# Explainability in Deep Learning by Means of Communication

DISSERTATION

von

STEPHAN ALANIZ

Tübingen

2022

# Explainability in Deep Learning by Means of Communication

DISSERTATION

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
STEPHAN ALANIZ
aus Groß-Gerau

Tübingen
2022

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation: 08.07.2022

| | |
|---|---|
| Dekan: | Prof. Dr. Thilo Stehle |
| 1. Berichterstatterin: | Prof. Dr. Zeynep Akata |
| 2. Berichterstatter: | Prof. Dr. Gerard Pons-Moll |

# ACKNOWLEDGEMENTS

My PhD has been an exciting journey during which I had the privilege of being part of a diverse and welcoming research community, which has led to meeting, collaborating, and making new friends with many amazing people along the way.

I would like to express my most sincere and deepest gratitude to my supervisors Prof. Zeynep Akata and Prof. Bernt Schiele for supervising and supporting me while giving me the freedom to work on the research problems that excite me. Prof. Zeynep Akata has been particularly influential for guiding me right from the beginning at the University of Amsterdam, helping me grow both as a researcher and as a person. Her dedication and strive are inspiring and facilitate an encouraging research environment, not only for me, but for the whole group at the University of Tübingen.

Thank you very much, Prof. Bernt Schiele and the whole D2 group at the Max Planck Institute for Informatics, for welcoming me after my move, enabling to quickly establish new relationships through the retreats and widening my views by introducing me to a whole new breadth of research topics.

I am also particularly thankful to Diego Marcos for both his positivity as well as critical feedback that pushed me on the right track numerous times. My gratitude goes to all of my other collaborators Anjan, Marco, Massi, Rudy and Thomas who not only helped me shape my research, but have also become friends along the way that I enjoy spending time with. I am lucky to have met many amazing colleagues during my PhD at AMLab, MPI, and the EML group in Tübingen, who were always available for intriguing discussions not only about research and who I shared a lot of memorable experiences with outside of work. A special shout-out goes to Massi and Thomas, not only for being a pleasure to work with, but also for being incredible chess, flat, and “club” mates during my time in Tübingen.

Last but not least, I cannot express enough how thankful I am to my girlfriend Anna for her constant encouragement and support as well as my friends and family, who stand by me even when I could not spend as much time with them as I would have liked.

# ABSTRACT

Research in deep learning has seen extraordinary advances at a fast pace leading to the emergence to many prospective applications, especially in the computer vision domain. Nonetheless, most of the machine learning pipeline remains opaque and hard to explain to humans, limiting their real-world use due to the lack of understanding and trust to deploy these systems in critical scenarios such as health care.

In this thesis, we propose machine learning models that are inherently more explainable than the base model they are derived from. By exposing parts of the decision-making process and increasing the overall transparency, we develop tools that allow humans to better evaluate strengths and weaknesses of the machine learning models as well as assess their suitability for deployment.

The research of this thesis takes inspiration from how explanations are formed between humans through communication. We present novel approaches that look at different aspects of the communication process and embed them into neural network models to make them more interpretable and/or more integrated into human-computer interactions. Communication — and to a greater extend human language — is a natural way for humans to compose explanations, such that explanation systems would not impose onto the users a learning curve of interacting with the artificial intelligence.

Specifically, we propose a multi-agent communication setting, where messages between the agents resemble a yes/no question-answer discourse. Agents trained to solve an image classification task, learn to build a decision tree that globally describes the prediction procedure.

We find that human-understandable side-information is key for making the framework truly explainable. While this approach finds one globally viable explanation for a given problem, humans are diverse in their own communication, languages, and perception. To this end, we build an agent that creates purposefully crafted messages by observing its communication partner in their effectiveness in acting upon the communicated data and adjusting its own policy accordingly. Then, we show that efficient communication can also be used as a metric that, when optimized for, leads to both highly compressed abstractions and interpretable insights on humanly drawn sketches and sketch-based tasks.

Finally, we take a broader look at how semantic information, e.g., from language, can enrich vision models and make them more explainable.

# ZUSAMMENFASSUNG

Die Forschung in Deep Learning erlebt herausragende Fortschritte, die zu vielen zukünftigen Anwendungen führen können, besonders im Bereich Computer Vision. Dennoch bleiben die meisten Teile eines auf maschinellem Lernen basierenden Systems verborgen und schwer dem Menschen zu erklären. Der daraus resultierende Mangel an Verständnis und Vertrauen beschränkt die Anwendung solcher Methoden in kritischen Szenarien wie dem Gesundheitssystem.

In dieser Dissertation schlagen wir Modelle vor, die von Grund auf erklärbarer sind als die Basismodelle, auf denen sie aufbauen. Durch das Offenbaren von Teilen des Entscheidungsprozesses entwickeln wir Hilfsmittel, die es Menschen erlauben, die Stärken und Schwächen der Modelle besser zu erkennen und einzuschätzen, ob sie sich für einen Einsatz eignen.

Die Forschung dieser Dissertation ist von der Art inspiriert wie Erklärungen durch Kommunikation entstehen. Wir präsentieren neuartige Herangehensweisen, welche verschiedene Aspekte des Kommunikationsablaufs in neuronale Netze integrieren, sodass sie interpretierbarer werden und/oder die Interaktion zwischen Mensch und Maschine verbessern. Kommunikation — und damit auch unsere Sprache — ist ein natürlicher Weg für Menschen eine Erklärung zu verfassen, sodass ein Benutzer von erklärbaren Systemen keiner Lernphase ausgesetzt ist, um mit der künstlichen Intelligenz interagieren zu können.

Insbesondere schlagen wir eine Multiagenten-Kommunikationsumgebung vor, bei der die Nachrichten zwischen den Agenten einem Ja/Nein Frage-Antwort-Diskurs ähneln. Agenten, die trainiert wurden, Bilder zu klassifizieren, bauen dabei einen Entscheidungsbaum, der den Ablauf der Vorhersagen als Ganzes beschreibt. Während dieser Ansatz eine einheitliche Erklärung für eine Problemstellung findet, stellen wir fest, dass Menschen vielfältig in ihrer Kommunikation, Sprache und Wahrnehmung sind. Aus diesem Grund erschaffen wir einen Agenten, der zielgerichtete Nachrichten erstellt und seine Vorgehensweise anpasst, indem er seinen Kommunikationspartner dabei beobachtet, wie effektiv dieser die übertragenen Informationen in Taten umsetzen kann.

Im Anschluss zeigen wir, dass effiziente Kommunikation auch als Maßstab verwendet werden kann, der, wenn man sich ihn als Ziel setzt, zu hoch komprimierten Abstraktionen und zu interpretierbaren Erkenntnissen führt. Schließlich werfen wir einen umfassenderen Blick darauf, inwiefern semantische Informationen, z.B. von Sprache, Vision-Modelle bereichern und sie erklärbarer machen können.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1

# INTRODUCTION

The usage of Artificial Intelligence (AI) has increased over the past years and keeps demonstrating a steady further growth: sophisticated models enter more areas and find more application cases, from entertainment and gaming [Vin+19] to urban planning and "artificially intelligent cities" [Yig+20]. Such a fast-paced inclusion into human society as well as its expanding influence over our everyday life makes AI face certain expectations and even demands.

On one hand, AI is awaited to reach a cognitive ability comparable to human's and, thereby, be able to perform both ordinary and specialized tasks at an equal or super-human level. In certain fields, this expectation is already being fulfilled. For instance, AlphaGo [Sil+16], "arguably the strongest Go player in history" [Dee22], is able to defeat a human grand master at the board game of Go, which was long believed to be one of the hardest AI challenges in game playing. Similarly, the most recent version MuZero [Sch+20] does so without ever seeing a human playing the game, i.e., only by playing against itself.

On the other hand, there is a growing need for users to be able to understand how exactly AI would make up its "mind". In other words, in addition to making correct decisions and solving problems successfully, the machine should be able to expose its "thinking" process. This way, humans could utilize it and develop a better understanding of why some steps are selected over others as well as how some choices result in a desired outcome while others do not. Such an explainability appears especially important for critical applications:

- A health-care system should not only predict a treatment, but also provide arguments in favor of its suggestions in an intelligible way, for doctors to analyze the situation and make informed final decisions.

- A self-driving car could indicate its uncertainty about proceeding in a given environment, e.g., a decision to overtake another vehicle, and specify what is causing the confusion, effectively allowing a driver to intervene and make a better decision, potentially avoiding a dangerous situation.

The ability to uncover the decision-making mechanisms and logic that led to a specific result would also be beneficial for cases of under- or nonperformance on either side:

- Having AlphaGo's moves together with its reasoning, human players could improve their own game and skills.
- When a chatbot failed to understand a human prompt and be helpful, specifying why miscommunication happened on its end, a machine learning practitioner could retrospectively debug the model and correct its behavior, to avoid facing the same problem in the future.
- If a biased model systematically puts a certain group of people at a disadvantage, seeing an explanation how the model arrives at its conclusions would help tweaking it for more balanced decisions.
- In general, shedding some light on why AI acts the way it does under given circumstances would help human users build more awareness and establish more trust in machines [MKR21].

Despite this seemingly clear objective, however, working on explainability in AI becomes anything but trivial. The biggest challenge here stems from the AI's decision-making process remaining hidden.

- AI does not really interact with a user, at least, not in a way usual for humans. Having fed a required input to the machine, a user could receive its final decision for that input, but the model's choices are not something that could be easily deduced from the architecture or intermediate outputs.
- The decision-making process itself is non-linear, and the large amount of parameters and layers only boost the overall complexity, impeding the understanding on the human side.

For example, winning the Go game is an impressive achievement, but Go is a deterministic environment, where each player has perfect information. Unlike Go, many real-world problems require AI to act in complex, uncertain, and probabilistic environments: in an ever-changing world and climate, autonomous cars need to deal with unforeseeable weather effects, AI-supported healthcare systems may be exposed to new diseases, chatbots must understand slang and be capable of handling possible speech- and audio-signal defects. This contributes to the general image of AI being a black box in the eyes of both an average consumer and a machine learning practitioner. As a result, a difficulty to understand translates into a difficulty to accept, integrate, and make use of.

Explainable AI [1] could fill in this gap by extending existing machine learning methods, such that they could reveal, how and why a decision or prediction is made, and do so in an interpretable manner.

Belonging to the said field of research, this thesis gets inspiration from human interaction and explanations and attempts to develop similar approaches for machines by modeling communication, enhancing non-interpretable model components, and experimenting with model architectures that would help disclosing interpretable information to a human, who then could extract valuable insights and further analyze the model's decision-making process.

[1] The research community does not seem to have unity regarding the terms Explainable and Interpretable AI: some scientists view explainability and interpretability as different concepts [Arr+20], while others do not make a separation [Zha+21]. In this work, both terms are used interchangeably.

## 1.1 Contributions

This thesis primarily deals with tackling explainable AI in the computer vision domain. It addresses communication as a concept and how several aspects of communication could enable novel introspections into deep models while fulfilling an explanatory role. Furthermore, our work extends the scope of explainable AI by establishing new use cases and enriching existing tasks.

Acknowledging the importance of natural language for communication and explanations, we adjust our work, for it to appropriately account for the peculiarities of computer vision. Since in this domain, the data usually consist of natural images, we experiment with sketches as a more abstract representation of human drawings of objects, include textual data to bridge concepts between different domains and use natural language as a means of communication.

A variety of tasks and use cases are considered in the course of work. We cover classification, retrieval, image referencing, weakly supervised object detection, and image generation. While improving the state of the art in terms of evaluation metrics for these tasks is not generally a primary goal of explainable AI, we pay attention to carefully considered trade-offs and limitations of our proposed methods while maintaining their applicability in challenging scenarios.

In the following sections, we discuss three lines of contributions.

**First, we build introspective models and representations.** To build more inherently interpretable models, we present Recurrent Decision Trees through Communication (RDTC) (Chapter 3), a framework of two neural-network agents communicating to solve an image classification task. By making the agents communicate in the form of binary questions and answers and using their communication as a discrete bottleneck, we make these agents jointly learn to solve a task iteratively, step by step. Communication functions as an interface for understanding the learned decision-making process.

Instead of only obtaining an output prediction, our model exposes a sequence of binary features responsible for causing the prediction. Due to the choice of binary communication, the global decision mechanism is equivalent to a decision tree for the whole dataset, where each image takes a single path through the tree, and leaf nodes contain the model's predictions. Such an approach allows to explain single instances, i.e., images, but also facilitates the extraction of clustering relations of the entire dataset. In addition to being more transparent in its decision-making process, our RDTC model retains the classification accuracy of the baseline black-box model it is derived from, on a variety of datasets including ImageNet.

In the sketch domain, we develop a Primitive Matching Network (PMN) (Chapter 5) that learns to match human-drawn strokes to primitive shapes under affine transformations. For instance, a common drawing of a human face consists of a circle for the outline, two straight lines for the eyes, and a curve for the mouth. While neural networks observe this data as a sequence of lines consisting of a sequence of points on a 2D canvas, our network matches frequently occurring patterns to primitive shapes, such as lines, rectangles, circles, etc. By replacing the original raw data with arrangements of primitives, we create a new representation of the data that retains the original information while being explicit and highly compressed. This allows us to qualitatively and quantitatively explore the data and distill structural pattern for each semantic class in an

unsupervised way.

While PMN creates new representations of the data in the original input space, representation-learning approaches usually learn a distinct latent space. With our Compositional Mixture Representations for Vision and Text (Section 6.1), we propose the CoMix model that learns a multi-modal alignment between images and sets of labels in a latent space. CoMix utilizes a spatial-transformer module that transparently assigns cropped image sections of the input to Gaussian components of the feature space. Through the compositional nature of this model, the latent space becomes interpretable while retaining expressibility by allowing multiple mixture components to correspond to separate objects in the image.

**Second, we establish communication tasks.** We develop a novel communication task in the form of an image-reference game that we use to model conceptual understanding of a speaker agent about populations of listener agents (Chapter 4). We motivate this task by drawing the analogy to adjusting explanations depending on the context and conceptual understanding of the receiver.

More specifically, a speaker is tasked with communicating the identity of a target image from a set of two images by pointing to a discriminative feature of the target image. The conceptual understanding of the features varies for every listener: e.g., if a listener is color-blind, the speaker has to adapt its communication, to be able to solve the task successfully. Further, we present a model that can predict these discrepancies between listeners to outperform sensible baselines.

PMN (Chapter 5) allows us to further abstract human sketches by matching and replacing human strokes with primitive shapes. We evaluate the effectiveness of this interpretable compression mechanism on sketch classification and sketch-based image retrieval by communicating sketches iteratively to a network performing the downstream task. Our experiments demonstrate that our primitive-based representation outperforms raw sketches as fewer communication steps are required to convey sketch semantics and achieve higher scores at low communication budgets. With this communication task, we showcase that primitive abstractions not only allow us to better understand how objects are composed of shapes, but also to efficiently tackle downstream tasks with compact representations.

**Third, we incorporate semantics.** A key part of human communication is language. It contains semantics that allows to draw associations with the world around us. When we let neural agents communicate in their own artificial language, extracting explanations for humans becomes challenging. With this point in mind, we align the products of our models with human-understandable concepts by incorporating semantic data into the training procedure.

Our RDTC framework (Chapter 3) learns to communicate from scratch and can discover visual concepts as binary features on its own. However, to make these features interpretable, a human would have to create associations by hand through inspection of the learned decision tree. Since, depending on the dataset, this is impractical and time consuming, we propose to use human-understandable attribute information as the vocabulary for the communication. Due to these additional constraints during training, the communication loop as well as the decision tree become inherently explainable.

The compositional representations of our CoMix (Section 6.1) model explicitly align latent

components with input regions in the image. At that point, components from the same region in the latent correspond to perceptually similar concepts in images but naming them would require human intervention. Therefore, we propose to align image latents with latents derived from an unordered set of textual labels describing the objects in the image, effectively transferring the semantics from the text to the image regions. With matching vision and text domains, CoMix can perform weakly-supervised object detection while generalizing to unseen combination of objects through its compositional latent structure.

Finally, we explore vector-quantized (VQ) models (Section 6.2) for their unique characteristic of learning discrete visual tokens which, when arranged in a grid, encode entire images. VQ models are a popular method for image synthesis as training a generative model on these sequences of tokens resembles a language model. To get a better association with semantic contexts and guide the generation process, we develop a semantically coupled VQ-model that leverages semantic maps as conditioning information. Through the tight integration of semantic information, we improve image synthesis results and allow fine-grained control over the generated content in an image.

## 1.2 Outline

This section provides a brief overview of every thesis chapter, referencing respective publications and collaborations including their contribution to the overall work. The content of the chapters corresponds to the published content. All the publications are first-author or shared first-author publications. Appendix C lists all the publications including their co-authors.

**Chapter 1: Introduction** motivates the work of this thesis and explains why researching explainable AI is necessary yet difficult to conduct. This chapter also describes the contribution of this thesis to the progress of the scientific community as well as contains an structural overview of the thesis in general.

**Chapter 2: Background** reviews the preliminary basis for this thesis. It discusses deep learning as a field and tasks involved in the subsequent chapters, gives an overview of related explainable AI research, and describes how communication fits into the scientific picture. Chapters following the Chapter 2 contain their own respective sections about related work, specific to the research topics they discuss.

**Chapter 3: Learning Decision Trees Recurrently Through Communication** introduces our RDTC model that utilizes a communication framework between two agents to solve an image-classification task. Explainability is obtained, on one hand, by learning a decision tree that describes a prediction structure of the communication loop, and, on the other, by restricting the vocabulary, for it to consist of human-understandable attributes. On both attribute and non-attribute datasets, we demonstrate that explainable models can maintain state-of-the-art performance while being more interpretable at the same time. This work was published at CVPR 2021 [Ala+21] in collaboration with Diego Marcos, who had an advisory role and assisted with setting up the user-study experiment.

