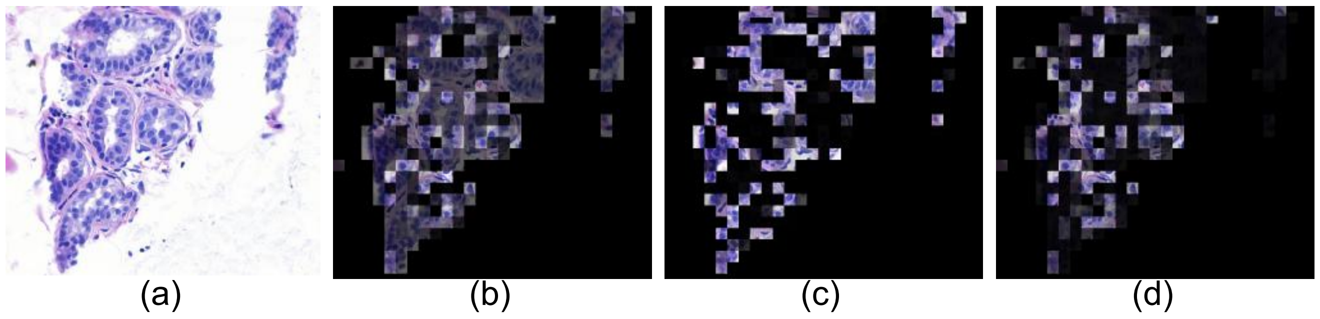


Figure 4: An example image from the breast cancer dataset (a), weights of patches obtained by AbMILP (b) and SA-AbMILP (c), and SA dependencies obtained for the most relevant patch in SA-AbMILP (d). One can observe that SA-AbMILP highlights fewer patches than AbMILP, which simplifies their analysis. Additionally, SA dependencies are justified histologically, as they mostly focus on nuclei located in the neighborhood of crypts.

and batch size 1. We also apply extensive data augmentation, including random rotations, horizontal and vertical flipping, Gaussian noise, and patch normalization. To increase the training set, each iteration randomly selects a different subset of image's patches as an input.

Results. Results for DIFaS database are presented in Table 2. Our method improves almost all of the scores for both feature pooling networks. The exception is the precision for ResNet-18, where our method is on par with maximum and mean instance representation. Moreover, we observe that while non-standard kernels can improve results for representations obtained with ResNet-18, they do not

Table 2: Results for DIFaS dataset (mean and standard error of the mean over 5 repetitions). See Section 3 for description of the shortcuts.

DIFaS (ResNet-18)					
method	accuracy	precision	recall	F-score	AUC
instance+voting	78.3 ± 2.0	78.0 ± 1.4	76.0 ± 1.7	75.8 ± 2.0	N/A
embedding+max	77.1 ± 0.7	$\mathbf{83.1 \pm 0.5}$	77.1 ± 0.7	75.5 ± 0.9	95.3 ± 0.2
embedding+mean	78.1 ± 0.8	$\mathbf{83.3 \pm 0.5}$	78.1 ± 0.8	76.4 ± 1.0	95.2 ± 0.2
AbMILP	77.5 ± 0.6	82.6 ± 0.5	77.5 ± 0.6	75.6 ± 0.8	96.1 ± 0.3
SA-AbMILP	$\mathbf{80.1 \pm 0.6}$	$\mathbf{84.6 \pm 0.6}$	$\mathbf{80.1 \pm 0.6}$	$\mathbf{78.4 \pm 0.8}$	$\mathbf{96.8 \pm 0.3}$
GSA-AbMILP	$\mathbf{79.1 \pm 0.4}$	83.5 ± 0.7	$\mathbf{79.1 \pm 0.4}$	77.2 ± 0.5	$\mathbf{97.0 \pm 0.3}$
IQSA-AbMILP	$\mathbf{79.4 \pm 0.4}$	$\mathbf{83.7 \pm 0.7}$	$\mathbf{79.4 \pm 0.4}$	$\mathbf{77.6 \pm 0.6}$	96.8 ± 0.3
LSA-AbMILP	77.6 ± 0.5	82.5 ± 0.5	77.6 ± 0.5	75.7 ± 0.7	96.2 ± 0.3
MSA-AbMILP	$\mathbf{79.2 \pm 0.4}$	83.5 ± 0.4	$\mathbf{79.2 \pm 0.4}$	$\mathbf{77.5 \pm 0.6}$	$\mathbf{96.9 \pm 0.3}$

DIFaS (AlexNet)					
method	accuracy	precision	recall	F-score	AUC
instance+voting	77.3 ± 1.9	78.4 ± 0.8	76.6 ± 1.2	76.2 ± 1.0	N/A
embedding+max	82.9 ± 1.2	87.1 ± 0.9	82.9 ± 1.2	82.3 ± 1.4	98.4 ± 0.2
embedding+mean	82.3 ± 0.7	87.2 ± 0.4	82.3 ± 0.7	81.5 ± 1.0	98.1 ± 0.3
AbMILP	83.6 ± 1.2	87.8 ± 0.7	83.6 ± 1.2	82.9 ± 1.5	98.6 ± 0.2
SA-AbMILP	$\mathbf{86.0 \pm 1.0}$	$\mathbf{89.6 \pm 0.6}$	$\mathbf{86.0 \pm 1.0}$	$\mathbf{85.7 \pm 1.3}$	$\mathbf{98.9 \pm 0.1}$
GSA-AbMILP	84.6 ± 1.0	89.1 ± 0.6	84.6 ± 1.0	84.2 ± 1.3	98.8 ± 0.2
IQSA-AbMILP	84.1 ± 1.2	88.4 ± 0.6	84.1 ± 1.2	83.4 ± 1.4	$\mathbf{98.9 \pm 0.2}$
LSA-AbMILP	83.9 ± 1.3	88.0 ± 0.7	83.9 ± 1.3	83.2 ± 1.6	98.6 ± 0.2
MSA-AbMILP	83.1 ± 0.8	87.8 ± 0.4	83.1 ± 0.8	82.4 ± 1.0	98.7 ± 0.2

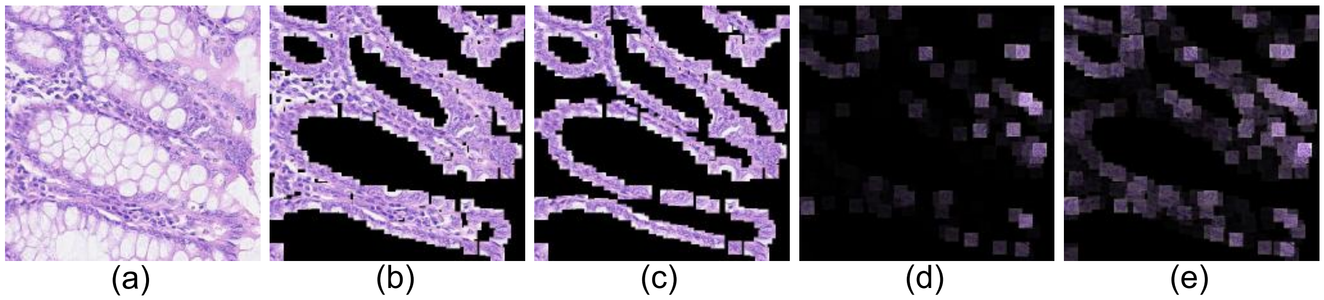


Figure 5: An example image from the colon cancer dataset (a) with annotated nuclei (b) and epithelium nuclei (c), as well as, the weights of patches obtained by AbMILP (d) and SA-AbMILP (e). One can observe that SA-AbMILP strengthens epithelium nuclei and, at the same time, weakens nuclei in the lamina propria.

operate well with those generated with AlexNet. Additionally, to interpret the model outputs, we visualize patches with the lowest and highest weights in the average pooling. As shown in Figure 6, our method properly [38] assigns lower weights to blurred patches with a small number of cells. Also, in contrast to the baseline method, it assigns high weight to clean patches (without red artifacts).