**Chapter 4: Modeling Conceptual Understanding in Image Reference Games** proposes employing an image-reference game to scrutinize the variation in conceptual understanding of

a population of agents. We develop a neural-network agent that learns how to maximize the communication effectiveness by observing the task performance of its communication partners. The content of this chapter was published at NeurIPS 2019 [CAA19] as a shared-author paper together with Rodolfo Corona. Contributions were divided equally, with the specifications of the task being shaped more by the concepts of Rodolfo Corona and the reinforcement learning approaches as well as model integration being more influenced by the ideas of Stephan Alaniz.

**Chapter 5: Abstracting Sketches through Simple Primitives** covers our work on matching primitive shapes to human strokes in their sketch drawing. By replacing raw data with a more structured set of visual words, we could abstract sketches to a more compact representation while maintaining their visual appearance. With classification and sketch-based image retrieval, we present a model that enables a more efficient communication of semantic information to solve these tasks. The research from this chapter was published at ECCV 2022 [Ala+22]. The initial idea of abstracting sketches through primitive shapes was developed in close collaboration with the co-author Massimiliano Mancini. Stephan Alaniz guided developing the model and required losses to tackle the task. Massimiliano Mancini implemented some of the baselines and assisted with integrating datasets. Diego Marcos and Anjan Dutta contributed with their domain expertise in an advisory role.

**Chapter 6: Language-inspired Extensions beyond Communication** comprises two works that go beyond explainability through communication but are connected with their inspiration taken from semantics and the discrete nature of language respectively.

Firstly, we propose CoMix, a probabilistic model that learns compositional mixture representations from image-text pairs. By imposing a mixture of Gaussians distribution on the latent space, we can separate and match entities in both modalities such that object-containing image regions are aligned with their corresponding textual labels without direct supervision. Such an architecture allows performing multiple tasks, ranging from weakly-supervised object detection to image retrieval, and its compositional properties enable it to generalize to unseen combinations of objects at test time. The content of this work was published at the CVPR 2022 L3DIVU Workshop [AFA22] in collaboration with Marco Federici, who contributed to the loss design and implementation of the CoMix model.

Secondly, we present a semantic image synthesis with semantically coupled VQ-model, where we employ vector quantization to, first, learn discrete codes that make up images and then learn a model to generate them. We contribute by integrating semantic maps to outline the image structures into the network architecture and coupling the latents of both semantics and image, to make them more expressive together. Our experiments show an improved semantic synthesis of images by closely tying the generated content to the semantic maps. This work was published at the ICLR 2022 DGM4HSD Workshop [AHA22] together with Thomas Hummel as a shared first author. Both Stephan Alaniz and Thomas Hummel contributed equally to the design and implementation of the developed sVQ model. Stephan Alaniz took the lead by initially proposing novelties to vector-quantized modeling with the semantic conditioning information, which resulted in this work.

**Chapter 7: Discussion and Conclusion** completes the thesis and puts its results into perspective of the research field. We discuss the contributions, point to limitations of our current approaches, and suggest how they could be addressed in the future in the domain of explainable AI.

# 2

# BACKGROUND

This chapter briefly reviews current research in the deep learning field as we employ this background knowledge as well as certain assumptions and tasks as a foundation for our work. We also provide an overview of different approaches to explainable AI as it is utilized to enhance deep learning systems. Finally, we take a look at how communication is defined and describe how it becomes both a tool and motivation for our research.

## 2.1 Deep Learning and its Application

Deep learning is a subfield of machine learning, a discipline dealing with algorithms that learn from data to perform a task without a human specifically designing and implementing the solution by hand. For instance, in computer vision, processing of images and other high-dimensional data is required for many tasks. Thus, hand-designing filters and functions to detect the semantic content of an image becomes infeasible at scale.

A machine learning algorithm can learn a mapping function $f_{\text{cls}} : X \to Y$ directly from data, where $x \in X$ is an image and $y \in Y$ is a label we want the algorithm to predict.

A common approach is to supervise the learning of this function by providing a dataset of paired input-output samples $\{x_i, y_i\}_i^N$. The task of predicting categorical labels from images is also referred to as classification and is tackled as part of the thesis.

Classification tasks find various applications, especially in perception problems such as medical diagnosis or detection of other cars and pedestrians in self-driving cars. The latter is an extension to classification called object detection, where there is a label for each object in the image, and we are not only interested in classifying the objects, but also in identifying their location in the image.

Another relevant task in the context of the thesis is image retrieval, where the goal is to match images $X_{\text{img}}$ to a corresponding pairing from another modality $X_{\text{alt}}$ such as sketches or language.

While supervised pairing data is generally provided for retrieval tasks, the output of function $f_{\text{ret}} : X \to Z$ is often a previously unknown embedding space that captures similarities between the two domains. We also refer to $Z$ as the latent space because it is inferred from the observable data. The training objective is to place data pairs close in the latent space so that we can perform image retrieval by, first, embedding a query sample to $z_{\text{alt}} = f_{\text{ret}}(x_{\text{alt}})$ and then finding the closest

image $x_{\text{ret}} = \arg\min_{x_{\text{img}} \in X_{\text{img}}} d(f_{\text{ret}}(x_{\text{img}}), z_{\text{alt}})$ for some distance function $d$. Retrieval can be used for content discovery, search, and exploration of content across domains, for example, in commercial settings.

We also employ unsupervised learning, where paired data is not available, but only a single modality, e.g., a dataset of images. Unsupervised learning can be used to learn a latent space either for generative modeling or as a pre-text task since data acquisition is not as expensive as in supervised tasks, where labeling work is conducted by humans. Common training objectives for unsupervised tasks include reconstructing the original input or learning a probabilistic model, which approximates the true data-generating distribution by maximizing the likelihood, with which the model would generate samples from the dataset. Generative models are used for drug discovery and as an artistic tool in image processing, for example.

Deep learning provides tools to solve these learning tasks with a model class of neural networks as function $f$. At its core, neural networks are composed of parameterized linear transformations and non-linear activation functions. When chaining there two operations many times as neural network layers, we obtain powerful function approximators that can model complex relations in the data.

Stochastic gradient decent (SGD) [KW52; RM51] and backpropagation [RHW86] allow to efficiently train neural networks on large amounts of data, facilitating fast-paced progress in research. Since its inception, neural network architectures have made tremendous progress to develop both general and specialized designs. The Convolutional Neural Network (CNN) [LeC+89] alongside popular such architectures as ResNet[He+16] and InceptionV3 [Sze+16] have enabled steady advances in the computer vision field and are being used throughout this thesis. They have been successfully applied to classification [He+16; KSH12; Liu+22; SZ15; Sze+16], object detection [RF17; Ren+15], semantic segmentation [He+17; NHH15], zero-shot recognition [Xia+19] and image generation [Goo+15; KLA19].

Apart from images, we also work with sequential data, e.g., in the communication between agents. Neural networks have been purposefully designed to process sequential data, most prominently with the introduction of the Long Short-Term Memory (LSTM) [HS97] network. It incorporates a memory component that allows information from previous time steps to be retained while new data is sequentially observed. More recently, the Transformer [Vas+17] architecture has become the state-of-the-art by employing self-attention layers that can aggregate information across long time spans in sequential data. Their flexibility and expressiveness have not only established the Transformer in the natural language domain [Bro+20; Cho+22], but its design has been also adopted in computer vision [Dos+21; Liu+21; Tou+21] and inspired new architectures for CNNs [Liu+22].

## 2.2 Explainable AI

Most deep learning models are predominantly opaque and non-interpretable. The extraction of valuable insights from the trained model is non-trivial due to the large number of parameters, layers, and non-linearities these models comprise.

The importance of explanations for an end-user has been studied from the psychological perspective [Lom12; Mil19], showing that humans use explanations as a guidance for learning and understanding by building inferences and seeking propositions or judgments that enrich their prior knowledge about the goal in question. This creates a need for explainable AI to make deep models more understandable for humans. The research community has not converged to a unified definition of explainable AI, so in the context of this thesis, we define explanations as any human understandable insight into the deep learning model or its decision-making process.

Explainability has been growing as a field in computer vision and machine learning [Aga+21; Den+22; Hen+16; JN20; Par+18; Ped+20; Ped+19; Ram+20; Zin+17]. A common approach to making such models interpretable involves post-hoc explanation methods. The idea is to, first, train a non-interpretable model and then try to understand its internals. Hence, the neural network is treated as a black box, and its input and output spaces are analyzed.

One of the most widely used forms of visual explanation is a saliency map of feature attribution that highlights, which regions in the images contribute the most to the decision. Consider, for example, the task of classifying an image of a dog. The input to the network is an array of RGB pixels from the image, and the output of the network is the predicted probability distribution over all training classes including the target class "dog". The post-hoc explanation method now attempts to assign an attribution value to each pixel in the input image based on how much the pixel contributed (positively or negatively) to producing the probability value for "dog". Saliency maps are most commonly obtained either by backpropagating gradients [Bac+15; Sel+17; SGK17; SF19; SF21] or by perturbing the input image [PDS18; SK19; Zin+17].

For example, Grad-CAM [Sel+17] uses gradients from the target class to the final convolutional layer and combines them with the activations of that layer to create a coarse attribution map of input regions. RISE [PDS18] perturbs input images with random occlusion patterns and measures the change in the model's output, where larger changes constitute more important regions in the image. Richer explanations can be drawn from combining these saliency maps with part and concept detectors [ZNZ18; Zha+19], text generators [Hen+16], or the combination of the two [Hen+18].

Contrary to post-hoc explanations, introspection is achieved by modifying the neural network either through training the model to generate an explanation along with the result or by making its internal representation interpretable per se. Inherently interpretable models aim to build a neural network structure in a way that displays additional interpretable information about their inner workings faithfully. Sometimes these models come at a cost of reduced model capacity as these inherent structures pose a constraint on the expressiveness of the model class. For example, linear models and decision trees are generally considered to be interpretable models, however, these models are not universal approximators as opposed to neural networks [HSW89]. In return, input feature importance is directly exposed by linear models and decision trees break down the whole prediction process into interpretable steps.

In computer vision, for instance, one can use a local surrogate [RSG16], interpretable prototypes [Che+19a; Kim+18], local features [BB19], or dynamic alignment networks [BFS21; BFS22]. CoDA networks [BFS21] reformulate the convolutional layer such that the whole network can be described by a single linear transformation that changes based on the input image. Hence, CoDA

neworks integrate the faithful input attribution known from linear models into existing network architectures.

This thesis focuses on advancing built-in introspections as they provide better guarantees for the faithfullness of the explanation. We explore the means of communication that are either integral parts of the network's architecture or parts of the task, to make our models and their representations more interpretable.

## 2.3 Reinforcement Learning and Communication

The reinforcement learning problem [SB98] is defined by finding an agent policy that maximizes the expected amount of reward the agent receives from interacting with an observable environment. More formally, starting from its initial condition, the agent receives its first observation from the state of the environment. With each time step, the agent performs an action that impacts the environment and, as a return, receives a reward alongside the observation of the new state of the environment. This interaction between the agent and environment can be repeated indefinitely or ends when a so-called terminal state is encountered. The goal of the agent is to choose optimal actions such that the accumulated future rewards are maximized. This generally implies that the agent has to learn how to act optimally by evaluating its past experience gained from interacting with the environment. One key difference of reinforcement learning when compared to other forms of machine learning is the absence of a dataset to train a model on. Instead, reinforcement learning can be considered an online learning problem because the agent gathers data from the environment simultaneously to learning and updating its policy.

A simple policy is, for instance, $\epsilon$-greedy, where the agent estimates the expected future return for every action and selects the action with the highest score with the probability $1-\epsilon$ and a random action otherwise. Choosing a random action every now and then is important as the agent needs to explore the environment to obtain new experience. More sophisticated algorithms are often based on policy gradient methods [Sut+00] such as the REINFORCE algorithm [Wil92]. In addition to acting in an environment, algorithms developed in the field of reinforcement learning can also be used for solving other problems, where the solution set is discrete, e.g., it has been successfully applied to tackle vehicle-routing problems [KHW19]. Hence, it is also frequently used to study communication and the emergence of language [Cao+18; Far+22; Foe+16; HT17; LB20; Laz+18; LPT20]. However, the setup for learning and training communicating agents differs across the studies as the goal they are trying to achieve varies.

This thesis views communication as a bi-directional interaction between two neural network agents. They communicate by exchanging information in the form of discrete tokens that come from a vocabulary shared and understood by both agents. Depending on the task and goal, an agent may communicate one or more tokens at a time before the second agent responds. Generally, the communication continues in a loop until a stopping criteria is reached with the exact details of the communication protocols being discussed in the respective chapters. It's worth mentioning that they have in common that agents have different roles during the communication, e.g., one

agent asks questions and the other provides answers. Such a distinction draws similarities to the imbalanced communication between an explainer and the explainee.

The communication problem also exists in the field of information theory as coined by Claude Shannon in his seminal work on the mathematical theory of communication [Sha48]. In his context, communication is about sending a message from a source over a channel with limited capacity and reconstructing the message at the receiver's side. While the work on communication in this thesis is not concerned with the problem of transmitting arbitrary signals (e.g., continuous data over noisy channels), we can still put our definition of communication into perspective. According to Shannon's understanding, our communication systems are discrete and noiseless. The discrete tokens we use come from a predefined vocabulary, e.g., words from the English language (Chapter 3 & 4), and the average information content of each token is bounded proportionally to the size of the vocabulary. Each token, however, can transmit a different amount of information for the task at hand and might not use the channel capacity optimally. In some cases, we allow the discovery of tokens, so that the neural network could learn to maximize the transmitted information (Chapter 3 & 6.2), although, this comes at the cost of explainability. This is because these tokens become harder to decode by humans. An encoding of a message could make better use of the channel capacity, and we explore this property of communication in Chapter 5, where we re-encode sketches with a new set of tokens.

# 3

# Learning Decision Trees Recurrently Through Communication

Integrated interpretability without sacrificing the prediction accuracy of decision making algorithms has the potential of greatly improving their value to the user. Instead of assigning a label to an image directly, we propose to learn iterative binary sub-decisions, inducing sparsity and transparency in the decision making process. The key aspect of our model is its ability to build a decision tree whose structure is encoded into the memory representation of a Recurrent Neural Network jointly learned by two models communicating through message passing. In addition, our model assigns a semantic meaning to each decision in the form of binary attributes, providing concise, semantic and relevant rationalizations to the user. On three benchmark image classification datasets, including the large-scale ImageNet, our model generates human interpretable binary decision sequences explaining the predictions of the network while maintaining state-of-the-art accuracy.

## 3.1 Introduction

The decision mechanism of deep Convolutional Neural Networks (CNNs) is often hidden from the user, hindering their employment in critical applications such as health-care, where a thorough understanding of this mechanism may be required. The aim for analyzing the decision mechanism, i.e. *introspection*, is to reveal the internal process of the decision maker to a machine learning practitioner or user [Par+18]. However, models offering explanations through introspection may result in a performance loss [GA19; Mar+04].

Incorporating recent advances in multi-agent communication [Foe+16], we formulate the decision process as an iterative decision tree and embed its structure into the memory representation of a Recurrent Neural Network (RNN). Our model uses message-passing [HT17] with discrete symbols from a vocabulary. A tunable parameter controls whether to learn this vocabulary from scratch or to map it to human-understandable attributes assigning a meaning to every decision to improve its interpretability. Further, encoding the decision tree into the memory of an RNN retains

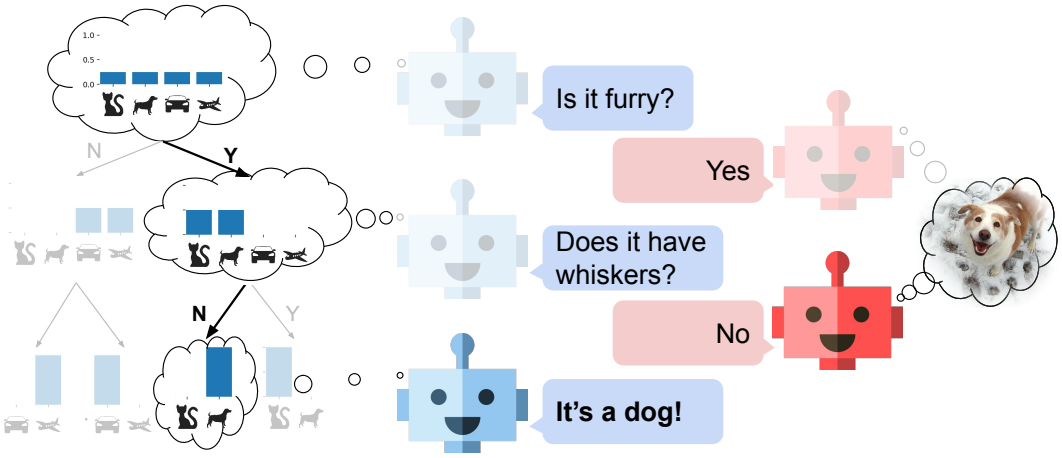



Figure 3.1: Our *Recurrent Decision Tree (RDT)* (left) asks questions, *Attribute-based Learner (AbL)* (right) answers with a yes/no s.t. the accuracy improves after each step.

the flexibility and performance of CNNs while being scalable. Instead of requiring an exponential number of tree nodes with increasing depth, our model learns orders of magnitude fewer nodes with a constant number of model parameters for an arbitrary tree depth. After training, our neural model can be converted exactly into a standard decision tree, being computationally efficient at test time.

Our framework (see Figure 3.1) exposes a decision path in the form of an explainable decision chain by breaking down the decision process into multiple binary decisions. Our recurrent decision-tree (RDT) (blue) does not see the image, and has to infer the image class, e.g. *dog*, by recurrently asking binary questions, e.g. *does it have whiskers?* Our attribute-based learner (AbL) (red) answers these questions with yes/no by looking at the image, allowing the RDT to update the class probabilities and the memory representation of the previous questions and answers. This is repeated until the RDT reaches a final decision and the decision tree becomes easily understandable as it associates the binary answers with semantic attributes, e.g. *has whiskers*.

Our contributions are: 1) We propose a recurrent decision tree model (RDTC) with hard node splits and overcome current limitations of decision trees in terms of depth scalability and flexibility; 2) We predict attributes in an end-to-end manner allowing human-interpretable explanations; 3) We showcase on three datasets that our model generates explainable decision trees more efficiently than related methods while retaining the performance of non-explainable CNNs. Our code is publicly available at: `https://github.com/ExplainableML/rdtc`.

## 3.2 Related Work

**Decision Trees with Neural Networks.** Decision trees are used across many machine learning tasks and applications, including medical diagnosis [AE13; Kon01], remote sensing [FB97; Han+00] and judicial decision making [Kle+18]. They make no assumptions on the data, and are inherently interpretable [Huy+11].

To improve their performance, combining decision trees with neural networks has been explored by building hierarchical classifiers [BD20; Mur+16; Zha+17a; Zha+17b], by transferring models [FH17; HVD15; HPM19; Siu19], and through regularization [Wu+18]. Recently, [Kon+16; Tan+19; Wan+21] have proposed learning decision trees directly with neural networks. NBDT [Wan+21] constructs trees in the weight space of a neural network and Adaptive Neural Trees [Tan+19] directly model the neural network as a decision tree, where each node and edge correspond to one or more network modules. The prior work closest to ours is the dNDF [Kon+16], which first uses a CNN to determine the routing probabilities on each node and then combines nodes to an ensemble of decision trees that jointly make the prediction. Our method differs in that 1) we focus on explainability by explicitly only considering a hard binary decision and 2) the depth and branching structure of our decision trees is learned by an RNN instead of being fixed a priori.

**Multi-Agent Communication.** Learning to communicate in a multi-agent setting has gained interest with the emergence of deep reinforcement learning [Cao+18; CAA19; Das+19; Foe+16; HT17; JL18; LB20; Laz+18; LPT20]. Most works focus on establishing a novel communication protocol from scratch. [Foe+16] and [Cao+18] train multiple agents to maximize a shared utility by establishing their own language. However, large scale multi-agent settings can suffer from too much communication, as valuable information comes with extensive computations [JL18]. Targeted communication focuses on key information and allows iterative exchange of information before performing a task that can improve both performance and interpretability [Das+19].

Image reference games are used to study the emergence of language [Laz+18] and effectiveness in communication also when concepts are being misunderstood [CAA19]. [HT17] propose an agent that composes a message of categorical symbols to another agent that uses the information in these messages to solve a referential game. Our model in contrast allows both to learn a communication protocol from scratch or use human-understandable concepts as a vocabulary.

**Attributes.** Attributes are human understandable visual properties of objects that are shared between classes. Attributes have been used for image description [CGG12; FZ08], caption generation [OKB11], face recognition [CGG13], image retrieval [KBN08; SFD11], action recognition [Yao+11; ZW12], novelty detection [WB13] and object classification [Che+19b; LNH14; MSN11; SQL12; Shi+14]. In this work we propose to use attributes as explanations, i.e. they label the branches in the learned decision tree, allowing users to easily inspect the reasoning encoded by the tree.

**Explainability through sparsity.** Optimizing representations to be sparse [ZLJ16] when seeking interpretability [Wri+10] draws some resemblance with the working memory of humans [MHB14], which is limited to a handful of items at the same time. [DK17] hypothesizes that the nature of these items (they need to be understandable *per se*), their number and the structure in which they are presented all impact the interpretability of a representation. Furthermore, interpretability can be achieved by regularizing neural networks such that their representations, not only to become sparse [Mar+20], but also adopt the structure of a decision tree [Wu+18].

Although both sparsity and tree depth have been used as proxies for interpretability in decision trees, human studies suggest that the best proxy is problem-dependent [Lag+18]. Beyond

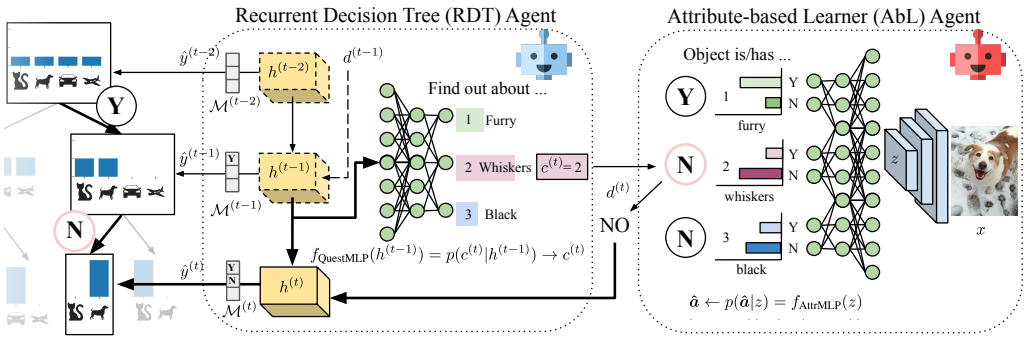



Figure 3.2: A single communication step between the RDT (left) and AbL (right) in our RDTC framework. RDT uses the hidden state $h^{(t-1)}$ of its LSTM (yellow) to requests a single attribute $a_{c^{(t)}}$ by selecting it through its $f_{\text{QuestMLP}}$. AbL uses its $f_{\text{AttrMLP}}$ to predict a binary response $d^{(t)} = \hat{\mathbf{a}}_{c^{(t)}}$ indicating the presence/absence of the attribute. Finally, RDT updates its state $h^{(t)}$ and explicit memory $\mathcal{M}^{(t)}$ with the binary response to improve its classification prediction $\hat{y}^{(t)}$.

explainable ML, a sparse representation is considered to be essential for moving towards hybrid deep learning-symbolic models [Cra+20; Mar20] and for obtaining representations that are closer to conscious reasoning [Ben17]. Indeed, a recent model of how human brains work postulate a conceptualization step, linked to dimensionality reduction, followed by an attention mechanism that sparsely selects concepts [CDK19].

## 3.3 RDTC Framework

Our Recurrent Decision Tree via Communication framework is a sequential interaction between the Recurrent Decision Tree (RDT) and Attribute-based Learner (AbL) models trained to classify images by communicating (see Figure 3.2). RDT learns a decision path allowing introspection and AbL provides attribute-based rationales to make the communication human-understandable.

### 3.3.1 Communication between RDT and AbL

For any single image $x$, our RDT model iteratively aggregates information into an explicit memory $\mathcal{M}$ that is sufficient to predict the correct class label $y \in \mathcal{Y}$. Initially, it starts with no prior information $\mathcal{M}^{(0)}$. To gather more information, the RDT agent sends a query message $c^{(t)}$ to the AbL agent. The AbL answers the query $c^{(t)}$ with a binary response $d^{(t)} \in \{0, 1\}$ that RDT uses to update its explicit memory $\mathcal{M}^{(t)} = \mathcal{M}^{(t-1)} \oplus (c^{(t)}, d^{(t)})$ to improve its class prediction. This constitutes one iteration $t$ of the agent-to-agent communication. The interaction repeats until a maximum number of steps is reached or until convergence.

**Communication Protocol.** The vocabulary size $|A|$ is set to the total number of attributes for every dataset. RDT and AbL learn to communicate with the set of tokens provided by the vocabulary in an end-to-end manner. Note that, the AbL agent attaches a human-understandable meaning to these tokens when annotated attribute data is available.

At each communication step $t$, RDT chooses one attribute $a_{c^{(t)}}$ from the vocabulary, identified by its index $c^{(t)}$, and requests its presence or absence in the image. AbL then provides its binary

prediction of this attribute, i.e. $d^{(t)}$. We deliberately limit the messages of AbL to be binary as clear yes/no answers are easier to interpret.

**Discrete Messages.** RDT asks for the attribute via the index $c^{(t)}$ and the AbL responds with a binary $d^{(t)}$. The Gumbel-softmax estimator [JGP17; MMT17] allows to sample from a discrete categorical distribution via the reparameterization trick [KW14; RMW14] to obtain the gradients of this sampling process. We sample $g_i$ from a Gumbel distribution and then compute a continuous relaxation of the categorical distribution:

$$\text{GumbelSoftmax}(\log \boldsymbol{\pi})_i = \frac{\exp((\log \pi_i + g_i)/\tau)}{\sum_{j=1}^{K} \exp((\log \pi_j + g_j)/\tau)} \tag{3.1}$$

where $\log \pi$ are the unnormalized log-probabilities of the categorical distribution, $\tau$ is the temperature that parameterizes the discrete approximation. When $\tau \approx 0$, the output is a one-hot vector and otherwise, it is a continuous signal.

Stochasticity is important for exploring all possible indices $c^{(t)}$ of vocabulary $A$ to find the most relevant attribute at each step $t$. Therefore, we use Gumbel-softmax with $K = |A|$ to sample the attribute index $c^{(t)}$ for RDT. As each $d^{(t)}$ corresponds to the presence or absence of an attribute in $x$, a deterministic prediction is beneficial. By introducing a temperature $\tau$ to a regular softmax [HVD15] in AbL, we approximate the $\arg\max$ function deterministically as $\tau$ approaches 0:

$$\text{TempSoftmax}(\log \boldsymbol{\pi})_i = \frac{\exp(\log \pi_i/\tau)}{\sum_{j=1}^{K} \exp(\log \pi_i/\tau)} \tag{3.2}$$

Since we use binary attributes, in this case $K = 2$. Popular training strategies include (a) annealing $\tau$ over time and (b) augmenting the soft approximation with an $\arg\max$ that discretizes the activation in the forward pass and results in the identity function in the backward pass. Using (b) guarantees the communication to be always discrete.

### 3.3.2 Recurrent Decision Tree (RDT) Model

RDT consists of three parts: an explicit memory $\mathcal{M}$, an LSTM [HS97], and a question-decoder module, *Question MLP* (see Figure 3.2 (left)). $\mathcal{M}^{(t)}$ contains all the binary attributes, i.e. the responses of AbL $d_{1:t}$ up to step $t$. The LSTM keeps track of the attribute order with its hidden state $h^{(t)}$ to encode the current point in the decision tree and decide on the next question. RDT decodes its last hidden state $h^{(t-1)}$ into a categorical distribution via $f_{\text{QuestMLP}}$:

$$\log p(c^{(t)}|h^{(t-1)}) = f_{\text{QuestMLP}}(h^{(t-1)}) \tag{3.3}$$

where $p(c^{(t)}|h^{(t-1)})$ indicates the likelihood of requesting a particular attribute. We denote the attribute index $c^{(t)} \in \{1, \ldots, |A|\}$ sampled by:

$$c^{(t)} = \text{GumbelSoftmax}(f_{\text{QuestMLP}}(h^{(t-1)})). \tag{3.4}$$

After each iteration of the communication loop, RDT updates its explicit memory $\mathcal{M}^{(t)} = \mathcal{M}^{(t-1)} \oplus (c^{(t)}, d^{(t)})$. Concretely, $\mathcal{M} \in \{0, 1\}^{|A| \times 2}$ is initialized with all zeros and at each time

step, we set $\mathcal{M}_{c^{(t)},d^{(t)}} := 1$. Encoding the attribute in a one-hot vector helps to indicate missing information with all zeros. $\mathcal{M}^{(t)}$ keeps track of already observed attributes and their values. RDT updates $h^{(t)}$ with:

$$h^{(t)} = \text{LSTM}(h^{(t-1)}, \mathcal{M}^{(t)}, c^{(t)}, d^{(t)}). \tag{3.5}$$

and at each time step $\mathcal{M}$ is used to predict the class label:

$$\hat{y}^{(t)} = f_{\text{ClassMLP}}(\mathcal{M}^{(t)}). \tag{3.6}$$

Since the primary objective of RDT is to maximize the classification performance, we minimize the CE loss between the predicted and the true class probabilities:

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{CE}(y, \hat{y}^{(t)}) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i} y_i \log \hat{y}_i^{(t)}. \tag{3.7}$$

By averaging the $\mathcal{L}_{CE}$ over all $T$ time steps, RDT predicts the correct class in a small number of communication steps which also allows it to be evaluated at any intermediate step unlike most other decision tree models that classify only at the leaf nodes (see supplementary for a comparison).

Since our decision tree can be evaluated after every communication step, the depth of the tree is not a fixed hyperparameter, but can be adaptively chosen at test time. This provides a flexible model that can be tuned for higher interpretability (shallow tree) or higher performance (deeper tree) at test time without the need for retraining.

### 3.3.3 Attribute-based Learner (AbL) Agent

The AbL feeds its CNN image features $z$ to $f_{\text{AttrMLP}}$ to predict a set of learned binary attributes queried by the RDT (Figure 3.2, right) where softmax with temperature gives us binary attributes $\hat{\boldsymbol{a}} \in \{0,1\}^{|A|}$, the discretization of $p(\hat{\boldsymbol{a}}|z)$:

$$\hat{\boldsymbol{a}} = \text{TempSoftmax}(f_{\text{AttrMLP}}(z)). \tag{3.8}$$

When the RDT requests the attribute with the index $c^{(t)}$, the AbL simply returns the binarized response about the attribute using $c^{(t)}$, i.e. $d^{(t)} = \hat{\boldsymbol{a}}_{c^{(t)}}$. The attributes are either discovered in an end-to-end manner by optimizing the loss in Equation 3.7 (`RDTC`, i.e. Recurrent Decision Tree via Communication) or they are predicted as human-interpretable concepts using an attribute loss (`aRDTC`, i.e. attribute-based Recurrent Decision Tree via Communication).

**Attribute Loss.** Minimizing the classification loss at each time step is equivalent to finding a binary data split that reduces the class-distribution entropy the most, i.e. information gain in classical decision trees. However, a split that best separates the data is not always easy to interpret, especially when the features used for this split result from a non-linear transformation as in a CNN.

We propose to integrate further interpretability by learning $\hat{\boldsymbol{a}}$ that align with class-level human-annotated attributes $\alpha$ using a second cross-entropy term weighted by $\lambda$:

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\Big[(1-\lambda)\mathcal{L}_{CE}(y, \hat{y}^{(t)}) + \lambda\mathcal{L}_{CE}(\alpha_{y,c^{(t)}}, \hat{\boldsymbol{a}}_{c^{(t)}})\Big]. \tag{3.9}$$

**Algorithm 1** RDTC decision tree distillation

**Input:** Training images $X$
Stopping *threshold*
**Output:** Decision tree DT

DT = empty decision tree
**for** $x$ in $X$ **do**
  DT.reset_to_root_node()
  $\hat{\boldsymbol{a}} = \text{AbL}(x)$ # attributes from image
  **for** $t = 1$ to $n$ **do**
    $\hat{y}^{(t)}, c^{(t)} = \text{RDT.step}(d^{(t)})$ # class/attribute of node
    **if** not DT.node_exists() **then**
      DT.add_node($\hat{y}^{(t)}, c^{(t)}$)
    **end if**
    **if** $\max_i \hat{y}_i^{(t)} >$ *threshold* **then**
      break # prune when confident
    **end if**
    $d^{(t)} = \hat{\boldsymbol{a}}_{c^{(t)}}$ # attribute yes/no
    DT.to_next_branch($d^{(t)}$) # 1 $\rightarrow$ left; 0 $\rightarrow$ right
  **end for**
**end for**
**return** DT

Note that the attribute loss is imposed only on those attributes employed by the model. If an attribute is deemed not to be useful, e.g., if an attribute is weak or hard to predict, our RDT model learns to ignore that attribute.

When $\lambda > 0$, our model (aRDTC) learns to use ground-truth attributes and gives the binary splits a semantic meaning. For instance, the question of RDT for attribute with index $c^{(t)}$ can be interpreted as "does it have a black beak?" with $a_{c^{(t)}}$: "has black beak". When $\lambda = 0$, RDTC does not use any human-annotated attributes and automatically discovers them. Either of these settings may be desirable given the application as we show empirically.

### 3.3.4 Decision Tree Distillation

The RDT and AbL are trained end-to-end since the communication between these two models is differentiable. At test time we distill the RDT into an explicit decision tree, i.e., the global structure of nodes, including splitting feature and threshold. The distilled decision tree then models the trained neural network $f_{\text{RDT}} \equiv f_{\text{DT}}$.

Algorithm 1 describes the procedure of extracting a decision tree from RDT. The decision nodes of the decison tree make hard splits based on the presence/absence of an attribute. Gumbel-Softmax adds stochasticity to RDT, which is useful for training, but at test time deterministically choosing the attribute with highest probability is essential for improving the performance and learning a static tree. Hence, it is replaced with TempSoftmax.

We start with an empty decision tree and fill it with nodes as we run the training data through

the whole RDTC model (lines 1-2). Whenever a previously unseen node is discovered, we add it to the tree including information about the attribute ($c^{(t)}$), where the next node is added, i.e., left of the current node if $d^{(t-1)}$ is 1 or to the right if $d^{(t-1)}$ is 0 (line 14), and the current prediction of the class labels $\hat{y}^{(t)}$ (lines 7-9). We prune the distilled decision tree, i.e. we stop adding nodes to the tree once $\hat{y}$ is greater than a threshold (=0.95) for a class (lines 10-12). The end result is a decision tree that outputs the same class predictions as our trained neural network RDT given the attribute prediction from our AbL, while being fully explainable by matching learned attributes with human-annotated attribute data.