4.4. Retinal image screening dataset.

Experiment details. Dataset "Messidor" from [5] contains 1200 images with 654 positive (diagnosed with diabetes) and 546 negative (healthy) images. The size of each image is 700×700 pixels. Each image is partitioned into patches of 224×224 pixels. Patches containing only background are dropped. We are using ResNet18 [12] pretrained on the ImageNet [6] as an instance feature vectors generator and SA-AbMILP to obtain the final prediction, which is trained in an end to end fashion. The model is trained using Adam [15] optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate $5 * 10^{-6}$, and batch size 1. We also apply data augmentation as in Section 4.3.

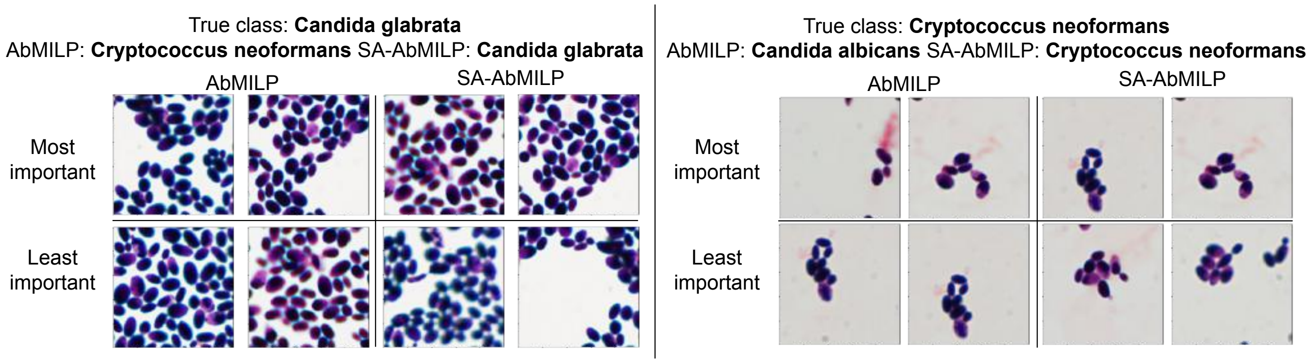

True class: **Candida glabrata**
AbMILP: **Cryptococcus neoformans** SA-AbMILP: **Candida glabrata**
AbMILP
SA-AbMILP
Most important
Least important
True class: **Cryptococcus neoformans**
AbMILP: **Candida albicans** SA-AbMILP: **Cryptococcus neoformans**
AbMILP
SA-AbMILP
Most important
Least important

Figure 6: Example patches from the DIFaS dataset with the lowest and highest weights in the average pooling. One can observe that SA-AbMILP properly [38] assigns lower weights to blurred patches with a small number of cells. Moreover, in contrast to the AbMILP, it assigns high weight to clean patches (without red artifacts).

Table 3: Retinal image screening dataset results. *Results with asterisk are sourced from [30].

Retinal image screening dataset		
method	accuracy	F-score
LSA-AbMILP	**76.3%**	**0.77**
SA-AbMILP	75.2%	0.76
AbMILP	74.5%	0.74
GSA-AbMILP	74.5%	0.75
IQSA-AbMILP	74.5%	0.75
MSA-AbMILP	73.5%	0.74
MIL-GNN-DP*	74.2%	**0.77**
MIL-GNN-Att*	72.9%	0.75
mi-Graph*	72.5%	0.75
MILBoost*	64.1%	0.66
Citation k-NN*	62.8%	0.68
EMDD*	55.1%	0.69
MI-SVM*	54.5%	0.70
mi-SVM*	54.5%	0.71

Results. Results for "Messidor" database are presented in Table 3 alongside results of other approaches which are used for comparison and were taken from [30]. Our method improves accuracy and the highest scores are achieved using Laplace kernel variation, which obtains F1 score on par with the best reference approach. This is the only database for which Laplace kernel obtains the best results, which only confirms that the kernel should be optimized together with the other hyperparameters when applied to new problems.

5. Conclusions and discussion

In this paper, we apply Self-Attention into Attention-based MIL Pooling (SA-AbMILP), which combines the multi-level dependencies across image regions with the trainable operator of weighted average pooling. In contrast to Attention-based MIL Pooling (AbMILP), it covers not only the standard but also the presence-based and threshold-based assumptions of MIL. Self-Attention is detecting the relationship between instances, so it can embed into the instance feature vectors the information about the presence of similar instances or find that a combination of specific instances defines the bag as a positive. The experiments on five datasets (MNIST, two histological datasets of breast and colon cancer, microbiological dataset DIFaS, and retinal image screening) confirm that our method is on par or outperforms current state-of-the-art methodology based on the Wilcox pair test. We demonstrate that in the case of bigger datasets, it is advisable to use various kernels of the self-attention instead of the commonly used dot product. We also provide qualitative results to illustrate the reason for the improvements achieved by our method.

The experiments show that methods covering a wider range of MIL assumptions fit better for real-world problems. Therefore, in future work, we plan to introduce methods for more challenging MIL assumptions, e.g. collective assumption, and apply them to more complicated tasks, like digestive track assessment using the Nancy Histological Index. Moreover, we plan to introduce better interpretability, using Prototype Networks.

6. Acknowledgments

The POIR.04.04.00-00-14DE/18-00 project is carried out within the Team-Net programme of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund.

References

[1] Gaston Baudat and Fatiha Anouar. Kernel-based methods and function approximation. In *IJCNN'01. International Joint Conference on Neural Networks.*

Proceedings (Cat. No. 01CH37222), volume 2, pages 1244–1249. IEEE, 2001.

[2] Gabriele Campanella, Matthew G Hanna, Luke Geneslaw, Allen Miraflor, Vitor Werneck Krauss Silva, Klaus J Busam, Edi Brogi, Victor E Reuter, David S Klimstra, and Thomas J Fuchs. Clinical-grade computational pathology using weakly supervised deep learning on whole slide images. *Nature medicine*, 25(8):1301–1309, 2019.

[3] Gabriele Campanella, Vitor Werneck Krauss Silva, and Thomas J Fuchs. Terabyte-scale deep multiple instance learning for classification and localization in pathology. *arXiv preprint arXiv:1805.06983*, 2018.

[4] 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.

[5] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, et al. Feedback on a publicly distributed image database: the messidor database. *Image Analysis & Stereology*, 33(3):231–234, 2014.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] 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.

[8] Ji Feng and Zhi-Hua Zhou. Deep miml network. In *AAAI*, pages 1884–1890, 2017.

[9] James Foulds and Eibe Frank. A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(1):1–25, 2010.

[10] Elisa Drelie Gelasca, Jiyun Byun, Boguslaw Obara, and BS Manjunath. Evaluation and benchmark for biological image segmentation. In *2008 15th IEEE International Conference on Image Processing*, pages 1816–1819. IEEE, 2008.

[11] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Attention-based deep multiple instance learning. *arXiv preprint arXiv:1802.04712*, 2018.

[14] Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Jiayun Li, Wenyuan Li, Arkadiusz Gertych, Beatrice S Knudsen, William Speier, and Corey W Arnold. An attention-based multi-resolution model for prostate whole slide imageclassification and localization. *arXiv preprint arXiv:1905.13208*, 2019.

[19] Katherine Li, Richard Strauss, Colleen Marano, Linda E Greenbaum, Joshua R Friedman, Laurent Peyrin-Biroulet, Carrie Brodmerkel, and Gert De Hertogh. A simplified definition of histologic improvement in ulcerative colitis and its association with disease outcomes up to 30 weeks from initiation of therapy: Post hoc analysis of three clinical trials. *Journal of Crohn's and Colitis*, 13(8):1025–1035, 2019.

[20] Ming Y Lu, Richard J Chen, Jingwen Wang, Debora Dillon, and Faisal Mahmood. Semi-supervised histology classification using deep multiple instance learning and contrastive predictive coding. *arXiv preprint arXiv:1910.10825*, 2019.

[21] Gwenolé Quellec, Guy Cazuguel, Béatrice Cochener, and Mathieu Lamard. Multiple-instance learning for medical image and video analysis. *IEEE reviews in biomedical engineering*, 10:213–234, 2017.

[22] Gwénolé Quellec, Mathieu Lamard, Michael D Abràmoff, Etienne Decencière, Bruno Lay, Ali Erginay, Béatrice Cochener, and Guy Cazuguel. A multiple-instance learning framework for diabetic retinopathy screening. *Medical image analysis*, 16(6):1228–1240, 2012.

[23] Priya Rani, Rajkumar Elagiri Ramalingam, Kumar T Rajamani, Melih Kandemir, and Digvijay Singh. Multiple instance learning: Robust validation on retinopathy of prematurity. *Int J Ctrl Theory Appl*, 9:451–459, 2016.

[24] Lucia Ricci-Vitiani, Dario G Lombardi, Emanuela Pilozzi, Mauro Biffoni, Matilde Todaro, Cesare Peschle, and Ruggero De Maria. Identification and expansion of human colon-cancer-initiating cells. *Nature*, 445(7123):111–115, 2007.

[25] 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.

[26] Christoph Straehle, Melih Kandemir, Ullrich Koethe, and Fred A Hamprecht. Multiple instance learning with response-optimized random forests. In *2014 22nd International Conference on Pattern Recognition*, pages 3768–3773. IEEE, 2014.

[27] Jakub M Tomczak, Maximilian Ilse, Max Welling, Marnix Jansen, Helen G Coleman, Marit Lucas, Kikki de Laat, Martijn de Bruin, Henk Marquering, Myrtle J van der Wel, et al. Histopathological classification of precursor lesions of esophageal adenocarcinoma: A deep multiple instance learning approach. 2018.

[28] Tong Tong, Robin Wolz, Qinquan Gao, Ricardo Guerrero, Joseph V Hajnal, Daniel Rueckert, Alzheimer's Disease Neuroimaging Initiative, et al. Multiple instance learning for classification of dementia in brain mri. *Medical image analysis*, 18(5):808–818, 2014.

[29] Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

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

[31] Chen Wang, Jianfei Yang, Lihua Xie, and Junsong Yuan. Kervolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 31–40, 2019.

[32] Shujun Wang, Yaxi Zhu, Lequan Yu, Hao Chen, Huangjing Lin, Xiangbo Wan, Xinjuan Fan, and Pheng-Ann Heng. Rmdl: Recalibrated multi-instance deep learning for whole slide gastric image classification. *Medical image analysis*, 58:101549, 2019.

[33] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European conference on computer vision (ECCV)*, pages 399–417, 2018.

[34] Hiroshi Yoshida, Taichi Shimazu, Tomoharu Kiyuna, Atsushi Marugame, Yoshiko Yamashita, Eric Cosatto, Hirokazu Taniguchi, Shigeki Sekine, and Atsushi Ochiai. Automated histological classification of whole-slide images of gastric biopsy specimens. *Gastric Cancer*, 21(2):249–257, 2018.

[35] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

[36] Zhendong Zhao, Gang Fu, Sheng Liu, Khaled M Elokely, Robert J Doerksen, Yixin Chen, and Dawn E Wilkins. Drug activity prediction using multiple-instance learning via joint instance and feature selection. In *BMC bioinformatics*, volume 14, page S16. Springer, 2013.

[37] 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*, pages 1249–1256, 2009.

[38] Bartosz Zieliński, Agnieszka Sroka-Oleksiak, Dawid Rymarczyk, Adam Piekarczyk, and Monika Brzychczy-Włoch. Deep learning approach to description and classification of fungi microscopic images. *arXiv preprint arXiv:1906.09449*, 2019.

ProtoMIL: Multiple Instance Learning with Prototypical Parts for Whole-Slide Image Classification

Dawid Rymarczyk[1,2](✉), Adam Pardyl[1], Jarosław Kraus[1], Aneta Kaczyńska[1], Marek Skomorowski[1], and Bartosz Zieliński[1,2]

[1] Faculty of Mathematics and Computer Science, Jagiellonian University, 6 Łojasiewicza Street, 30-348 Kraków, Poland
{dawid.rymarczyk,adam.pardyl,jarek.kraus, aneta.kaczynska}@student.uj.edu.pl, {marek.skomorowski,bartosz.zielinski}@uj.edu.pl
[2] Ardigen SA, 76 Podole Street, 30-394 Kraków, Poland

Abstract. The rapid development of histopathology scanners allowed the digital transformation of pathology. Current devices fastly and accurately digitize histology slides on many magnifications, resulting in whole slide images (WSI). However, direct application of supervised deep learning methods to WSI highest magnification is impossible due to hardware limitations. That is why WSI classification is usually analyzed using standard Multiple Instance Learning (MIL) approaches, that do not explain their predictions, which is crucial for medical applications. In this work, we fill this gap by introducing ProtoMIL, a novel self-explainable MIL method inspired by the case-based reasoning process that operates on visual prototypes. Thanks to incorporating prototypical features into objects description, ProtoMIL unprecedentedly joins the model accuracy and fine-grained interpretability, as confirmed by the experiments conducted on five recognized whole-slide image datasets.

Keywords: Multiple instance learning · Digital pathology · Interpretable deep learning

1 Introduction

A typical supervised learning scenario assumes that each data point has a separate label. However, in Whole Slide Image (WSI) classification, only one label is usually assigned to a gigapixel image due to the laborious and expensive labeling. Because of the hardware limitations, the direct application of supervised

A. Pardyl, J. Kraus and A. Kaczyńska–Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-26387-3_26.

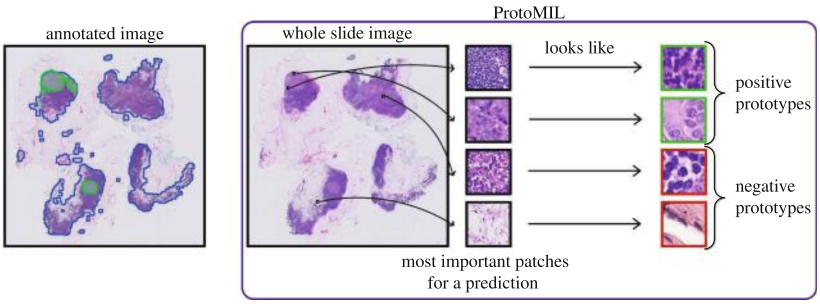


Fig. 1. ProtoMIL divides the whole slide image into patches and analyzes their similarity to the reference prototypical parts that describe the given data class. As a result, it can provide a visual explanation of its prediction. One can observe that ProtoMIL identifies the most important patches with attention weights, that can appear both inside and outside a cancer region (marked as green and blue areas, respectively). Moreover, these patches are described by cancer or healthy tissue prototypes (corresponding to patches in green and red frames, respectively), showing their resemblance to the training examples. (Color figure online)

deep learning methods to WSI two highest magnification is impossible. That is why recent approaches [24] divide the WSI into smaller patches (instances) and process them separately to obtain their representations. Such representations form a bag of instances associated with only one label, and it is unspecified which instances are responsible for this label [15]. This kind of problem, called Multiple Instance Learning (MIL) [12], appears in many medical problems, such as the diabetic retinopathy screening [30,31], bacteria clones identification using microscopy images [7], or identifying conformers responsible for molecule activity in drug design [42,47].