## 3.4 Experiments

**Datasets and attributes.** We validate our model on the large-scale ImageNet [Rus+15] with 1.2M images from 1K classes. In addition, we use AWA2 [LNH14; Xia+19] and CUB [Wah+11], i.e. two medium-scale benchmark attribute datasets. AWA2 comprises 37K images from 50 animal classes with 85 attributes, while CUB contains 11K images from 200 fine-grained bird species with 312 attributes. Since our model considers splits on hard decisions, we binarize the attributes on all datasets with a threshold at 0.5, i.e., an attribute is present if more than 50% of the annotations agree. When an official classification test set is not provided, for all experiments across the datasets, we randomly assign 20% of each class as test data and 10% of the training data as a validation set to tune hyperparameters.

**Architecture and parameters.** The MLPs consist of two layers with a ReLU non-linearity. We learn the temperature $\tau$ of the Gumbel-softmax estimator jointly with the network from an initial value for $\tau$. During training, we always roll out the decision sequence to a maximum number of steps. At test time, we apply our decision tree distillation and stop as soon as the RDT reaches a confidence level specified by a *threshold* parameter (or once the maximum number of decisions is reached). We report the mean per-class accuracy over 5 runs to avoid bias towards highly populated classes.

### 3.4.1 Comparing with the State of the Art

We compare our `aRDTC` and `RDTC` with classical decision trees (aDT and DT) as baselines, ResNet (ResNet [He+16] and aResNet) and Deep Neural Decision Forests (dNDF) [Kon+16] as the state of the art.

**ResNet and aResNet.** ResNet-152 pre-trained on ImageNet and fine-tuned on each of the datasets including its softmax classifier serves as non-explainable deep neural network (ResNet). Augmented with attribute data, we train aResNet by first predicting the attributes with the same architecture as our AbL model and then a linear layer on top.

**Our `aRDTC` and `RDTC`.** Our attribute-based recurrent decision tree (`aRDTC`) (Section 3.3.3) uses the attribute loss to associate a human-understandable meaning to the binary decisions. On the other hand, our recurrent decision tree (`RDTC`) does not use an attribute loss ($\lambda = 0$), and therefore purely optimizes classification performance.

| Model | **AWA2** | **CUB** | **ImageNet** |
|---|---|---|---|
| ResNet [He+16] | $98.2 \pm 0.0$ | $79.0 \pm 0.2$ | $73.0 \pm 0.1$ |
| aResNet | $98.3 \pm 0.0$ | $77.3 \pm 0.5$ | N/A |
| DT | $92.3 \pm 0.4$ | $43.5 \pm 0.3$ | $55.2 \pm 1.0$ |
| dNDF [Kon+16] | $97.6 \pm 0.2$ | $73.8 \pm 0.3$ | $72.6 \pm 0.1$ |
| RDTC (Ours) | $\mathbf{98.0} \pm 0.1$ | $\mathbf{78.1} \pm 0.2$ | $\mathbf{72.8} \pm 0.1$ |
| aDT | $97.9 \pm 0.9$ | $70.6 \pm 1.3$ | N/A |
| aRDTC (Ours) | $\mathbf{98.1} \pm 0.0$ | $\mathbf{77.9} \pm 0.6$ | N/A |

Table 3.1: Comparing our aRDTC ($\lambda = 0.2$) and RDTC ($\lambda = 0$) to the decision tree (aDT and DT), closely related dNDF [Kon+16], and ResNet [He+16] (aResNet, i.e. ResNet with attribute prediction). As ImageNet do not have attributes, aResNet, aRDTC and aDT are not applicable (over 5 runs).

**dNDF.** The dNDF explicitly models the decision tree by mapping each inner node to an output neuron with sigmoid activation. These nodes define the routing probabilities of the input to the leaves through exhaustive tree traversal where each leaf node stores a class distribution. The final prediction is the averaged class prediction weighted by the routing probabilities of every leaf. As using multiple randomized trees weakens the interpretability, for a fair comparison, we use a single tree instead of random forests.

**aDT and DT.** The classical decision tree (DT) is learned on top of the same image features $z$ by the perceptual module. At each time step, the dataset is split using a single dimension of $z$ until a leaf node only contains samples of the same class or a regularization strategy leads to early stopping. We incorporate attributes into the DT baseline, i.e. Attribute Decision Tree (aDT). First, we train a MLP on top of the image features $z$ to predict class attributes using a binary cross-entropy loss analogously to the attribute loss of our aRDTC model. Second, we fit a decision tree on these predicted attributes for each image to determine the class. Both DT and aDT are learned using the CART algorithm [Bre+84] and the Gini impurity index as splitting criterion due to its computational advantage over entropy-based methods [RS04].

**Classification results.** As observed in Table 3.1, compared to the Decision Tree baselines of their kind, our model variants achieve significantly higher accuracy across all datasets, e.g. RDTC vs DT achieves $98.0\%$ vs $92.3\%$ and aRDTC vs aDT achieves $98.1\%$ vs $97.9\%$ on AWA2 because our model scales better and reaches consistent results through gradient-based optimization. Moreover, although RDTC and aRDTC work with constrained single-bit communications to improve explainability, they succeed in maintaining the accuracy of the non-explainable state-of-the-art across all datasets, e.g. $72.8\%$ vs $73.0\%$ on ImageNet.

Fine-grained decision splits are extremely challenging to explain because objects are visually similar to each other and the distinguishing factor is nuanced. Despite this challenge on CUB, the classification accuracy of RDTC is almost twice as high than classical decision trees that use the same deep features, i.e., $78.1\%$ vs $43.5\%$ DT. On the other hand, our RDTC not only outperforms dNDF ($78.1\%$ vs $73.8\%$), our model exhibits improved interpretability, because we use hard instead of soft binary splits. As it is hard for non-experts to judge the correctness of

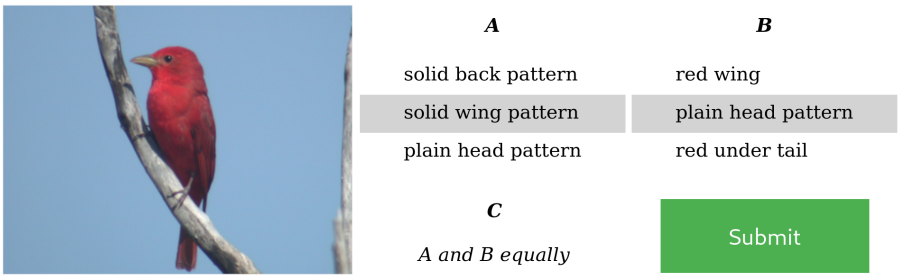
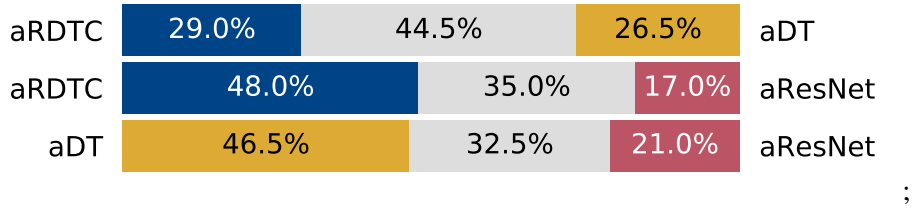



Figure 3.3: The user picks which set of 3 attributes best fit the image or if they match equally well (attributes come from 2 models out of aRDTC, aDT, aResNet at a time).




;

Figure 3.4: User study results. We show how often the attributes of one model were preferred over any other and when both were found equal (middle).

the predictions, explanations in this domain are particularly important. Typically, associating a semantic meaning to the decision path improves human interpretability with a significant loss in accuracy, e.g. aResNet vs ResNet (77.3% vs 79.0%). On the other hand, on our model this trade-off is less pronounced. When trained with the attribute loss, i.e. `aRDTC` achieves a higher accuracy compared to aDT (77.9% vs 70.6% on CUB) as well as aResNet in addition giving a semantic meaning to the splits.

**User study.** The use of named attributes enables humans to understand the decision of the model. However, if all attributes are allowed to be used simultaneously, such as in aResNet, this decision becomes less comprehensible. In contrast, aRDTC provides a sparse solution that considers only a subset of attributes for each prediction. To quantify the relevance of the selected attributes, we perform a user study with aRDTC, aDT and aResNet on CUB. Since our aRDTC predicts the class label at each step, we select the attributes that change the class probability the most to determine the most critical attributes for the decision. For aDT we use the Mean Decrease Impurity (MDI) [Lou+13] to find features of maximum importance and for aResNet we select the attributes with the highest weight for the output class.

The user is prompted with an image as well as two sets of three attributes, i.e., the three most relevant attributes from two models at a time. As some attributes are difficult to recognize, e.g. cone beak, we provide attribute icons with their names and a bird anatomy sketch. The task is to select the set of three attributes that best match the image (see Figure 3.3). The user can also report that both sets of attributes fit the image equally well. We repeat the study on 600 randomly selected images from the CUB test such that each model is compared 200 times against every other model.

We measure how often the attributes of each model are chosen over the other models. Since we only show attributes of two models at a time, we obtain a direct comparison for all pairs of models.

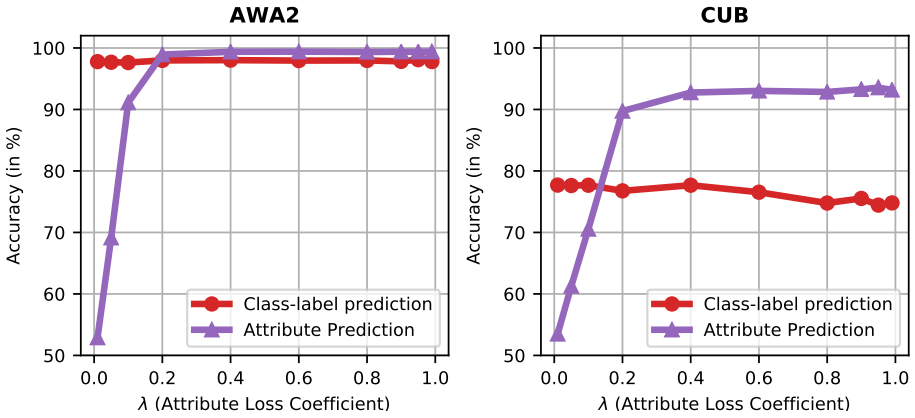



Figure 3.5: Explainability trade-off with aRDTC on AWA2 and CUB. We vary $\lambda$ of the attribute loss and report image classification accuracy of RDT (red) and attribute prediction accuracy of AbL (purple). $\lambda \in [0.01, 0.99]$.

Our results in Figure 3.4 indicate that decision tree models select more relevant attributes than aResNet. The attributes of aRDTC are preferred much more often than aResNet ($48\%$ vs. $17\%$). Similarly, aDT is selected more often than aResNet ($46.5\%$ vs. $21\%$). When comparing the two decision tree models, our aRDTC is slightly favored at $29\%$ over aDT at $26.5\%$ with the majority of users finding them produce equally fitting attributes ($44.5\%$). These results suggest that the tree structure of the decision making also helps in isolating more relevant attributes by putting more weight on individual attributes selected early by the decision tree rather than spreading the contribution among all attributes.

### 3.4.2 Evaluating The Model Components

In this section, we evaluate several aspects of our model such as its behavior towards accuracy-explainability tradeoff, ablating its memory mechanism and scalability.

**Accuracy and Explainability Trade-Off** The trade-off between the classification loss and the attribute loss in our aRDTC model can be measured by varying $\lambda \in [0.01, 0.99]$. Our results on AWA2 and CUB in Figure 3.5 show a slight decrease in the overall classification accuracy (red curve), when $\lambda$ approaches to 1.0 which gives more weight to the attribute prediction as opposed to class label prediction. Indeed, RDTC achieves a higher accuracy than aRDTC that is trained with the attribute loss indicating a tradeoff between explainability and accuracy. Increasing $\lambda$ leads to a slight decrease in classification accuracy, and generally similar to that of fully optimizing class prediction when $\lambda = 0$.

Furthermore, we measure the effect of $\lambda$ to the attribute prediction accuracy of the decision tree as compared to their ground-truth (purple curve). We observe a high attribute prediction accuracy even with a small $\lambda$, e.g. $\lambda = 0.2$. As we increase $\lambda$ in the range of 0.2 to 1.0, there is only a slight increase in attribute prediction accuracy, indicating that our aRDTC is robust against the choice of $\lambda$ across datasets as long as it is chosen to be at least 0.2.

**Ablating the Memory Mechanism** The LSTM state $h$ and explicit memory $\mathcal{M}$ in RDT contains

| Model | **AWA2** (# att) | **CUB** (# att) | **ImageNet** (# att) |
|---|---|---|---|
| RDTC-L | 97.7 (19) | 73.0 (50) | 60.8 (159) |
| RDTC-M | 97.9 (57) | 77.2 (93) | 71.6 (82) |
| RDTC | **98.0** (30) | **78.1** (42) | **72.8** (46) |
| aRDTC-L | 97.9 (29) | 69.1 (32) | N/A |
| aRDTC-M | 98.0 (37) | 76.4 (52) | N/A |
| aRDTC | **98.1** (34) | **77.9** (38) | N/A |

Table 3.2: Ablating the memory mechanism of aRDTC ($\lambda = 0.2$) and RDTC ($\lambda = 0$). Tree state is encoded as either only the LSTM (L), only the explicit memory (M) or both (+ median number of distinct attributes the model learns).

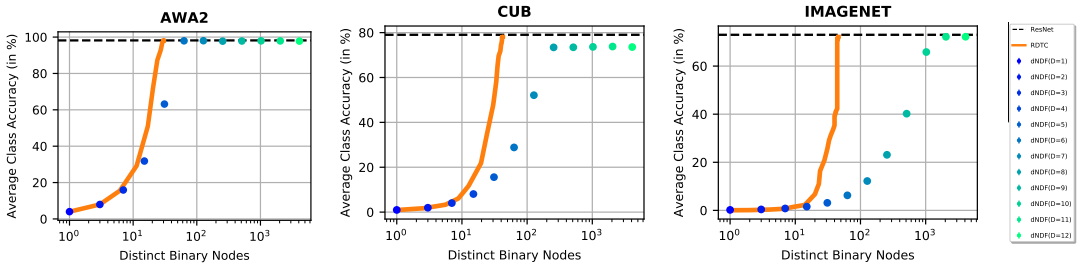



Figure 3.6: Accuracy with increasing number of nodes in RDTC and dNDF on AWA2, CUB and ImageNet. As RDTC can reuse learned nodes in the tree and has adaptive tree depth, we train it once and evaluate it at different depths. dNDF needs to be retrained for every depth hyperparameter (D) and the number of nodes scales exponentially with tree depth.

previously observed decision nodes and the current decision. While the LSTM state allows to encode the attribute order, the explicit memory serves as a more direct representation of all gathered information about the image. We ablate our RDT model with respect to its tree encoding-types.

Table 3.2 shows the classification accuracy of the following configurations: aRDTC-L, i.e. with only LSTM and attribute loss, and aRDTC-M, i.e. with only explicit memory, (vs RDTC-L and RDTC-M without the attribute loss). We observe that aRDTC-M consistently performs better than aRDTC-L, e.g. on CUB (76.4% vs. 69.1%). Moreover, combining the two in our full model generally improves the performance (up to 1.5% on CUB). These results indicate that the explicit memory is important for accuracy.

The median number of distinct attributes in paranthesis shows that aRDTC-L retains fewer decision nodes than aRDTC-M. For instance, aRDTC learns to only use 38 out of all 312 attributes of CUB ($\approx$ 12%) and on ImageNet RDTC uses only 46 learned binary attributes as opposed to the 1000 continuous features commonly used in ResNet. This increases sparsity of our model in the attribute space and improves interpretability by using fewer nodes when using the LSTM. We conclude that combining the two memory types in our RDTC model provides the best of both worlds, a high classification accuracy in few binary decisions such that the explanations of our model are concise and accurate.

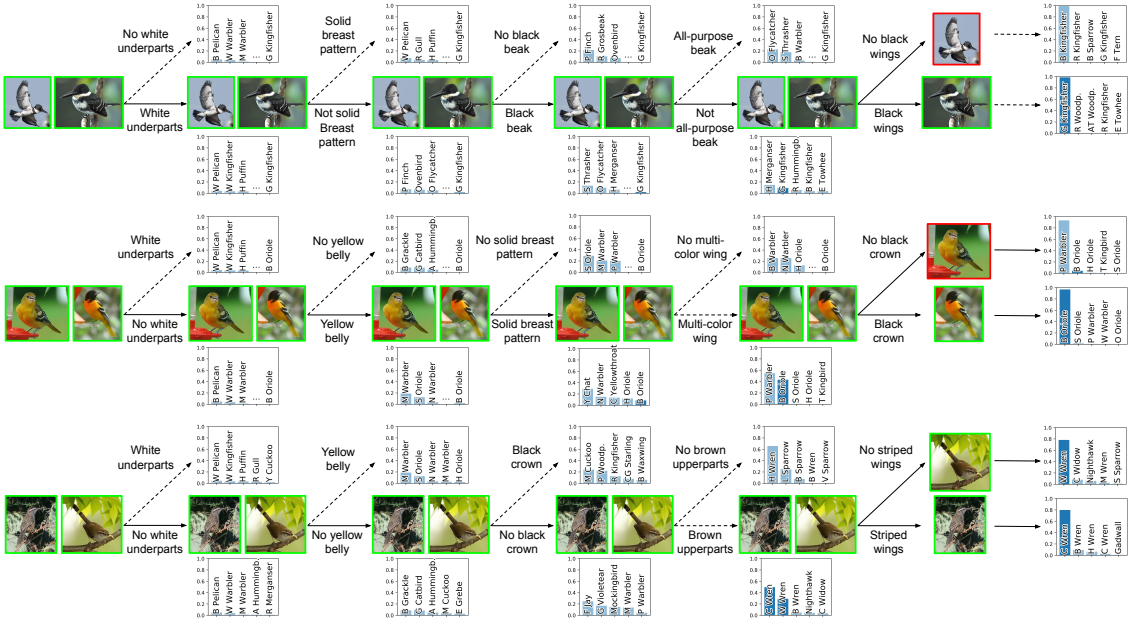



Figure 3.7: Top: Two "Green Kingfisher" images follow the same path except for "black wings", i.e. the flying bird gets misclassified as a "belted kingfisher" as black wings are not visible. Middle: Baltimore Oriole image (left) gets incorrectly classified as Prothonotary Warbler because of the missing "black crown" in the female bird. Such discrepancies, e.g. per-class attributes not reflecting the image content, make CUB difficult. Bottom: Cactus Wren (left) and Bewick Wren (right) share many characteristics except from "striped wings" which our model uses to split these classes.