In recent years, with the rapid development of deep learning, MIL is combined with many neural network-based models [14,20,24,27,34,38,39,43–45]. Many of them embed all instances of the bag using a convolutional block of a deep network and then aggregate those embeddings. Moreover, some aggregation methods specify the most important instances that are presented to the user as prediction interpretation [20,24,27,34,39]. However, those methods usually only exhibit instances crucial for the prediction and do not indicate the cause of their importance. Naturally, there were attempts to further explain the MIL models [6,7,25], but overall, they usually introduce additional bias into the explanation [33] or require additional input [25].

To address the above shortcomings of MIL models, we introduce *Prototypical Multiple Instance Learning* (ProtoMIL). It builds on case-based reasoning, a type of explanation naturally used by humans to describe their thinking process [23]. More precisely, we divide each WSI into patches and analyze how similar they are to a trainable set prototypical parts of positive and negative data classes, as defined in [8]. Since, the prototypes are trainable, they are automatically derived by ProtoMIL. Then, we apply an attention pooling operator to accumulate those similarities over instances. As a result, we obtain bag-level representation classified with an additional neural layer. This approach significantly differs from non-MIL approaches because it applies an aggregation layer and introduces a novel regularization technique that encourages the model to derive prototypes from the instances responsible for the positive label of a bag. The latter is a challenging problem because those instances are concealed and underrepresented. Lastly, the prototypical parts are pruned to characterize the data classes compactly. This results in detailed interpretation, where the most important patches according to attention weights are described using prototypes, as shown in Fig. 1.

To show the effectiveness of our model, we conduct experiments on five WSI datasets: Bisque Breast Cancer [16], Colon Cancer [41], Camelyon16 Breast Cancer [13], Lung cancer subtype identification TCGA-NSCLC [5] and Kidney cancer subtype classification [2]. Additionally, in the Supplementary Materials, we show the universal character of our model in different scenarios such as MNIST Bags [20] and Retinopathy Screening (Messidor dataset) [11]. The results we obtain are usually on par with the current state-of-the-art models. However, at the same time, we strongly enhance interpretation capabilities with prototypical parts obtained from the training set. We made our code publicly available at https://github.com/apardyl/ProtoMIL.

The main contributions of this work are as follows:

– Introducing the ProtoMIL method, which substantially improves the interpretability of existing MIL models by introducing case-based reasoning.
– Developing a training paradigm that encourages generating prototypical parts from the underrepresented instances responsible for the positive label of a bag.

The paper is organized as follows. In Sect. 2, we present recent advancements in Multiple Instance Learning and deep interpretable models. In Sect. 3, we define the MIL paradigms and introduce ProtoMIL. Finally, in Sect. 4, we present the results of conducted experiments, and Sect. 5 summarizes the work.

2 Related Works

Our work focuses on classification of whole slide images which is described using Multiple Instance Learning (MIL) framework. Additionally, we develop an interpretable method which relates to eXplainable Artificial Intelligence (XAI). We briefly describe both fields in the following subsections.

2.1 Multiple Instance Learning

Before the deep learning era, models based on SVM, such as MI-SVM [3], were used for MIL problems. However, currently, MIL is addressed with numerous deep models. One of them, Deep MIML [14], introduces a sub-concept layer that is learned and then pooled to obtain a bag representation. Another example is mi-Net [44], which pools predictions from single instances to derive a bag-level prediction. Other architectures adapted to MIL scenarios includes capsule networks [45], transformers [38] and graph neural networks [43]. Moreover, many works focus on the attention-based pooling operators, like AbMILP introduced in [20] that weights the instances embeddings to obtain a bag embedding. This idea was also extended by combining it with mi-Net [24], clustering similar instances [27], self-attention mechanism [34], and sharing classifier weights with pooling operator [39]. However, the above methods either do not contain an XAI component or only present the importance of the instances. Hence, our ProtoMIL is a step towards the explainability of the MIL methods.

2.2 Explainable Artificial Intelligence

There are two types of eXplainable Artificial Intelligence (XAI) approaches, post hoc and self-explaining methods [4]. Among many *post hoc* techniques, one can distinguish saliency maps showing pixel importance [32,36,37,40] or concept activation vectors representing internal network state with human-friendly concepts [9,17,21,46]. They are easy to use since they do not require any changes in the model architecture. However, their explanations may be unfaithful and fragile [1]. Therefore *self-explainable* models were introduced like Prototypical Part Network [8] with a layer of prototypes representing the activation patterns. A similar approach for hierarchically organized prototypes is presented in [18] to classify objects at every level of a predefined taxonomy. Moreover, some works concentrate on transforming prototypes from the latent space to data space [26] or focus on sharing prototypical parts between classes and finding semantic similarities [35]. Other works [28] build a decision tree with prototypical parts in the nodes or learn disease representative features within a dynamic area [22]. Nonetheless, to our best knowledge, no fine-grained self-explainable method, like ProtoMIL, exists for MIL problems.

3 ProtoMIL

Due to the large resolution of whole slide images, which should not be scaled down due to loss of information, we first divide an image into patches. However, we do not know which patches correspond to the given disease state. Therefore, this problem boils down to Multiple Instance Learning (MIL), where there is a bag of instances (in our case patches) and only one label for the whole bag. This bag is passed trough the four modules of ProtoMIL (see Fig. 2): convolutional network f_{conv}, prototype layer f_{proto}, attention pooling a, and fully connected last layer g. Convolutional and prototype layers process single instances, whereas attention pooling and the last layer work on a bag level. More precisely, given a bag of patches $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$, each $\mathbf{x} \in X$ is forwarded through convolutional layers to obtain low-dimensional embeddings $F = \{f_{conv}(\mathbf{x}_1), \ldots, f_{conv}(\mathbf{x}_k)\}$. As $f_{conv}(\mathbf{x}) \in H \times W \times D$, for the clarity of description, let $Z_{\mathbf{x}} = \{\mathbf{z}_j \in f_{conv}(\mathbf{x}) : \mathbf{z}_j \in \mathbb{R}^D, j = 1..HW\}$. Then, the prototype layer computes vector $\mathbf{h}$ of similarity scores [8] between each embedding $f_{conv}(\mathbf{x})$ and all prototypes $\mathbf{p} \in P$ as

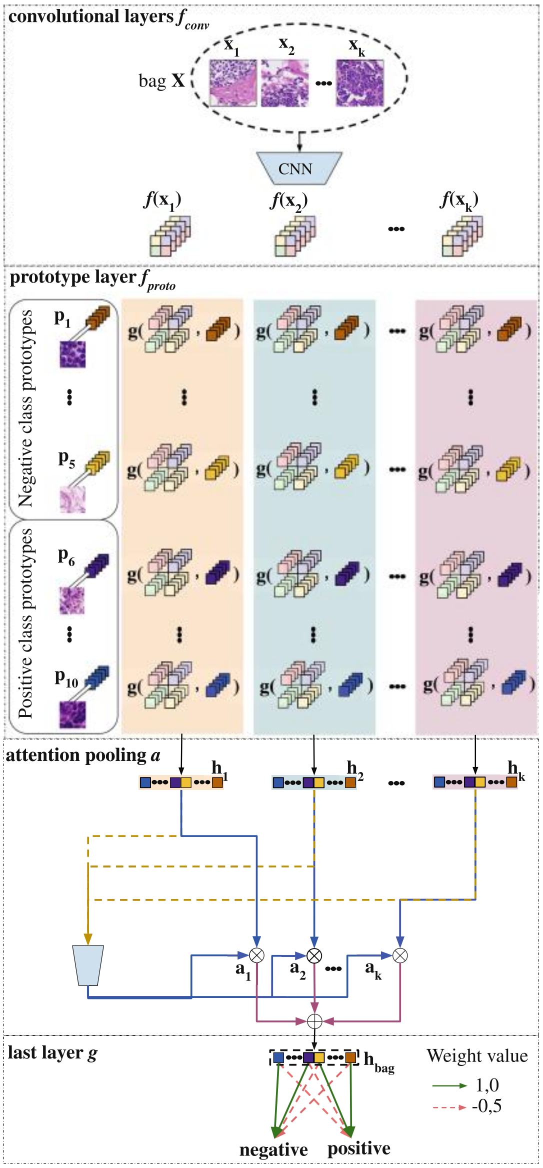


Fig. 2. ProtoMIL passes a bag of patches through four modules. First, convolutional layer f_{conv} generates embeddings for each patch. Then, the prototype layer f_{proto} calculates similarities between patches representations and its prototypes. The similarities are aggregated using the attention pooling a to obtain the bag similarity scores classified using the last layer g. Notice that particular colors in vectors $\mathbf{h_i}$ and $\mathbf{h_{bag}}$ correspond to prototypes similarities.

$$\mathbf{h} = \left(g(Z_{\mathbf{x}}, \mathbf{p}) = \max_{\mathbf{z} \in Z_{\mathbf{x}}} \log \left(\frac{\| \mathbf{z} - \mathbf{p} \|^2 + 1}{\| \mathbf{z} - \mathbf{p} \|^2 + \varepsilon} \right) \right)_{\mathbf{p} \in P} \quad \text{for} \ \ \varepsilon > 0.$$

This results in a bag of similarity scores $H = \{\mathbf{h}_1, \ldots, \mathbf{h}_k\}$, which we pass to the attention pooling [20] to obtain a single similarity scores for the entire bag

$$\mathbf{h}_{bag} = \sum_{i=1}^{k} a_i \, \mathbf{h}_i, \quad \text{where} \ \ a_i = \frac{\exp\{\mathbf{w}^T (\tanh(\mathbf{V}\, \mathbf{h}_i^T) \odot \text{sigm}(\mathbf{U}\, \mathbf{h}_i^T)\}}{\sum_{j=1}^{k} \exp\{\mathbf{w}^T (\tanh(\mathbf{V}\, \mathbf{h}_j^T) \odot \text{sigm}(\mathbf{U}\, \mathbf{h}_j^T)\}}, \tag{1}$$

$\mathbf{w} \in \mathbb{R}^{L \times 1}$, $\mathbf{V} \in \mathbb{R}^{L \times M}$, and $\mathbf{U} \in \mathbb{R}^{L \times M}$ are parameters, tanh is the hyperbolic tangent, sigm is the sigmoid non-linearity and $\odot$ is an element-wise multiplication. Note that weights a_i sum up to 1, and thus the formula is invariant to the size of the bag. Such representation is then sent to the last layer to obtain the predicted label $\check{y} = g(h_{bag})$ as in [8].

Regularization. In MIL, the instances responsible for the positive label of a bag are underrepresented. Hence, training ProtoMIL without additional regularizations can result in a prototype layer with only prototypes of a negative class. That is why we introduce a novel regularization technique that encourages the model to derive positive prototypes. For this purpose, we introduce the loss function composed of three components

$$\mathcal{L}_{\text{CE}}(\check{y}, y) + \lambda_1 \, \mathcal{L}_{\text{Clst}} + \lambda_2 \, \mathcal{L}_{\text{Sep}},$$

where $\check{y}$ and y denotes respectively the predicted and ground truth label of bag X, $\mathcal{L}_{\text{CE}}$ corresponds to cross-entropy loss, while

$$\mathcal{L}_{\text{Clst}} = \frac{1}{|X|} \sum_{\mathbf{x}_i \in X} a_i \min_{\mathbf{p} \in P^y} \min_{\mathbf{z} \in Z_{\mathbf{x}_i}} \|\mathbf{z} - \mathbf{p}\|_2^2,$$

$$\mathcal{L}_{\text{Sep}} = -\frac{1}{|X|} \sum_{\mathbf{x}_i \in X} a_i \min_{\mathbf{p} \notin P^y} \min_{\mathbf{z} \in Z_{\mathbf{x}_i}} \|\mathbf{z} - \mathbf{p}\|_2^2,$$

where P^y is a set of prototypes assigned to class y. Comparing to [8], components $\mathcal{L}_{\text{Clst}}$ and $\mathcal{L}_{\text{Sep}}$ additionally use a_i from Eq. 1. As a result, we encourage the model to create more prototypes corresponding to positive instances, which usually have higher a_i values.

4 Experiments

We test our ProtoMIL approach on five datasets, for which we train the model from scratch in three steps: (i) *warmup* phase with training all layers except the last one, (ii) prototype projection, (iii) and fine-tuning with fixed f_{conv} and f_{proto}. Phases (ii) and (iii) are repeated several times to find the most optimal set of prototypes. All trainings use Adam optimizer for all layers with $\beta_1 = 0.99$,

$\beta_2 = 0.999$, weight decay 0.001, and batch size 1. Additionally, we use an exponential learning rate scheduler for the *warmup* phase and a step scheduler for prototype training. All results are reported as an average of all runs with a standard error of the mean. In the subsequent subsections, we describe experiment details and results for each dataset.

Across all datasets we use convolutional block from ResNet-18 followed by two additional 1×1 convolutions as the convolutional layer f_{conv}. We use ReLU as the activation function for all convolutional layers except the last layer, for which we use the sigmoid activation function. The prototype layer stores prototypes shared across all bags, while the attention layer implements AbMILP. The last layer is used to classify the entire bag. Weights between similarity scores of prototypes corresponding class logit are initialized with 1, while other connections are set to -0.5 as in [8]. Together with the specific training procedure, such initialization results in a positive reasoning process (we rather say "this looks like that" instead of saying "this does not look like that").

4.1 Bisque Breast Cancer and Colon Cancer Datasets

Experiment Details. We experiment on two histological datasets: Colon Cancer and Bisque Breast Cancer. The former contains 100 H&E images with $22,444$ manually annotated nuclei of four different types: epithelial, inflammatory, fibroblast, and miscellaneous. To create bags of instances, we extract 27×27 nucleus-centered patches from each image, and the goal is to detect if the bag contains one or more epithelial cells, as colon cancer originates from them. On the other hand, the Bisque dataset consists of 58 H&E breast histology images of size 896×768, out of which 32 are benign, and 26 are malignant (contain at least one cancer cell). Each image is divided into 32×32 patches, resulting in 672 patches per image. Patches with at least 75% of the white pixels are discarded, resulting in 58 bags of various sizes.

We apply extensive data augmentation for both datasets, including random rotations, horizontal and vertical flipping, random staining augmentation, staining normalization, and instance normalization. We use ResNet-18 convolutional parts with the first layer modified to 3×3 convolution with stride 1 to match the size of smaller instances. We set the number of prototypes per class to 10 with a size of $128 \times 2 \times 2$. Warmup, fine-tuning, and end-to-end training take 60, 20, and 20 epochs, respectively. 10-fold cross-validation with 1 validation fold and 1 test fold is repeated 5 times.