**Scalability of the Learned Decision Trees.** For our RDTC, increasing the tree depth simply translates to increasing the number of binary decisions, i.e., time steps of the model-to-model communication. Hence, RDTC scales linearly with the depth of the tree while the number of weights stays constant. On the other hand, DT and dNDF grow exponentially in their number of parameters with the depth of the tree. When the same attribute is needed at different locations in the tree, our model learns the meaning of this attribute once and reuses it, while DT and dNDF would have to relearn the split. Finally, RDTC does not require finetuning a depth parameter. Hence, we have the flexibility of changing the tree depth at test time without retraining.

We compare the classification accuracy of RDTC and dNDF with an increasing number of distinct tree nodes on three datasets. As shown in Figure 3.6, RDTC (orange line) is trained only once and evaluated at different tree depths at test time while we have to retrain dNDF for each depth parameter. While the number of nodes of dNDF scales exponentially with depth (note the log-scale on the x-axis), our model adaptively learns the number of binary attributes needed to solve these classification tasks. Hence, it stops using more attributes when no further distinct nodes are necessary. We observe that RDTC uses up to an order of magnitude fewer tree nodes on AWA2, CUB and ImageNet to achieve the same or better performance. At the same time, RDTC only needs to be trained once and can be adaptively reduced in tree depth at test time.

### 3.4.3 Qualitative Results

Zooming into the decision process of misclassifications on CUB, we investigate how our model treats counterfactual classes which is useful as explanations are often contrastive [Hen+18]. We provide further qualitative examples revealing the decision tree of our aRDTC model on the fine-grained CUB and the decision tree of our RDTC model on ImageNet without the attributes in the supplementary.

In Figure 3.7 (Top), we inspect the point in the tree where the error occurred. The lower path corresponds to the most probable path taken for birds of class Green Kingfisher. Both images follow the same path for four decisions, the error occurs in the fifth decision. For the flying bird, our model decides that it "does not have black wings" and incorrectly classifies it as a Belted Kingfisher, a closely related class to Green Kingfisher, but without black wings. In addition, our model depicts its current belief of the correct class at any time during the process, i.e., probability plots at every branch which reveals some critical binary decisions, when the predicted class changes drastically, such as the "black wings" decision. This way, a user inspecting our explainable decision tree can make a more informed decision on the value of the prediction of the model.

In Figure 3.7 (Middle), the Baltimore Oriole image on the left gets incorrectly classified as Prothonotary Warbler because of the missing male-specific "black crown" attribute in the female bird. Such discrepancies, e.g. per-class attributes not reflecting the image content, make CUB an extremely challenging dataset. In Figure 3.7 (Bottom), the Cactus Wren image on the left and Bewick Wren image on the right share many characteristics except from "striped wings". The decision path is common until then where our model uses this attribute to split these classes.

## 3.5 Conclusion

In this work, we propose to learn a decision tree recurrently through communication between two-agents. Our `RDTC` framework adaptively changes tree depth at test time, allows to reuse of the learned decision nodes and improves scalability. It also uses human understandable attributes and hard binary splits for easier interpretation. Our experiments show that combining an explicit memory and an LSTM is important to obtain good performances with few inquiries. Our model maintains the accuracy of non-explainable deep models and outperforming the state-of-the-art deep decision tree learners. Qualitatively inspecting individual examples demonstrates the reasoning behind the failure and other challenging fine-grained cases, while a user study shows that `RDTC` selects more visually relevant attributes than a comparable linear semantic bottleneck model.

# 4

# Modeling Conceptual Understanding in Image Reference Games

An agent who interacts with a wide population of other agents needs to be aware that there may be variations in their understanding of the world. Furthermore, the machinery which they use to perceive may be inherently different, as is the case between humans and machines. In this work, we present both an image reference game between a speaker and a population of listeners where reasoning about the concepts other agents can comprehend is necessary and a model formulation with this capability. We focus on reasoning about the conceptual understanding of others, as well as adapting to novel gameplay partners and dealing with differences in perceptual machinery. Our experiments on three benchmark image/attribute datasets suggest that our learner indeed encodes information directly pertaining to the understanding of other agents, and that leveraging this information is crucial for maximizing gameplay performance.

## 4.1 Introduction

For a machine learning system to gain user trust, either its reasoning should to be transparent [Fre14; LBL16; Let+15; Rud19], or it should be capable of justifying its decisions in human-interpretable ways [Gil+18; Hen+16; Huk+18; WM19]. If a system is to interact with and justify its decisions to a large population of users, it needs to be cognizant of the variance users may have in their conceptual understanding over task-related concepts, i.e., an explanation could make sense to some users and not to others. Although there has been work studying what affects users' ability to understand the decisions of machine learning models [Cha+17], to the best of our knowledge existing work in explainable AI (XAI) does not explicitly reason about user understanding when generating explanations for model decisions.

As an additional complication, variations in understanding can only be inferred from observed behavior, as the system typically has no access to the internal state of its users. Further, usually not only the understanding among the population of users vary, but also how the system and its users perceive information about the world significantly differs, as is the case between human eyes and

digital cameras artificial agents use for perception.

In this work, we focus on the ability of a machine learning system, i.e. an agent, to form a mental model of the task-related, conceptual understanding other communication partners have over their environment. Particularly, we are interested in an agent that can form an internal, human-interpretable representation of other agents that encodes information about how well they would understand different descriptions presented to them. Further, we would like our agent to be capable of forming this representation quickly for novel agents that it encounters. Similar to [Rab+18], we wish to generate a representation of other agents solely from observed behavior. Rather than implicitly encoding information about the agents' policies, we explicitly encourage our learned representation to encode information about their understanding of task-related concepts. We accomplish this through a value function over concepts conditioned on observed agent behavior, yielding a human-interpretable representation of other agents' understanding.

As a testbed, we formulate an image reference game played in sequences between pairs of agents. Here, agents are sampled from a population which has variations in how well they understand different visual attributes, necessitating a mental model over other agents' understanding of those visual attributes in order to improve the overall game performance. For example, an agent might understand color attributes poorly, leading it to have trouble differentiating between images when they are described in terms of color. We present ablation experiments evaluating the effectiveness of learned representations, and build simple models for the task showing that actively probing agents' understanding leads to faster adaptation to novel agents. Further, we find that such a model can form clusters of agents that have similar conceptual understanding.

With this work, we hope to motivate further inquiry into models of conceptual understanding. Our exemplar task, i.e. image reference game, based on real-world image data allows us to explore and observe the utility of agents who are able to adapt to others' understanding of the world.

## 4.2 Related Work

**Modeling Other Agents.** Inspired by [Rab+18], we would like to model another agent solely from observed behavior, focusing on forming representations which encode information about their understanding of task-related concepts.

Recent works have also employed a similar idea to other multi-agent settings. In [ST19], an agent learns the abilities and preferences of other agents for completing a set of tasks, however, in their work they assume that the identities of the agents the learner interacts with are given and that their representation is learned over a large number of interactions. In contrast, we are interested in a learner that can quickly adapt to agents without having prior knowledge of who they are. The model presented by [Shu+18] learns how to query the behavior of another agent in order to understand its policy. However, in their work only the environmental conditions vary, with the agent being modeled remaining the same. Here, we vary both agent and environment. There also exists a body of work on computational models of theory of mind [But+09; War+12], particularly employing Bayesian methods [BST11; Bak+17; NBT16], although they use discrete state spaces rather than continuous ones.

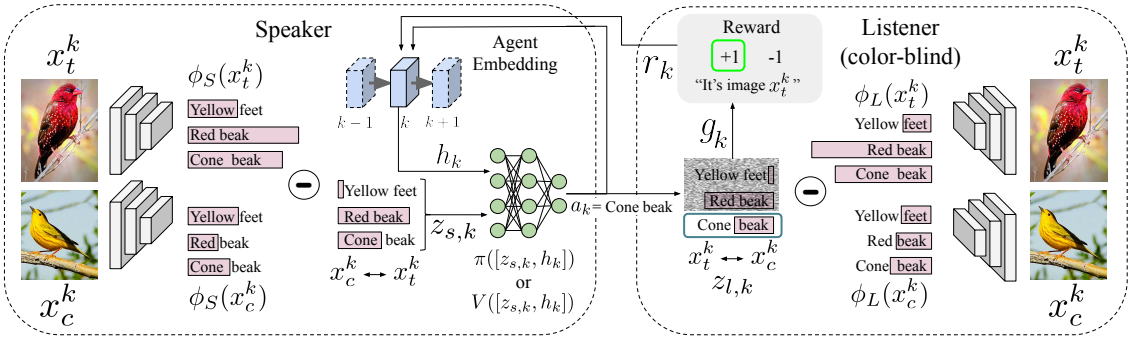



Figure 4.1: Our image reference game with varied agent population. In a given episode $k$, the speaker and listener encode the image pair $(x_t^k, x_c^k)$ using their perceptual modules $\phi_S, \phi_L$. The speaker selects a target image $x_t^k$ and an attribute $a_k$ to describe it using parameterized functions $\pi_S$ and $V$ conditioned on the image representations and agent embedding $h_{k-1}$. Given $a_k$, the listener guesses the target image. Finally, the speaker incorporates information about the listener into embedding $h_k$ given the reward $r_k$ received for using $a_k$ in that game.

**Meta Learning.** In meta-learning [Ben+92; FAL17; Sch87], an agent is tasked with learning how to solve a family of tasks such that it can quickly adapt to new ones. In our work, we are interested in an agent that can learn to quickly adapt to the conceptual understanding of novel gameplay partners whose understanding is correlated to other agents from the population (e.g. such as learning to identify when someone is color-blind).

**Emergent Language.** There have been a number of works presenting multi-agent systems where agents must collaboratively converge on a communication protocol for specifying goals to each other [CLF18; Das+17; Evt+18; Foe+16; HT17; JKG16; Kot+17; Laz+18; LPB17]. Whereas in these works the main focus is to learn an effective communication protocol and to analyze its properties, here we are interested in modeling other agents' understanding of the environment. We therefore assume a communication protocol is given so that we test agent modeling in isolation. Further, many of these works assume that gradients are passed between agents. Here, we assume a discrete bottleneck in that agents only have access to observations of each other's behavior. Although some domains have a population of agents [Cog+19; MA18], the tasks do not use real images and all agents either share a single policy or have equal capacity to understand task-related concepts. We believe that incorporating an emergent communication component to our domain would be an exciting avenue for future work.

## 4.3 Image Reference Game with Varied Agent Population

In a multi-agent communication setting, it is generally best to send a message which maximizes the amount of task-related information, such as describing an image by appealing to its most discriminative attributes. However, the recipient of the message may not be familiar enough with certain attributes, meaning that some messages are not useful to them despite being maximally informative. In line with this motivation, we formulate an image reference game where agents must describe images to each other using visual attributes.

**Task definition.** In our visual reference game (see Figure 4.1) we have a single learner, referred to as the *speaker*, who must learn to play sequences of episodes in an image reference game with gameplay partners, referred to as *listeners*, that are randomly sampled from a population of agents. Both the speaker and the listeners are given a pair of images in each episode $k$, and the speaker selects an image $x_t^k$ to serve as the target, with the second image $x_c^k$ serving as confounder. The speaker must then generate a description, in the form of an image attribute $a_k \in A$, which the listeners use to compare the two images before guessing the target's identity.

As listeners are effectively black-boxes to the speaker, it can be difficult to disentangle potential sources of error when they behave unexpectedly. Namely, when a listener guesses incorrectly, it is difficult to tell whether the mistake was due to a lack in its understanding of 1) the game, 2) the language used to communicate, or 3) the attribute (i.e. concept) used to describe the image. In this work, we focus on the third option (conceptual understanding), and isolate this problem from the other two by assuming that the speaker can communicate attribute identities noiselessly, and that listeners are all rational game players sharing a static gameplay policy.

**Perceptual Module.** An agent's perceptual module $\phi$ encodes images into a list of shared concepts weighted by their relevance to the image. Specifically, the perceptual module first extracts image features using a CNN. The image features are further processed with a function $f$ that predicts attribute-level features $\phi(x) = f(\text{CNN}(x))$, where $\phi(x) \in [0,1]^{|A|}$, and $|A|$ is the number of visual attribute labels in an attribute-based image classification dataset.

Every element in $\phi(x)$ represents a separate attribute, such as "black wing", giving us a disentangled representation. The speaker and listener policies reason about images in the attribute space $A$; we are interested in disentangled representations because they will allow for the speaker's mental model of listeners' understanding to be human interpretable. In our setting, the speaker is given a separate module $\phi_S$, while all listeners share a single module $\phi_L$.

### 4.3.1 Modeling Listener Populations

If a listener has a good understanding of an attribute, we would expect that it would be able to accurately identify fine-grained differences in that attribute between a pair of images. For example, someone with a poor understanding of the attribute "red" may not be able to distinguish between the red in a tomato and the red in a cherry, although they might be capable of distinguishing between the redness of a fire truck and that of water. Following this intuition, we generate a population of listeners $L = \{(\delta_l, p_l)\}$, where each listener $l \in L$ is defined by a vector of thresholds $\delta_l \in [0,1]^{|A|}$ and a vector of probabilities $p_l \in [0,1]^{|A|}$.

Given an image and attribute feature pair $\big(\phi_L(x_t^k), \phi_L(x_c^k)\big)$, the listener $l$ first computes the difference between the attribute features $\phi_L$ of image $x_t$ and $x_c$ for attribute $a$:

$$z_l^a = \phi_L^a(x_t^k) - \phi_L^a(x_c^k). \tag{4.1}$$

Using its attribute-specific threshold $\delta_l^a$, if $|z_l^a| < \delta_l^a$, then the listener does not understand the concept well enough and will choose the identity of the target image uniformly at random. Conversely, if $|z_l^a| \geq \delta_l^a$, then the listener will guess rationally with probability $p_l^a$ and randomly with

probability $(1 - p_l^a)$. Here, a rational guess $g = \arg\max_{x \in \{x_t^k, x_c^k\}} \phi_L^a(x)$ means choosing the image which maximizes the value of the attribute $a$.

To simplify the setup, we specify a total of two different levels of understanding. An agent can either understand an attribute, i.e. $u = (\delta, p)$, or not understand an attribute, i.e. $\bar{u} = (\bar{\delta}, \bar{p})$. For $u$, $\delta$ is small and $p$ is set to 1, respectively meaning that attributes are easily understood and the agent always plays rationally on understood attributes. Conversely, $\bar{u}$ specifies a high value for $\bar{\delta}$ and $\bar{p}$ is lower than 1, such that an attribute that is not understood rarely leads to rational gameplay.

To form a diverse population of listeners, we create a set of clusters $C$ where each cluster is defined by the likelihood of assigning either $u$ or $\bar{u}$ to each individual attribute. Thus, listeners sampled from the same cluster will have correlated sets of understood and misunderstood attributes, while remaining diverse.

### 4.3.2 Modeling the Speaker

In a given sequence, the speaker plays $N$ practice episodes, each consisting of a single time-step, where the purpose is to explore and learn as much about the understanding of the listener as possible, purely from observed behavior. During the $k$'th game in a sequence with a given listener, the speaker first encodes the image pair with $\phi_S$. From the previous $k - 1$ games, the speaker also has access to an agent embedding $h_{k-1}$, which encodes information about the listener. The speaker uses an attribute selection policy to select an attribute $a_k$ for describing the target image. After the listener guesses, the reward $r_k$ from the game is used to update the agent embedding into $h_k$. After the practice episodes, $M$ evaluation episodes are used to evaluate what the speaker has learned.

**Agent Embedding Module.** To form a mental model of the listener in a given sequence of episodes, the speaker makes use of an agent embedding module. This module takes the form of an LSTM [HS97] which incorporates information about the listener after every episode, with the LSTM's hidden state serving as the agent embedding. Specifically, after selecting an attribute $a_k$ and receiving a reward $r_k \in \{-1, 1\}$, a one-hot vector $o_k$ is generated, where the index of the non-zero entry is $a_k$ and its value is $r_k$. The agent embedding $h_k = \text{LSTM}(h_{k-1}, o_k)$ is then updated by providing $o_k$ to the LSTM.

**Attribute Selection Policies.** The speaker has access to two parameterized functions, $V(s_k, a_k)$ and $\pi_S(s_k, a_k)$, represented by multi-layer perceptrons. The speaker uses these functions to select attributes during the $N$ practice and $M$ evaluation episodes, where $s_k = \left[\phi(x_t^k) - \phi(x_c^k); h_k\right]$ is a feature generated by concatenating the image-pair difference and agent embedding.

We estimate the value of using each attribute to describe the target image, i.e. $V(s_k, a_k) : \mathcal{R}^d \times \mathcal{A} \to \mathcal{R}$ using episodes from both the practice and evaluation phases optimizing the following loss:

$$\mathcal{L}_V = \frac{1}{N+M} \sum_{N+M} \text{MSE}(V(s_k, a_k), r_k) \tag{4.2}$$

As $V$ approximates the value of each attribute within the context of a listener's embedding and an image pair, it directly provides a human-interpretable representation of listeners' understanding. Therefore, every model presented uses it greedily to select attributes during evaluation games.

The purpose of practice episodes is to generate as informative an agent embedding as possible for $V$ to use during evaluation episodes. Therefore, speakers differ in how they select attributes during practice episodes, probing listeners' understanding with different strategies. One strategy is to use an attribute selection policy $\pi_S$, trained with policy gradient [Sut+00], which directly maps to probabilities over attributes. In the following, we describe different attribute selection strategies used during practice episodes.

**a. Epsilon Greedy Policy.** For this selection policy, we simply either randomly sample an attribute with probability $\epsilon$ or greedily choose the attribute $a_k = \arg\max_{a \in A} V(s_k, a)$ using $V$.

**b. Active Policy.** The active policy is trained using policy gradient:

$$\mathcal{L}_a = \frac{1}{N} \sum_{N} -R \log \pi_S(s_t, a_t) \text{ with } R = -\frac{1}{M} \sum_{M} \text{MSE}(V(s_k, a_k), r_k) \tag{4.3}$$

where the reward ($R$) for the policy is a single scalar computed from the evaluation episode performance. This encourages the policy to maximize the correctness of the reward estimate function $V$ during evaluation episodes, requiring the formation of an informative agent embedding during the practice episodes. Note that when optimizing the active policy $\pi_S$, gradients are not allowed to flow through $V$.

## 4.4 Experiments

In the following, we first evaluate the effects of using different attribute selection strategies during practice episodes and then the quality of agent embeddings generated by each model. We use the AwA2 [Xia+18], SUN Attribute [Pat+14], and CUB [Wah+11] datasets. Unless stated otherwise, the listener population consists of 25 clusters, each with 100 listeners. We use two variants of the perceptual module, ResNet-152 [He+16], fine-tuned for attribute-based classification with an ALE [Aka+13] head, and PNASNet-5 [Liu+18a] with an attribute classifier head. Both ResNet and PNASNet-5 are pre-trained on ImageNet [Den+09] and fine-tuned for the attribute-based image classification task. Note that unless stated otherwise, in each experiment both the speaker and listeners use the same perceptual module, i.e. $\phi_S = \phi_L$.

For all curves we plot the average over 3 random seeds, with error curves representing one standard deviation. We use the standard splits for CUB and SUN, but make our own split for AwA2 in order to have all classes represented in both train and test. The training splits are used for learning speaker parameters; we present performance on the test splits, using the same splits for each seed. We sample target and confounder images from the same dataset split. Listener clusters $C$ are shared across train and test but a novel population of listeners is sampled at test time[1].

### 4.4.1 Policy Comparison

We first compare the performance of the Epsilon Greedy and Active policies described in Section 4.3.2 against three baselines, the Random Agent, Reactive, and Random Sampling policies.

[1]Code with full specifications for experiments may be found at: `https://github.com/rcorona/conceptual_img_ref`

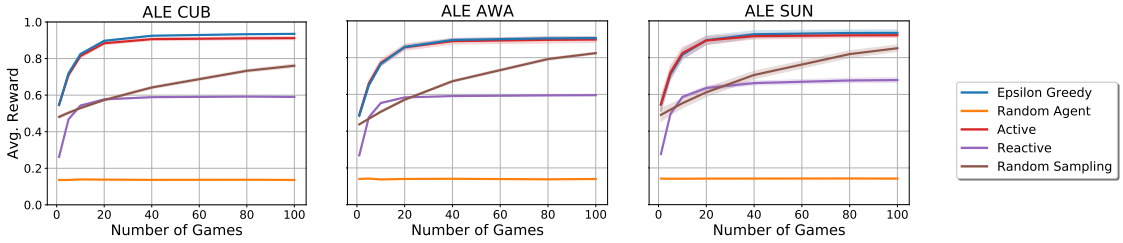



Figure 4.2: A comparison of average test set performance (Avg. Reward) for different attribute selection policies vs. the number of practice games. All agents learn from the listeners responses, i.e. using an embedding module, except for the random agent which always acts randomly. With an increasing number of games, the agent observes more responses providing information about the listener's conceptual understanding.

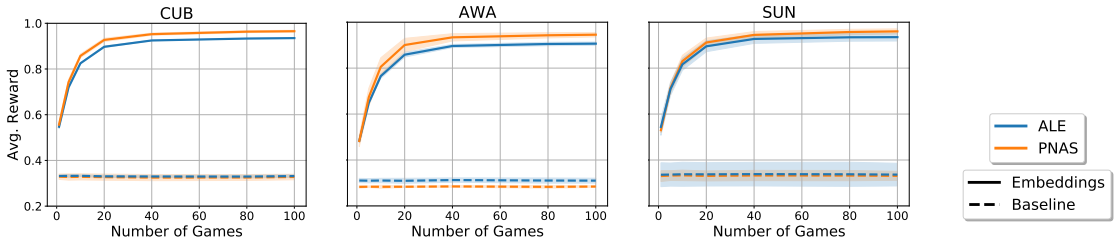



Figure 4.3: Ablation study on the importance of the agent embedding module. Average reward of the Epsilon Greedy policy on the test set as the number of practice episodes played increases. We evaluate the performance on two different perception modules (ALE, PNAS) with embedding module and without (baseline).

Among the baselines, the Random Agent policy simply always selects an attribute at random to describe images. The Reactive policy, at the beginning of each set of $N + M$ episodes, randomly selects an attribute. It continues using this attribute for each episode, only sampling a different attribute whenever it encounters a negative reward, keeping track over which attributes it has used. This policy is meant as a sanity check against a degenerate strategy of only using the LSTM to remember which attributes have worked and which have not, without incorporating useful information about the listener's conceptual understanding. Finally, the Random Sampling baseline selects random attributes during practice episodes, and then follows a greedy strategy over $V$ during evaluation episodes.

The performance of these policies on the test set is presented in Figure 4.2 which shows that the Epsilon Greedy and Active selection policies both outperform the Reactive baseline, suggesting that the agent embedding is encoding information about the conceptual understanding of listeners. After a large number of games, we would expect the performance of the Epsilon Greedy, Active and Random Sampling policy to be the same because at some point the speaker agent has learned about all the listener's understood and misunderstood attributes. By comparing against the Random Sampling policy, we can conclude that both the Epsilon Greedy and Active policies can learn more efficient strategies that identify the misunderstood attributes within the first 20 games, at least five times faster than the Random Sampling policy. This corroborates the positive effect of encouraging policies to query information that helps the speaker form a mental model of the listener.

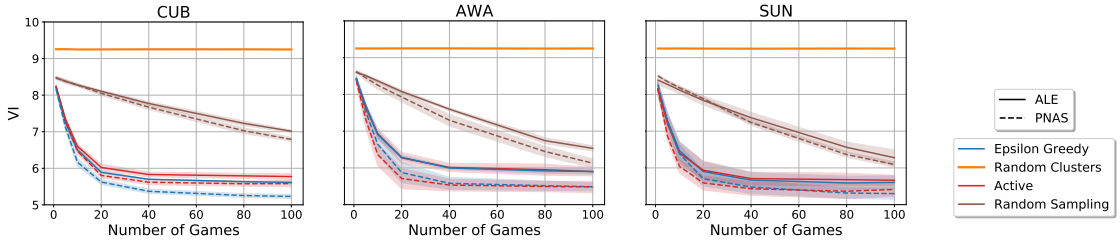



Figure 4.4: Variation of information ($VI$) of agent clusters $C'$ compared to ground-truth cluster assignments $C$. We present $VI$ for different policies as the number of practice games increases, lower is better. Cluster assignments $C'$ are obtained via K-Means ($k = |C|$) on agent embeddings from 50K test set sequences for each policy. Random Clusters (baseline) assigns each embedding to a random cluster. Each policy is evaluated using two different perception modules (ALE, PNAS).

### 4.4.2 Evaluating Agent Embedding

Here we present an ablation study to investigate the benefit of using agent embeddings when playing the game, training an epsilon-greedy policy for each dataset until convergence with (Embeddings) and without (Baseline) agent embeddings.

Models without agent embeddings are given zero vectors, $h_k = 0$, instead of agent embeddings as input for the attribute selection policies. In these experiments, the speaker and listeners share the same perception module; we test performance for both the ALE and PNAS perceptual modules. Intuitively, a speaker will improve its performance over the game sequence if it encodes useful information about the listener, since it will help it avoid using attributes which the listener does not understand well.

In Figure 4.3, we show the average reward at different intervals of the game sequence. Using an agent embedding module significantly improves the performance of the speaker over time in all cases. Most importantly, performance improves as the number of games increases, showing that a speaker using an agent embedding module can quickly adapt to individual listeners from experience to avoid using misunderstood attributes and, thus, achieve a higher average reward.

### 4.4.3 Evaluating Cluster Quality

Although we have shown that agents with an agent embedding module achieve better performance, these results do not necessarily imply that speakers with memory develop an informative mental model over the conceptual understanding of the listeners. In order to test this, we perform an additional experiment on the trained speaker models. Specifically, we play roughly 50K sequences on the test set in order to generate a dataset of agent embeddings. We then perform K-Means clustering on these embeddings with $k = |C|$ (i.e. the number of listener clusters in the population) to obtain cluster assignments $C'$ and compare them to the ground-truth listener cluster assignments $C$ .

To evaluate the cluster quality, we use the variation of information (VI) metric [Mei03]:

$$VI(C, C') = H(C) + H(C') - 2I(C, C') \tag{4.4}$$

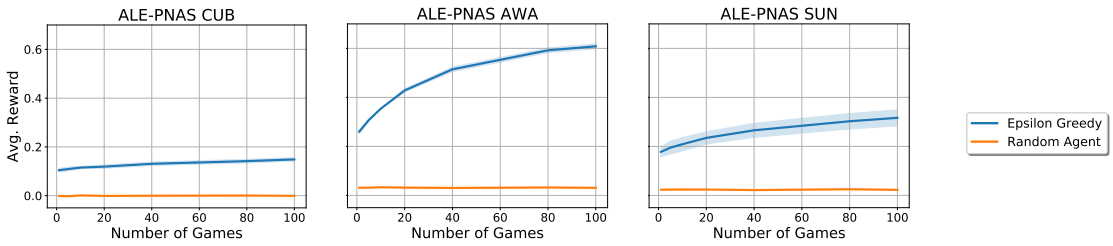



Figure 4.5: Average test set performance over different evaluation intervals for Epsilon Greedy and a random baseline. Here we test giving the speaker and listener population different perceptual modules (speaker uses ALE, listener uses PNAS).

Here, $C$ and $C'$ are two different clusters, $H$ is the entropy, and $I$ is the mutual information. Intuitively, the $VI$ measures how much information is lost or gained by switching from clustering $C$ to $C'$.

The more informative agent embeddings are about listeners' understanding, the greater the correlation will be between the inferred cluster and the ground-truth cluster. Figure 4.4 shows clustering performance for all parameterized policies as the number of practice games increases per sequence. We additionally compare this performance to a random cluster assignment baseline.

Firstly, we note that every policy outperforms the random assignment baseline. The Epsilon Greedy and Active policies experience nearly identical gameplay performance, suggesting that simply optimizing for reward yields similarly informative embeddings as more explicitly encouraging the policy to maximize the value function's accuracy. Finally, the Random Sampling baseline converges much more slowly, corroborating the idea that a more directed exploration of listeners' capabilities proves useful. Due to the significant improvement over random cluster assignments, we conclude that the speaker agent learns an embedding that clusters the listeners similar to the ground truth. This suggest that the agent not only learns from previous games, but it also forms a more general representation of listener groups with similar conceptual understandings.

### 4.4.4 Evaluating Different Perceptual Modules

If our speaker is to interact with a varied population of agents, it not only needs to be cognizant that those it interacts with could have varying levels of understanding; the population itself could have inherently different machinery for perceiving the world, as is the case between humans and machines.

Therefore, we repeat the experiment from section 4.4.1 with the Epsilon Greedy policy, and give the speaker and the listener population different perceptual modules. Specifically, in Figure 4.5, we show test performance when assigning ALE to the speaker and PNAS to the listeners comparing to a speaker which randomly selects attributes.

We observe a drastic change in performance, which suggests that the difficulty of the problem significantly increases when the speaker and listeners have fundamentally different perception. Notice, however, that the performance of the Epsilon Greedy policy still significantly outperforms the random baseline. Further, particularly in the case of the Animals with Attribute dataset, the

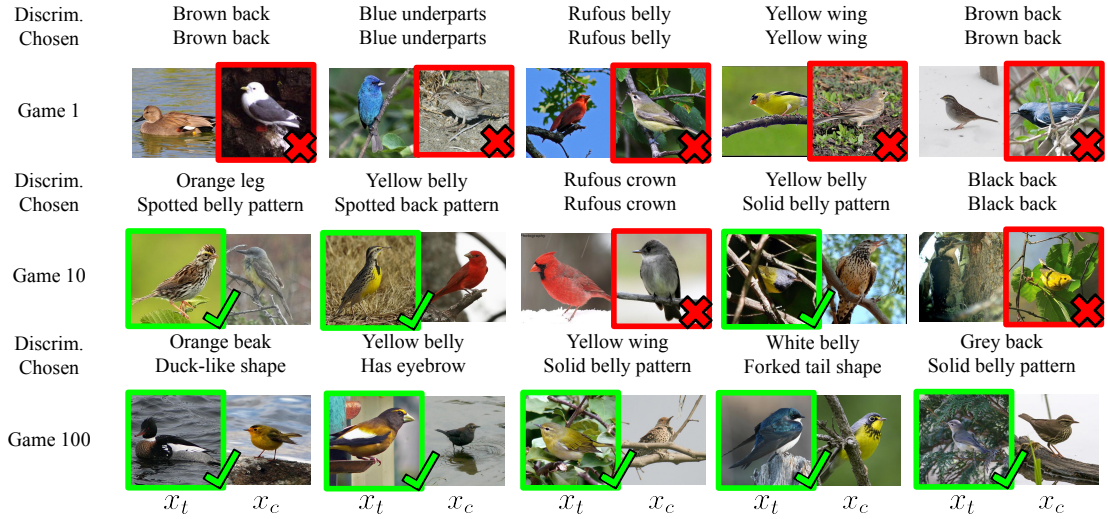



Figure 4.6: Qualitative examples of the Epsilon Greedy agent on CUB interacting with a color-blind listener. Red (green) indicates an incorrect (correct) pick by the listener. We show examples where the speaker loses the first game due to selecting a discriminative color attribute. Even though color attributes are objectively more discriminative, in game 10 the speaker communicates color attributes less frequently. Finally, at convergence, i.e. game 100, the speaker prominently mentions shape-based attributes or non-color patterns.

Epsilon Greedy speaker is still able to improve its performance as the number of episodes increases. This motivates further work in models that are capable not only of reasoning about conceptual understanding but also of adapting to fundamental differences in perception.

### 4.4.5 Qualitative Example

To provide an illustrative example of our reference game and the behavior of the agents, we train an Epsilon Greedy policy on the CUB dataset with 5 listener clusters, pertaining to the 5 attribute types found in the dataset (i.e. color, shape, size, pattern, and length). Each cluster in the listener population has a generally poor understanding of the attribute type it is assigned (e.g. the color cluster is color-blind). We visualize the center crop of the images as presented to both the speaker and the listener populations.

In Figure 4.6, we show sequences of games with color-blind listeners, where we can observe how the speaker adapts its strategy as it learns more about its gameplay partner – specifically, it adapts to using non-color attributes even in cases where color attributes would generally be most discriminative. In the first game, the speaker refers to objectively very discriminative color attributes such as brown back and rufous belly (columns 1 and 3). By game 10, the speaker already chooses color-invariant patterns over color attributes for some of the color-blind listeners, e.g. pointing out a spotted belly pattern over orange legs (column 1). After 100 games, we observe that the speaker almost always refers to non-color attributes, such as the duck-like shape or the presence of an eyebrow (column 1 and 2) because it leads to a higher average reward for color-blind listeners.

## 4.5 Conclusion

In this work, we presented a task in which modeling the understanding that other agents have over concepts is necessary in order to succeed. Further, we provide a formulation for an agent that is capable of modeling other agents' understanding and can represent it in a human-interpretable form. We believe that the ability to perform this kind of reasoning will allow XAI systems to tailor their explanations to the specific users with whom they interact. Learned agent embeddings can allow us to recover a clustering over other agents' conceptual understanding, which is a promising result to further tie this information into explanations. For example, by having explanations that are fitted to each cluster, generated explanations would be more easily digestible by users of the system. Further, we show that naively modeling this type of reasoning is not sufficient for cases where the perceptual machinery of the learner and the population is fundamentally different.

# 5

# ABSTRACTING SKETCHES THROUGH SIMPLE PRIMITIVES

Humans show high-level of abstraction capabilities in games that require quickly communicating object information. They decompose the message content into multiple parts and communicate them in an interpretable protocol. Toward equipping machines with such capabilities, we propose the Primitive-based Sketch Abstraction task where the goal is to represent sketches using a fixed set of drawing primitives under the influence of a budget. To solve this task, our Primitive-Matching Network (PMN), learns interpretable abstractions of a sketch in a self supervised manner. Specifically, PMN maps each stroke of a sketch to its most similar primitive in a given set, predicting an affine transformation that aligns the selected primitive to the target stroke. We learn this stroke-to-primitive mapping end-to-end with a distance-transform loss that is minimal when the original sketch is precisely reconstructed with the predicted primitives. Our PMN abstraction empirically achieves the highest performance on sketch recognition and sketch-based image retrieval given a communication budget, while at the same time being highly interpretable. This opens up new possibilities for sketch analysis, such as comparing sketches by extracting the most relevant primitives that define an object category. Code is available at `https://github.com/ExplainableML/sketch-primitives`.