Results. Table 1 presents our results compared to both traditional and attention-based MIL models. On the Bisque dataset, our model significantly outperforms all baseline models. However, due to the small size of the Colon Cancer dataset, ProtoMIL overfits, resulting in poorer AUC than attention-based models. Nevertheless, in both cases, ProtoMIL provides finer explanations than all baseline models (see Fig. 3 and Supplementary Materials).

Table 1. Results for small histological datasets, where ProtoMIL significantly outperforms baseline methods on the Bisque dataset. However, it achieves worse results for the Colon Cancer dataset, probably due to its small size. Additionally, interpretability of the methods is noted and further discussed in Sect. 4.6. Notice that values for comparison indicated with "*" and "**" comes from [20] and [34], respectively.

Method	Bisque		Colon Cancer		
	Accuracy	AUC	Accuracy	AUC	Inter.
instance+max*	61.4% ± 2.0%	0.612 ± 0.026	84.2% ± 2.1%	0.914 ± 0.010	+
instance+mean*	67.2% ± 2.6%	0.719 ± 0.019	77.2% ± 1.2%	0.866 ± 0.008	–
embedding+max*	60.7% ± 1.5%	0.650 ± 0.013	82.4% ± 1.5%	0.918 ± 0.010	–
embedding+mean*	74.1% ± 2.3%	0.796 ± 0.012	86.0% ± 1.4%	0.940 ± 0.010	–
AbMILP*	71.7% ± 2.7%	0.856 ± 0.022	88.4% ± 1.4%	0.973 ± 0.007	++
SA-AbMILP**	75.1% ± 2.4%	0.862 ± 0.022	**90.8% ± 1.3%**	**0.981 ± 0.007**	+
ProtoMIL (our)	**76.7% ± 2.2%**	**0.886 ± 0.033**	81.3% ± 1.9%	0.932 ± 0.014	+++

4.2 Camelyon16 Dataset

Experiment Details. The Camelyon16 dataset [13] consists of 399 whole-slide images of breast cancer samples, each labeled as *normal* or *tumor*. We create MIL bags by dividing each slide $20x$ resolution image into 224×224 patches, rejecting patches that contain more than 70% of background. This results in 399 bags with a mean of 8,871 patches and a standard deviation of 6,175. Moreover, 20 largest bags are truncated to 20,000 random patches to fit into the memory of a GPU. The positive patches are again highly imbalanced, as only less than 10% of patches contain tumor tissue.

Due to the size of the dataset, we preprocess all samples using a ResNet-18 without two last layers, pre-trained on various histopathological images using self-supervised learning from [10]. The resulting embeddings are fed into our model to replace the feature backbone net. ProtoMIL is trained for 50, 40, and 10 epochs in warmup, fine-tuning, and end-to-end training, respectively. The number of prototypes per class is limited to 5 with no data augmentation. The experiments are repeated 5 times with the original train-test split.

Results. We compare ProtoMIL to other state-of-the-art MIL techniques, including both traditional mean and max MIL pooling, RNN, attention-based MIL pooling, and transformer-based MIL pooling [38]. ProtoMIL performs on par in terms of accuracy and slightly outperforms other models on AUC metric (Table 2) while providing a better understanding of its decision process, as presented in Fig. 4 and Supplementary Materials.

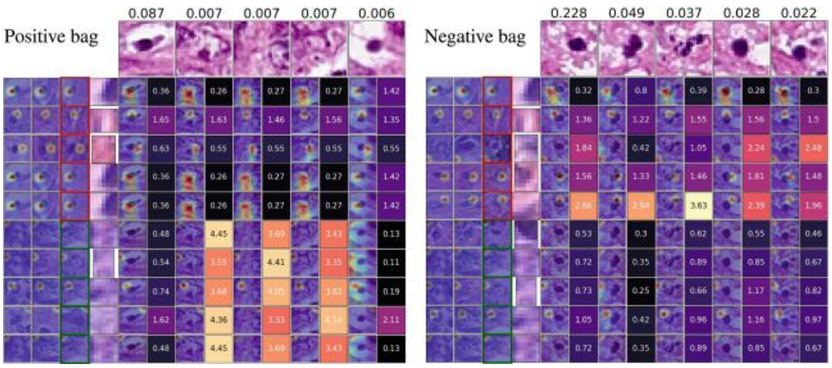


Fig. 3. Similarity scores between five crucial instances of a bag (columns) and ten prototypical parts (rows) for a positive and negative bag (left and right side, respectively) from the Colon Cancer bags. Each prototypical part is represented by a part of the training image and three nearest training patches, and each instance is represented by the patch and the value of its attention weight a_i. Moreover, each cell contains a similarity score and a heatmap corresponding to prototype activation. One can observe that instances of a negative bag usually activate prototypes of a negative class (four upper prototypes in red brackets), while the instances of positive bags mostly activate positive prototypes (four bottom prototypes in green brackets). (Color figure online)

4.3 TCGA-NSCLC Dataset

Experiment details. TCGA-NSCLC includes two subtype projects, i.e., Lung Squamous Cell Carcinoma (TGCA-LUSC) and Lung Adenocarcinoma (TCGA-LUAD), for a total of 956 diagnostic WSIs, including 504 LUAD slides from 478 cases and 512 LUSC slides from 478 cases. We create MIL bags using WSI Segmentation and Patching from [27] with default parameters, except patch-level parameter set to 1. Each slide image is cropped into a series of 224×224 patches. This results in $1,016$ bags with a mean of $3,961$ patches. We randomly split the data in the ratio of train:valid:test equal 60:15:25 and assure that there is no case overlap between the sets, and use the same ProtoMIL settings as in the Camelyon16 dataset are used. The results are reported for 4-fold cross-validation.

Results. Results for the TCGA-NSCLC dataset are presented in Table 2 alongside results of other state-of-the-art approaches from [38]. ProtoMIL performs slightly lower on the Area Under the ROC Curve (AUC) and accuracy metrics than the powerful transformer-based model TransMIL but still is competitive to other CNN-based approaches. However, the advantage of ProtoMIL is its capability to provide a detailed explanation of predictions as presented in Fig. 5 and Supplementary Materials.

Table 2. Our ProtoMIL achieves state-of-the-art results on the Camelyon16 dataset in terms of AUC metric, surpassing even the transformer-based architecture. Moreover, it is competitive on TCGA-NSCLC and slightly worse on TCGA-RCC, with a small drop of accuracy and AUC compared to TransMIL. Additionally, interpretability of the methods is noted and further discussed in Sect. 4.6. Notice that values for comparison marked with "*" and "**" are taken from [24] and [38], respectively.