## 5.1 Introduction

Consider the game *Pictionary*[1], where one player picks an object, e.g. a face, and draws the object in an iterative manner, e.g. using a large circle for the head, small lines for eyes and an arc for the mouth, until the other players guess the object correctly. The goal is to represent an object by decomposing it into parts that characterize this object using as few parts as possible such that another player can recognize it as fast as possible. The inherent human ability [FS88] that makes playing this game with multiple players possible is the ability to identify the most distinctive parts of the object and ground them into an interpretable communication protocol for the other players. In other words, humans are capable of a high level of abstraction when thinking about, recognizing

[1] https://en.wikipedia.org/wiki/Pictionary

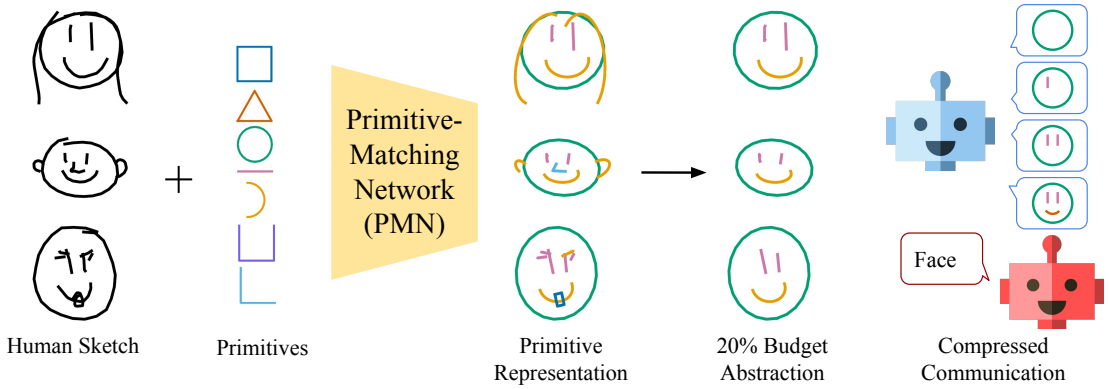



Figure 5.1: **Primitive-based Sketch Abstraction Task**. Our Primitive-Matching Network (PMN) takes human sketches and replaces their strokes with simple shapes from a set of 7 drawing primitives to create an abstract representation of the sketch. We can further compress the sketch by sub-selecting primitive strokes to meet a desired information budget. When communicating a sketch with a limited budget, our sketch abstractions retain the original semantics to perform well on downstream tasks.

and describing objects to other.

Inspired by this observation, we propose Primitive-based Sketch Abstraction as a new representation learning task, where the goal is to represent free-form drawings, i.e. sketches, by means of a fixed set of simple primitives. Sketches are an excellent tool for this task as they capture the essential parts of an object while removing the potentially adversarial texture and color information. However, humans have different drawing styles and skills, influenced by their upbringing and culture [SCH66]. This causes different participants to draw the same instance of a real object in different ways (e.g. see Fig. 5.1, left). We argue, however, that there exists a fundamental representation of each object class. As demonstrated in [BL99; FS88; RIA93] when a participant draws an object of their imagination using a fixed dictionary of shapes providing a heavily abstracted representation of the object, another participant still guesses the object correctly.

To solve the Primitive-based Sketch Abstraction task, we propose a self-supervised deep model, i.e. Primitive-Matching Network (PMN), to learn interpretable abstractions of a given object illustration without requiring any ground-truth abstraction. Differently from standard sketch-abstraction [Muh+19; Muh+18], which selects subsets of the original strokes, our model grounds them to a predefined vocabulary of primitives with a budget, see Fig. 5.1. This way of representing sketches has two main advantages. First, it reduces the memory footprint of the sketch representation, allowing to communicate sketches by their constituent primitives rather than stroke coordinates. Second, it increases the interpretability of the sketch itself, making it much easier to compare and contrast sketches, e.g. a *human face* is composed of a big circle for the head, two small lines for the eyes and one arc for the mouth whereas a *cat face* is similar to a *human face* but has triangles on top of the head for its ears.

Our PMN model replaces each stroke of a sketch with a single drawing primitive. This is achieved by mapping each stroke to its most similar primitive in a given set, and predicting an affine transformation that aligns the selected primitive to the target stroke. We train PMN by

comparing the distance-transform of target strokes and their primitive-based version. At test time, given a sketch, we can efficiently choose a set of primitives and their spatial transformations, such that the generated sketch is fully composed of primitive shapes while being as similar as possible to the original one. Experiments on sketch recognition and fine-grained sketch-based image retrieval tasks, show that the PMN abstraction achieves the highest performance given a communication budget (i.e. number of bytes necessary to communicate the sketch). Moreover, we show how we can use our abstraction to compare sketches, extracting the most relevant primitives and patterns that define an object category.

To summarize, our contributions are: i) we propose the task of Primitive-based Sketch Abstraction, where the goal is to produce interpretable sketch representations by means of predefined drawing primitives; ii) we propose the first method for this task, Primitive-Matching Network, which learns to match strokes to primitives using as supervision a reconstruction loss over their distance transforms; iii) we show that PMN provides reliable sketch representations, communicating more information with a lower budget when compared with standard sketch abstraction methods, and eases sketch analysis.

## 5.2 Related works

**Sketch Abstraction.** The goal of sketch abstraction [Ber+13; Muh+18] is to simplify the original strokes (or segments) from sketches without altering their semantic meaning. Abstracting sketches allows to communicate their information more effectively and efficiently, highlighting the most important traits of a sketch without corrupting its content [Ber+13]. This is used in many applications, ranging from sketch-based image retrieval from edge-maps, to controllable sketch synthesis at various abstraction levels. Previous approaches addressed this problem through reinforcement learning, learning to remove sketch parts while preserving some desired features (e.g. semantic category, attributes) [Muh+19; Muh+18]. Differently from previous works, we do not abstract sketches by removing strokes, but we ground them to a set of drawing primitives. This allows us to not only simplify the sketch representation itself, but to easily perform comparisons and analyses across sketches in a more straight forward manner than with stroke-based abstraction methods.

**Sketch Applications.** The release of the TU-Berlin [Eit+10] and QuickDraw [Jon+16] datasets attracted the attention of the research community towards sketch classification. Early works addressed the task with maximum margin classifiers over hand-crafted features [LSG13; ST14]. Advent of large-scale sketch datasets led to the development of deep learning models for this task that even surpassed human performance [Yu+17b]. Recent approaches explored deep and hand-crafted features [Jia+20], multi-graph transformers [XJB21], coarse-to-fine hierarchical features [Yan+21], and learned tokenization schemes [Rib+20].

Another popular application of sketches is *sketch-based image retrieval* (SBIR), where the goal is to match free-hand sketches with corresponding natural images, both at category [PM14; TC17] and at instance level [Bhu+21; Pan+20]. Existing approaches for this task bridge the domain gap between photos and sketches by means of two branch architectures focusing on each modality

independently [Cao+11; Cao+10; Fed+20], and even applying attention-based objectives [Pan+19; Son+17] or self-supervised ones [Pan+20]. Recently, [Bhu+20] proposed to perform retrieval online, while the human is drawing. [Saa17; SBO15] perform SBIR by matching keyshapes to patches of sketch and contour images for SBIR, e.g. through S-HELO [Saa14] descriptors. In this work, we do not directly address sketch recognition and SBIR, but we use them to quantitatively analyze the compression/quality of our abstract sketch representations.

**Reconstruction with primitives.** One way of simplifying a complicated shape is to build an approximation using simple primitives. This is a central aspect of how humans understand the environment [Bie87] and has been applied to vector-like bitmap images [HP11; Red+21; Wu+15], CAD sketches [Gan+21; Par+21; Sef+21], and 3D shape reconstruction from sketches [SBS21] or images [Liu+18b; Zen+20]. Interestingly, also many lossy image compression methods represent an image as a combination of predefined primitives [TM12; Wal92]. One closely related work [Wu+15] focuses on diagram-like sketches, using shape proposals and an SVM classifier to assign the best-matching primitive. [Xia+15] represents sketches and edge maps of real images through lines and arcs for sketch-based image retrieval. Differently from these approaches, we are not restricted to specific domains [Wu+15], or primitives [Xia+15]. PMN is generic and can be applied to any sketch, and any set of drawing primitives.

## 5.3 Abstracting Sketches by Drawing Primitives

Given a sketch, our goal is to obtain an abstract representation by replacing its strokes with a set of drawing primitives (e.g. squares, circles, lines). Formally, we have a training set $\mathcal{T} = \{\mathbf{s}^k\}_{k=1}^K$ of sketches, where $\mathbf{s}^k \in \mathcal{S}$ is a sketch in the set of possible drawings $\mathcal{S}$. Following previous works [Muh+18], we assume that each sketch $\mathbf{s}^k$ (henceforth $\mathbf{s}$ for readability) is composed of a set of strokes (i.e. $\mathbf{s} = \{s_1, \ldots, s_n\}$), and that each stroke is defined as a sequence of two-dimensional points of length $m(s_i)$. Additionally, we assume to have a set $\mathcal{P} \subset \mathcal{S}$ of drawing primitives that we want to use to represent our sketches, where each primitive is also a sequence of points. Note that no constraint is imposed on the primitives composing $\mathcal{P}$. At test time, given a sketch $\mathbf{s}$, our goal is to re-draw each stroke $s \in \mathbf{s}$ with a primitive $p \in \mathcal{P}$. This requires two steps: first, we need to map each stroke $s_i$ to its closest primitive $p_i \in \mathcal{P}$. Second, we need to compute the affine transform parameters making the primitive $p_i$ better fit the original stroke $s_i$. In the following, we describe how we achieve these goals.

### 5.3.1 Learning to match strokes and primitives

There are two main challenges in matching an arbitrary stroke $s$ with a primitive $p \in \mathcal{P}$. First, we have no ground-truth pairs available, thus we have no direct information on which primitive $p$ is the most similar to the stroke $s$. Second, even if we find the best primitive, we need still to align it to the stroke. As a simple example, if we take two straight lines of equal length, a perfect match in case they are parallel, they result in a bad match if they are orthogonal to each other. We overcome

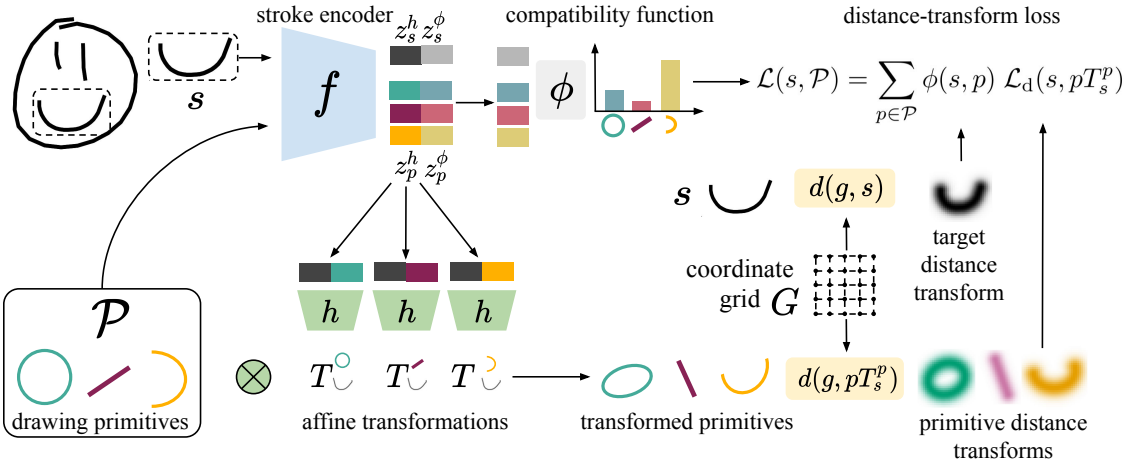



Figure 5.2: **PMN Model Architecture**. Given an input stroke (top left) and a set of primitives (bottom left), PMN encodes them into a shared embedding space using $f$. The embeddings are split in two parts, one for $h$ to compute the affine transformations aligning primitives to the target stroke, and one to compute the compatibility between primitives and the strokes with $\phi$. From a coordinate grid $G$, we compute a distance transform function of the stroke and the transformed primitives. We then use distance transforms and the compatibility scores to build the self-supervised objective of PMN.

these issues by i) applying an affine transformation to the primitives in $\mathcal{P}$ and ii) comparing the original strokes and the transformed primitives through their *distance transform*.

**Aligning strokes and primitives.** We need to transform a primitive in such a way that it better matches a given target stroke. To this end, we instantiate two functions, a stroke encoder $f : \mathcal{S} \to \mathbb{R}^d$, mapping a stroke (or primitive) to a d-dimensional embedding, and an alignment function $h : \mathbb{R}^d \times \mathbb{R}^d \to \mathrm{Aff}(\mathbb{R}^2)$, predicting the affine transformation that best aligns two strokes given their encoded representations. With $h$, we compute a transformation matrix $T_s^p$ as:

$$T_s^p = h(z_p, z_s) \tag{5.1}$$

where $z_y = f(y)$ is the feature vector of the encoded sketch/primitive $y$, and $T_s^p$ the transformation aligning the primitive $p$ to the stroke $s$.

**Distance transform loss.** Our goal is to find replacements for human strokes from a set of primitive shapes such that the visual difference is minimal. Given a stroke $s$, which is represented as a sequence of $m(s)$ connected points, i.e. $s = \{x_1, \ldots, x_m\}$ and given a coordinate $g \in G$, with $G$ being a sampled coordinate grid, we can define the influence of the stroke at $g$ as:

$$d(g,s) = \max_{i\in\{1,\ldots,m(s)-1\},\ r\in[0,1]} \exp\big(-\gamma\ ||g - r\,x_i - (1-r)x_{i+1}||^2\big). \tag{5.2}$$

Computing $d(g,s)$ for every coordinate in $G$ we obtain a distance map, also called *distance transform* [RP68]. Note that in Eq. equation 5.2 we do not use directly the distance transform but its exponentially inverted version. This allows us to highlight the map on points closer to the

stroke, with $\gamma$ acting as a smoothing factor. We can interpret this map as a visual rendering of the particular stroke, where the intensity of each pixel (coordinate) $g$ decreases with the distance of $g$ to the stroke itself. Considering a stroke $s$ and a primitive $p$, we can then define the distance transform loss as:

$$\mathcal{L}_{\text{d}}(s, p|h) = \sum_{g \in G} ||d(g, s) - d(g, pT_s^p)||. \tag{5.3}$$

With Eq. equation 5.3, we are defining a reconstruction loss that sidesteps possible mismatches in the number of points contained in $s$ and $p$ as well as the need of matching points across the two strokes. For simplicity, we normalize the coordinates of each point in $s$ and $p$ to the range $[-1, 1]$ before applying the loss and we consider $G$ as a set of linearly spaced coordinates in $[-1.5, 1.5]$.

**Exploiting stroke similarities.** Up to now we have discussed how we can align one primitive to a target stroke by means of the affine transformation computed by $h$ and how we can train $h$ by comparing distance transforms. However, during inference we want to replace $s$ with the best matching primitive selected from the set $\mathcal{P}$. With the current formulation, this could be done by replacing $s$ with the primitive $p \in \mathcal{P}$ for which the loss $\mathcal{L}_d(s, p|h)$ has the lowest value.

While straightforward, this solution entails two issues. First, during inference we would need to compute the distance transform $d(g, p)$ for each $g \in G$ and $p \in \mathcal{P}$. Computing this map for each primitive is costly and would increase the inference time of the model. Second, if we do not consider how well a primitive $p$ matches a stroke $s$, we may have misleading training signals for $h$. To clarify, let us consider a simple example, where $s$ is a full circle and $p$ a simple straight line. In such case, the loss $\mathcal{L}_d(s, p|h)$ would be high even if $h$ predicts the best possible alignment. This means that the loss would be dominated by primitives that, such as $p$, cannot represent the stroke $s$, making $h$ focus on an ill-posed problem rather than on matching compatible primitive-stroke pairs.

To address both issues, we inject the compatibility between a stroke and a primitive in the loss function. With this aim, we modify the stroke encoder as $f : \mathcal{S} \to \mathbb{R}^{2d}$ and, given an input $y$, we divide its embedding into two d-dimensional parts $z_y = [z_y^h, z_y^\phi] = f(s)$, where $z_y^h$ will be the part used to compute the alignment function through $h$ and $z_y^\phi$ will be used to compute the similarity between strokes/primitives. Given this embedding function, we calculate the relative similarity between a target stroke $s$ and a primitive $p$ as:

$$\phi(s, p) = \frac{\exp(\bar{z}_s^{\phi\intercal} \bar{z}_p^\phi / \kappa)}{\sum_{q \in \mathcal{P}} \exp\left(\bar{z}_s^{\phi\intercal} \bar{z}_q^\phi / \kappa\right)} \tag{5.4}$$

where $\kappa$ is a temperature value, and $\bar{z}_y^\phi$ is the L2-normalized version of $z_y^\phi$. Note that while $\phi$ needs to be invariant to the particular poses of $s$ and $p$ to score their compatibility, $h$ in Eq. (5.1) needs to capture their pose to better align them. These conflicting objectives are what lead us to split the output of $f$ in two parts. With the compatibility scores, we can define our final loss as:

$$\mathcal{L}(s, \mathcal{P}|h, f) = \sum_{p \in \mathcal{P}} \phi(s, p)\ \mathcal{L}_{\text{d}}(s, pT_s^p). \tag{5.5}$$

With this formulation, the lower the compatibility $\phi(s, p)$ between a primitive $p$ and the stroke $s$, the lower the weight of the distance transform loss between $p$ and $s$. Notably, the lowest value of $\mathcal{L}(s, \mathcal{P}|h, f)$ is achieved when i) the transformation matrices computed through $h$ align all primitives to the target stroke in the best way (w.r.t. the distance transforms), and ii) the primitives with the highest compatibility scores are the ones that better match the target stroke. Thus, minimizing $\mathcal{L}(s, \mathcal{P}|h, f)$ forces $h$ to output correct transformation matrices and $f$ to encode similar strokes close in the second half of the embedding space, fulfilling both our goals. We name the full model composed of $f$, $h$ and $\phi$ our *Primitive Matching Network* (PMN). Fig. 5.2 shows the PMN pipeline.