Method	Camelyon16		TCGA-NSCLC		TCGA-RCC		
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Inter.
instance+mean*	79.84%	0.762	72.82%	0.840	90.54%	0.978	–
instance+max*	82.95%	0.864	85.93%	0.946	93.78%	0.988	+
MILRNN*	80.62%	0.807	86.19%	0.910	–	–	–
ABMILP*	84.50%	0.865	77.19%	0.865	89.34%	0.970	++
DSMIL*	86.82%	0.894	80.58%	0.892	92.94%	0.984	++
CLAM-SB**	87.60%	0.881	81.80%	0.881	88.16%	0.972	+
CLAM-MB**	83.72%	0.868	84.22%	0.937	89.66%	0.980	+
TransMIL**	**88.37%**	0.931	**88.35%**	**0.960**	**94.66%**	**0.988**	+
ProtoMIL (our)	87.29%	**0.935**	83.66%	0.918	92.79%	0.961	+++

4.4 TCGA-RCC Dataset

Experiment details. TCGA-RCC consists of three unbalanced classes: Kidney Chromophobe Renal Cell Carcinoma (TGCA-KICH, 111 slides from 99 cases), Kidney Renal Clear Cell Carcinoma (TCGA-KIRC, 489 slides from 483 cases), and Kidney Renal Papillary Cell Carcinoma (TCGA-KIRP, 284 slides from 264 cases) for a total of 884 WSIs. We create MIL bags using WSI Segmentation and Paching from [27] with default parameters and a patch-level parameter set to 1. Each slide image is cropped into a series of 224×224 patches. This results in 884 bags with a mean of $4,309$ patches. A separate model is trained for each class, and scores are averaged for all classes. Other experiment settings are identical as for TCGA-NSCLC described above.

Results. We compare ProtoMIL to other state-of-the-art MIL techniques, including both traditional mean and max MIL pooling, attention-based MIL pooling, and transformer-based MIL pooling [38]. ProtoMIL performs on par in terms of accuracy and AUC metric (Table 2) while providing a better understanding of its decision process, as presented in Supplementary Materials.

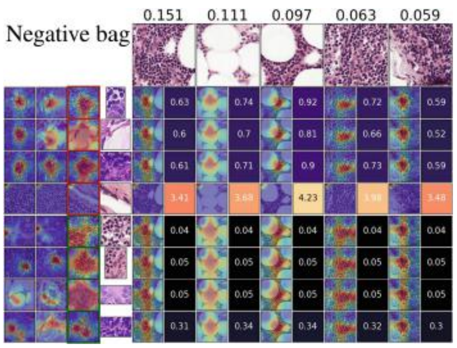


Fig. 4. Similarity scores between five crucial instances of a bag (columns) and eight prototypical parts (rows) for a negative bag from the Camelyon16 dataset. One can observe that ProtoMIL strongly activates only one prototype and focuses mainly on nuclei when analyzing the healthy parts of the tissue. Please refer to Fig. 3 for a detailed description of the visualization.

Table 3. The influence of ProtoMIL pruning on the accuracy and AUC score. One can notice that even though the pruning removes around 30% of the prototypes, it usually does not noticeably decrease the AUC and accuracy of the model.

Dataset	Before pruning			After pruning		
	Proto. #	Accuracy	AUC	Proto. #	Accuracy	AUC
Bisque	20 ± 0	76.7% ± 2.2%	0.886 ± 0.033	13.6 ± 0.25	73.0% ± 2.4%	0.867 ± 0.022
Colon Cancer	20 ± 0	81.3% ± 1.9%	0.932 ± 0.014	15.69 ± 0.34	81.8% ± 2.4%	0.880 ± 0.022
Camelyon16	10 ± 0	87.3% ± 1.2 %	0.935 ± 0.007	6.4 ± 0.24	85.9% ± 1.5%	0.937 ± 0.007
TCGA-NSCLC	10 ± 0	83.66% ± 1.6%	0.918 ± 0.003	7.6 ± 1.2	81.1% ± 1.4%	0.880 ± 0.003
TCGA-RCC	10 ± 0	94.66% ± 1.0%	0.988 ± 0.009	6.2 ± 1.2	91.5% ± 1.2%	0.955 ± 0.006

4.5 Pruning

Experiment Details. We run prototype pruning experiments on all the datasets to remove not class-specific prototypical parts and check their influence on the model performance. For each of them, we use the model trained in the previously described experiments. As pruning parameters, we use $k = 6$ and $l = 40\%$ and fine-tuned for 20 epochs. Details about pruning operation are described in the Supplementary Materials.

Results. The accuracy and AUC in respect to the number of prototypes before and after pruning are presented in Table 3. For all datasets, the number of prototypes after pruning has decreased around 30% on average. However, it does not result in a noticeable decrease in accuracy or AUC, except for Colon Cancer, where we observe a significant drop in AUC. Most probably, it is caused by the high visual resemblance of nuclei patches (especially between *epithelial* and *miscellaneous*) that after prototype projection may be very close to each other in the latent space.

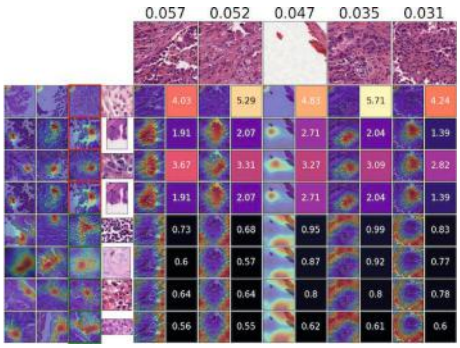


Fig. 5. Similarity scores between five crucial instances of a bag (columns) and eight prototypical parts (rows) for a LUAD type bag from the TCGA-NSCLC dataset.

4.6 Interpretability of MIL Methods

Column *Inter.* in Tables 1, and 2 indicates how interpretable are the considered models. Instances and embeddings-based methods, except instance-max, are not interpretable, similarly to MILRNN, since they lose information about instances crucial for the prediction. On the other hand, the AbMILP [20] identifies crucial instances within a bag and can present the local explanation to the users. However, other attention-based methods, such as SA-AbMILP [34], TransMIL [38] and CLAMs [27] perform additional operations, like self-attention, requiring more effort from the user to analyze the explanation. That is why those methods have been assigned with lower interpretability. Moreover, DS-MIL [24] finds a decision boundary on the bag level and can produce a more detailed explanation than AbMILP, but only for a single prediction (local explanations). In contrast, the ProtoMIL can produce both local (see Fig. 3) and global explanations (see Supplementary Materials).

5 Discussion and Conclusions

In this work, we introduce Prototypical Multiple Instance Learning (ProtoMIL), a method for Whole Slide Image classification that incorporates a case-based reasoning process into the attention-based MIL setup. In contrast to existing MIL methods, ProtoMIL provides a fine-grained interpretation of its predictions. For this purpose, it uses a trainable set of prototypical parts correlated with data classes. The experiments on five datasets confirm that introducing fine-grained interpretability does not reduce the model's effectiveness, which is still on par with the current state-of-the-art methodology. Moreover, the results can be presented to the user with a novel visualization technique.

The experiments show that ProtoMIL can be applied to a challenging problem like Whole-Slide Image classification. Therefore, in future works, we plan to generalize our method to multi-label scenarios and multimodal classification problems since WSI often comes with other medical data like CT and MRI.

5.1 Limitations

ProtoMIL limitations are inherited from the other prototype-based models, such as non-obvious prototype meaning. Ergo, prototype projection might still result in uncertainty on which attributes it represents. However, there are methods mitigating these, e.g. explainer defined in [29].

5.2 Negative Impact

Our solution is based on prototypical parts that are susceptible to different types of adversarial attacks such as [19]. That is why practitioners shall address this risk in a deployed system with ProtoMIL. What is more, it may be used in information war to disinform societies when prototypes are obtained with spoiled data or are shown without appropriate comment, especially in fields like medicine.

Acknowledgments. This work was founded by the POIR.04.04.00-00-14DE/18-00 project carried out within the Team-Net programme of the Foundation for Polish Science cofinanced by the European Union under the European Regional Development Fund. This research was funded by the National Science Centre, Poland (research grant no. 2021/41/B/ST6/01370). For the purpose of Open Access, the authors have applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

References

1. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. In: Advances in Neural Information Processing Systems, pp. 9505–9515 (2018)
2. Akin, O., et al.: Radiology data from the cancer genome atlas kidney renal clear cell carcinoma [tcga-kirc] collection. Cancer Imaging Arch. (2016)
3. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Advances in neural information processing systems. vol. 2, p. 7 (2002)
4. Arya, V., et al.: One explanation does not fit all: a toolkit and taxonomy of AI explainability techniques. arXiv preprint arXiv:1909.03012 (2019)
5. Bakr, S., et al.: A radiogenomic dataset of non-small cell lung cancer. Sci. Data **5**(1), 1–9 (2018)
6. Barnett, A.J., et al.: Iaia-bl: a case-based interpretable deep learning model for classification of mass lesions in digital mammography. arXiv preprint arXiv:2103.12308 (2021)
7. Borowa, A., Rymarczyk, D., Ochońska, D., Brzychczy-Włoch, M., Zieliński, B.: Classifying bacteria clones using attention-based deep multiple instance learning interpreted by persistence homology. In: International Joint Conference on Neural Networks (2021)
8. Chen, C., Li, O., Tao, C., Barnett, A.J., Su, J., Rudin, C.: This looks like that: deep learning for interpretable image recognition. arXiv preprint arXiv:1806.10574 (2018)

9. Chen, Z., Bei, Y., Rudin, C.: Concept whitening for interpretable image recognition. Nat. Mach. Intell. **2**(12), 772–782 (2020)
10. Ciga, O., Martel, A.L., Xu, T.: Self supervised contrastive learning for digital histopathology. arXiv preprint arXiv:2011.13971 (2020)
11. Decencière, E., et al.: Feedback on a publicly distributed image database: the Messidor database. Image Anal. Stereol. **33**(3), 231–234 (2014)
12. Dietterich, T.G., Lathrop, R.H., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artif. Intell. **89**(1–2), 31–71 (1997)
13. Ehteshami Bejnordi, B., et al.: Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. JAMA **318**(22), 2199–2210 (2017). https://doi.org/10.1001/jama.2017.14585
14. Feng, J., Zhou, Z.H.: Deep miml network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
15. Foulds, J., Frank, E.: A review of multi-instance learning assumptions. Knowl. Eng. Rev. **25**(1), 1–25 (2010)
16. Gelasca, E.D., Byun, J., Obara, B., Manjunath, B.: Evaluation and benchmark for biological image segmentation. In: 2008 15th IEEE International Conference on Image Processing, pp. 1816–1819. IEEE (2008)
17. Ghorbani, A., Wexler, J., Zou, J.Y., Kim, B.: Towards automatic concept-based explanations. In: Advances in Neural Information Processing Systems, pp. 9277–9286 (2019)
18. Hase, P., Chen, C., Li, O., Rudin, C.: Interpretable image recognition with hierarchical prototypes. In: Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, vol. 7, pp. 32–40 (2019)
19. Hoffmann, A., Fanconi, C., Rade, R., Kohler, J.: This looks like that... does it? Shortcomings of latent space prototype interpretability in deep networks. arXiv preprint arXiv:2105.02968 (2021)
20. Ilse, M., Tomczak, J., Welling, M.: Attention-based deep multiple instance learning. In: International Conference on Machine Learning, pp. 2127–2136. PMLR (2018)
21. Kim, B., et al.: Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV). In: International Conference on Machine Learning, pp. 2668–2677. PMLR (2018)
22. Kim, E., Kim, S., Seo, M., Yoon, S.: Xprotonet: diagnosis in chest radiography with global and local explanations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15719–15728 (2021)
23. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann, Burlington (2014)
24. Li, B., Li, Y., Eliceiri, K.W.: Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14318–14328 (2021)
25. Li, G., Li, C., Wu, G., Ji, D., Zhang, H.: Multi-view attention-guided multiple instance detection network for interpretable breast cancer histopathological image diagnosis. IEEE Access (2021)
26. Li, O., Liu, H., Chen, C., Rudin, C.: Deep learning for case-based reasoning through prototypes: a neural network that explains its predictions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
27. Lu, M.Y., Williamson, D.F., Chen, T.Y., Chen, R.J., Barbieri, M., Mahmood, F.: Data-efficient and weakly supervised computational pathology on whole-slide images. Nat. Biomed. Eng. **5**(6), 555–570 (2021)

28. Nauta, M., van Bree, R., Seifert, C.: Neural prototype trees for interpretable fine-grained image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14933–14943 (2021)
29. Nauta, M., Jutte, A., Provoost, J., Seifert, C.: This looks like that, because... explaining prototypes for interpretable image recognition. In: Kamp, M., et al. (eds.) ECML PKDD 2021. CCIS, vol. 1524, pp. 441–456. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-93736-2_34
30. Quellec, G., et al.: A multiple-instance learning framework for diabetic retinopathy screening. Med. Image Anal. **16**(6), 1228–1240 (2012)
31. Rani, P., Elagiri Ramalingam, R., Rajamani, K.T., Kandemir, M., Singh, D.: Multiple instance learning: robust validation on retinopathy of prematurity. Int. J. Ctrl. Theory Appl. **9**, 451–459 (2016)
32. Rebuffi, S.A., Fong, R., Ji, X., Vedaldi, A.: There and back again: revisiting backpropagation saliency methods. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8839–8848 (2020)
33. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
34. Rymarczyk, D., Borowa, A., Tabor, J., Zielinski, B.: Kernel self-attention for weakly-supervised image classification using deep multiple instance learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1721–1730 (2021)
35. Rymarczyk, D., Struski, Ł., Tabor, J., Zieliński, B.: Protopshare: prototype sharing for interpretable image classification and similarity discovery. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2021) (2021). https://doi.org/10.1145/3447548.3467245
36. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 618–626 (2017)
37. Selvaraju, R.R., et al.: Taking a hint: leveraging explanations to make vision and language models more grounded. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2591–2600 (2019)
38. Shao, Z., et al.: Transmil: transformer based correlated multiple instance learning for whole slide image classication. arXiv preprint arXiv:2106.00908 (2021)
39. Shi, X., Xing, F., Xie, Y., Zhang, Z., Cui, L., Yang, L.: Loss-based attention for deep multiple instance learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5742–5749 (2020)
40. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. arXiv:1312.6034 (2013)
41. Sirinukunwattana, K., Raza, S.E.A., Tsang, Y.W., Snead, D.R., Cree, I.A., Rajpoot, N.M.: Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. IEEE Trans. Med. Imaging **35**(5), 1196–1206 (2016)
42. Straehle, C., Kandemir, M., Koethe, U., Hamprecht, F.A.: Multiple instance learning with response-optimized random forests. In: 2014 22nd International Conference on Pattern Recognition, pp. 3768–3773. IEEE (2014)
43. Tu, M., Huang, J., He, X., Zhou, B.: Multiple instance learning with graph neural networks. arXiv preprint arXiv:1906.04881 (2019)
44. Wang, X., Yan, Y., Tang, P., Bai, X., Liu, W.: Revisiting multiple instance neural networks. Pattern Recogn. **74**, 15–24 (2018)

45. Yan, Y., Wang, X., Guo, X., Fang, J., Liu, W., Huang, J.: Deep multi-instance learning with dynamic pooling. In: Asian Conference on Machine Learning, pp. 662–677. PMLR (2018)
46. Yeh, C.K., Kim, B., Arik, S.O., Li, C.L., Pfister, T., Ravikumar, P.: On completeness-aware concept-based explanations in deep neural networks. In: Advances in Neural Information Processing Systems (2019)
47. Zhao, Z., et al.: Drug activity prediction using multiple-instance learning via joint instance and feature selection. BMC Bioinform. **14**, S16 (2013). Springer