### 5.3.2 Replacing strokes with primitives

After learning $f$ and $h$, we can replace strokes with primitives at test time. In particular, since computing the distance transform for each possible primitive is costly, we can directly use $f$ and $\phi$ to select the best matching primitive for a given stroke. Specifically, given a stroke $s$ of a sketch $\mathbf{s}$, we replace it by:

$$\hat{p} = \arg\max_{p \in \mathcal{P}} \ \phi(s, p) \tag{5.6}$$

where $\hat{p}$ is the best-matching primitive. Given the primitive $\hat{p}$, we can now compute the corresponding alignment matrix as $T_s^{\hat{p}}$ from Eq.equation 5.1, and the abstracted sketch $\hat{\mathbf{s}}$ as:

$$\hat{\mathbf{s}} = \{\hat{p_1}^\intercal T_{s_1}^{\hat{p_1}}, \cdots, \hat{p_n}^\intercal T_{s_n}^{\hat{p_n}}, \} \tag{5.7}$$

where $n = m(\mathbf{s})$ is the number of strokes in $\mathbf{s}$. We highlight that our formulation is agnostic to the number of strokes in a sketch, the shape and number of primitives in $\mathcal{P}$, and the number of points composing each stroke.

## 5.4 Experiments

In this section, we present our experimental results. We first discuss our experimental setting (Section 5.4.1) and show results on sketch classification (Section 5.4.2) and fine-grained sketch-based image retrieval (Section 5.4.3) under a limited communication budget. Finally, we study the impact of the primitives (Section 5.4.4) and show qualitative analysis on the abstract representations (Section 5.4.5).

### 5.4.1 Experimental setting

**Datasets and benchmarks.** Following previous works on sketch abstraction [Muh+19; Muh+18] we test our model on sketch classification using Quickdraw [HE18], and on fine-grained sketch-based image retrieval (FG-SBIR) on ShoeV2 [Yu+17a] and ChairV2 [Yu+17a].

For **Quickdraw**, we follow [Muh+18] and select, 630k sketches from nine semantic categories (cat, chair, face, fire-truck, mosquito, owl, pig, purse and shoe). In this benchmark, we train a classifier on the original training sketches (details in the Supplementary), testing it on sketches


Figure 5.3: **Message content**. Human-drawn sketches (top) are split into messages of three points each (gray-coded). A primitive from the PMN representation (bottom) encodes more semantic information while requiring the same bytes for a single message.

abstracted using PMN or the competing methods given a specific budget (details below). We measure the performance as classification accuracy of the pretrained classifier given the abstracted inputs.

**ShoeV2** comprises 5982 training and 666 testing image-sketch pairs of various shoes. For this task, we train a Siamese network [Son+17] on the original training sketches with the same architecture of [Bhu+20; Muh+18], replacing the standard triplet loss with a contrastive objective[2]. We measure the image-retrieval accuracy (top-10) of this network on test sketches abstracted using either PMN or one of competing methods, with the abstraction conditioned on a given budget.

**ChairV2** contains 952 training and 323 testing pairs of chairs. For this dataset, we follow the FG-SBIR evaluation protocol described for ShoeV2.

**Implementation details.** We train two neural networks $f$ and $h$ as described in Section 5.3. The stroke encoder $f$ is a 6-layer Transformer [Vas+17], each with 8 self-attention heads. In all datasets, sketch data is represented as a list of points with their 2D coordinates and a binary label denoting whether the human is drawing or lifting the pen. We use the latter label to identify strokes. We feed as input to $f$ the sequence of 2D-points of a stroke, together with an extra token used to obtain the final stroke embedding. We implement $h$ as a 3-layer MLP that takes the concatenated embedding $z_p^h$ and $z_s^h$ as input. We use 7 predefined primitives $\mathcal{P}$, as shown in Fig. 5.1, as they can represent a wide variety of human strokes. We restrict $T_s^p$ to be a composite transformation of rotation, anisotropic scale, rotation in sequence, since we found it to be flexible enough to represent a wide variety of hand-drawn strokes. The translation is directly taken from the coordinates of $s$ and not predicted. The Supplementary contains more details about the transformation. Hyperparameters $\gamma = 6$ and $\kappa = 0.2$ are the same on all datasets and chosen by performing grid-search on a validation set.

**Budget computation.** To quantify the level of abstraction of our primitive representation, we adopt a similar evaluation procedure as in [Muh+19]. Instead of measuring classification accuracy of full sketches, the evaluation is done at different subsets of the full sketch given a budget, amounting to different levels of sketch compression. Concretely, we test at budgets of 10%, 20% and 30% of the original sketch's information content to focus on the high compression regime. To

[2]We found the contrastive objective to stabilize and speed up the training without sacrificing retrieval accuracy.

compute the budget, we follow the same procedure of [Muh+18], considering a single message as made of three stroke points, i.e. three sets of 2D coordinates (see Fig. 5.3). Note that each message contains information equivalent to six floating points values and a categorical value indicating to which stroke the points belong. This is the same amount of information as a single primitive of our proposed model, defined as a 2D-translation, 2D-scale, 2D-rotation for its transformation and a categorical label indicating which of the 7 primitives is used. When evaluating the budget on human sketches, each message corresponds to three points of a hand-drawn stroke while, when using our abstract representation, each message is a single primitive. Given a $N\%$ budget, we calculate the number of messages that can be communicated as the $N\%$ of the total messages forming the sketch. In Fig. 5.3, we illustrate what constitutes a message for human-drawn sketches and our primitive representations.

**Compared methods.** There are two ways in which a sketch can be abstracted. The first is by keeping the input representation untouched (i.e. original hand-draw strokes) but ranking the messages based on their importance for preserving the sketch content. Given a budget, we can then select only the most important subset of the messages. This is the approach of standard sketch abstraction methods [Muh+19; Muh+18]. We categorize this strategy as *Selection*-based abstraction.

The second strategy is orthogonal and simplifies the sketch by grounding strokes to shapes in a fixed vocabulary, as in our PMN. This strategy does not define any ranking for the messages, but achieves abstraction by changing the stroke itself. We categorize this strategy under the name *Shape*-based abstraction. In the experiments, we consider both type of approaches.

**Selection-based.** For this category, we consider two state-of-the-art methods: *Deep Sketch Abstraction* (DSA) [Muh+18] and *Goal-Driven Sequential-data Abstraction* (GDSA) [Muh+19]. DSA and GDSA are reinforcement learning methods that learn to order messages based on the performance on a downstream task. Specifically, DSA models the importance of each stroke by means of a classification (retrieval) rank-based reward, encouraging the target class (photo instance) to be highly ranked at all communication steps. GDSA is a more general strategy, applicable to various type of data. It directly uses the accuracy on the downstream task as reward function for the reinforcement learning agent, enforcing that the performance is preserved when the number of messages increases.

**Shape-based.** Since PMN is the first approach, we did not find other competitors in the literature addressing the same abstraction problem. As additional baseline we consider Shape Words (SW) [Xia+15], proposed in the context of sketch-based image retrieval. SW uses an heuristic algorithm to split the original strokes into multiple parts, fitting either a line or an arc to each part through Least Squares. Since SW cannot use arbitrary primitives, we use the same set of the original paper, i.e. lines and arcs. When PMN and SW are applied alone, the message order is the same on which the original strokes were drawn.

**Shape+Selection-based.** Since the two type of approaches are orthogonal, it is interesting to test if they can benefit each other. For this purpose, we also test other two models, combining GDSA with SW and our PMN.

| Abstraction method | | Budget (%) | | | |
|---|---|---|---|---|---|
| Type | Name | 10 | 20 | 30 | 100 |
| Selection | DSA [Muh+18] | 20.12 | 43.85 | 64.04 | **97.20** |
| | GDSA [Muh+19] | 26.88 | 51.65 | 71.60 | |
| Shape | SW [Xia+15] | 51.21 | 68.20 | 75.60 | 78.30 |
| | PMN | 67.08 | 83.69 | 89.15 | 91.78 |
| Selection +Shape | SW+GDSA | 62.70 | 74.87 | 77.61 | 78.30 |
| | PMN+GDSA | **77.22** | **87.79** | **90.23** | 91.78 |

Table 5.1: Classification accuracy on Quickdraw at budgets of 10%, 20% and 30% evaluated with a classifier trained on the original human-drawn sketches.

### 5.4.2 Sketch classification

In Tab. 5.1, we report the classification accuracy for both our PMN and the competitors on Quickdraw for budgets 10%, 20%, 30%, and 100% as reference. From the experiments, we can see that methods SW and PMN, based on shape abstraction, outperform by a margin DSA and GDSA, based on message selection. This is a direct consequence of using shapes as messages rather than original stroke parts, since the former can communicate much more semantic information in a single message. We see the largest gain at low budgets, e.g. at a 10% budget, DSA achieves 20.12% accuracy, and GDSA 26.88%, whereas SW reaches 51.21% and PMN obtains 67.08%, outperforming the rest significantly. This shows how PMN is better than SW at preserving the content of the sketch. This is a consequence of the higher flexibility in terms of 1) shapes that PMN can use and 2) precision of the alignment procedure, guided by the distance transform loss rather than Least Squares on heuristically selected points. The trend is similar at 20% and 30% budgets, at which point PMN achieves an accuracy of 89.18% against 75.60% of SW and 71.60% of GDSA. Notably, abstracting strokes with PMN is not lossless and the data distribution is different from the classifier's training data such that the accuracy at 100% of PMN (91.78%) is lower than using human sketches (97.20%). On the up side, this allows PMN to reach an accuracy close to the upper bound of the original sketches at already 30% budget showing that PMN well retains the semantic of the original dataset.

Finally, if we couple a selection-based method (GDSA) with a shape-based ones, we see a consistent improvement of performance, with an improvement of 10% (77.22% accuracy) at 10% budget for PMN+GDSA over simple PMN. Despite the improvement, SW+GDSA achieves lower performance than PMN alone at every budget (e.g. 62.70% at 10%), showing again how the abstraction of PMN is more precise than SW one.

### 5.4.3 Sketch-based image retrieval

In Tab. 5.2, we show the results of our PMN abstractions and the competing methods in the fine-grained sketch-based image retrieval (FG-SBIR) task for the ShoeV2 (left) and ChairV2 (right) datasets. Similarly to classification, we report the results at three different budgets: 10%, 20% and 30%.

| Abstraction method | | ShoeV2, Budget (%) | | | | ChairV2, Budget (%) | | | |
|---|---|---|---|---|---|---|---|---|---|
| Type | Name | 10 | 20 | 30 | 100 | 10 | 20 | 30 | 100 |
| Selection | DSA [Muh+18] | 10.96 | 18.32 | 26.88 | **75.22** | 16.72 | 31.58 | 45.20 | **86.99** |
| | GDSA [Muh+19] | 14.86 | 21.32 | 31.08 | | 20.74 | 33.13 | 47.68 | |
| Shape | SW [Xia+15] | 15.47 | 25.53 | 29.13 | 29.82 | 28.79 | 45.82 | 48.92 | 51.27 |
| | PMN | 29.58 | 48.50 | 54.35 | 56.55 | 53.87 | 70.59 | 73.99 | 75.92 |
| Selection +Shape | SW+GDSA | 19.96 | 28.97 | 29.27 | 29.82 | 35.60 | 47.98 | 50.77 | 51.27 |
| | PMN+GDSA | **36.18** | **50.45** | **55.10** | 56.55 | **63.15** | **73.68** | **75.23** | 75.92 |

Table 5.2: Fine-grained sketch-based image-retrieval (FG-SBIR) results (top-10 accuracy) on ShoeV2 and ChairV2 at budgets of 10%, 20% and 30% evaluated with a retrieval network trained on the original sketch-image pairs.

FG-SBIR has different challenges from sketch classification, since the communicated representation should precisely capture the specific characteristics of an instance rather than the shared ones of object categories. Despite our PMN abstraction smooths the specific details of strokes when grounding them to drawing primitives, it still preserves the most recognizable characteristics of an instance given a specific budget. Overall, the results are consistent with the ones on sketch classification, with PMN achieving the best results in both datasets and for each level of abstraction. For instance, at 10% budget, PMN achieves a retrieval accuracy of $29.58\%$ on ShoeV2 and $53.87\%$ on ChairV2, surpassing by a comfortable margin SW (i.e. $15.47\%$ on ShoeV2, $28.79\%$ on ChairV2) and selection-based models (e.g. GDSA, $14.86\%$ on ShoeV2, $20.74\%$ on ChairV2).

As a direct consequence of the inherent challenges of this FG-SBIR (requiring more detailed information), we see that the higher the budget, the higher the the gap between PMN and the competitors. With $30\%$ budget, PMN achieves $54.35\%$ accuracy on ShoeV2 and $73.99\%$ on ChairV2 while SW best result is $29.13\%$ on ShoeV2 and $49.92\%$ on ChairV2 and GDSA achieves $31.08\%$ on ShoeV2 and $47.68\%$ on ChairV2. SW shows an opposite trend, with the performance gap with selection-based methods becoming smaller as the budget increases, performing lower than GSDA on ShoeV2 for a 30% budget. These results highlight that PMN makes a more precise use of the available primitives, achieving the best trade-off between compression and distinctiveness of the sketch representation.

As expected, coupling PMN with GDSA leads to the best results overall (e.g. 36.18% on ShoeV2 and 63.15% on ChairV2 at 10%), with the performance of PMN alone consistently surpassing the ones of SW+GDSA (e.g. 19.96% on ShoeV2 and 35.60% on ChairV2 at 10%), highlighting that while selection and shape-based methods are complementary, it is fundamental that the latter precisely reconstructs the input, something achieved by PMN and not by SW.

### 5.4.4 Ablation study

In Tab. 5.3, we analyze the importance of the primitive shapes by evaluating the PMN model with different subsets of primitives for Quickdraw and ChairV2. We use the PMN model trained with all seven primitives and at test time only provide a subset of them. We start with the most commonly

| Primitives | Quickdraw (Classification) | | | | ChairV2 (FG-SBIR) | | | |
|---|---|---|---|---|---|---|---|---|
| | Usage (%) | Budget (%) 10 | 20 | 30 | Usage (%) | Budget(%) 10 | 20 | 30 |
| | 22.72 | 45.99 | 70.58 | 79.82 | 39.17 | 41.79 | 62.53 | 69.34 |
| + | 20.97 | 62.48 | 79.68 | 86.15 | 10.63 | 42.72 | 63.77 | 69.65 |
| + | 19.60 | 62.04 | 79.92 | 86.63 | 15.89 | 43.03 | 64.08 | 69.65 |
| + | 15.84 | 64.07 | 81.26 | 87.81 | 13.81 | 43.96 | 64.70 | 69.96 |
| + | 8.67 | 64.98 | 82.64 | 88.85 | 7.57 | 44.58 | 65.94 | 70.27 |
| + | 6.20 | 66.93 | 83.59 | 89.12 | 5.79 | 49.22 | 69.34 | 73.37 |
| + | 6.00 | 67.08 | 83.69 | 89.15 | 7.12 | 53.87 | 70.59 | 73.99 |

Table 5.3: Results of PMN with different primitives on Quickdraw (acc.) and ChairV2 (top-10 acc.). Primitives added one at a time in order of usage in Quickdraw.

used shape, the arc, and add one primitive at a time in the order of their usage frequency in the Quickdraw dataset. While the arc alone does not provide enough flexibility in Quickdraw, with only two primitives, arc and circle, our PMN model achieves a higher classification accuracy than both GDSA and SW with 62.48% (vs. 51.21% in SW) at a 10% budget and 86.15% (vs. 75.60% in SW) at a 30% budget. To put these results into perspective, SW is able to represent line, circle and arc shapes, so even without using lines the PMN model can better fit shapes and reconstruct sketches while retaining their semantics. This is particularly evident for ChairV2, where, even by using only arcs, PMN surpasses SW at all budgets (e.g. 41.79% vs 28.79% at 10% budget).

As more shapes are added, there are diminishing returns in increasing classification accuracy for Quickdraw. However, every primitive contributes to the performance our model achieves. The triangle, square and U-shape stand out to provide a significant improvement despite their relatively low usage of 8.67%, 6.20% and 6.00% respectively. Interestingly, on ChairV2 we see a more monotonic increase in the performance (e.g. from 44.58% to 49.22% when adding squares at 10% budget). This is expected, since FG-SBIR requires a more precise reconstruction of the original sketch, thus having more primitives helps in better modeling the specificity of each stroke, improving the overall results.

As a final note, while there are many options on which primitives to include, these results validate the choice of these seven primitives. Nonetheless, depending on the dataset and use case, other choices could be considered.

### 5.4.5 Qualitative analysis

**How are objects represented through primitives?** An interesting aspect of PMN is that we can now compare strokes across different sketches, extracting possible patterns. To show one possible application, in Fig. 5.4, we analyze the use of primitives when reconstructing Quickdraw classes. We show a representative abstracted sample of each class and the distribution of the primitives per class.

When inspecting the primitive distributions, we observe that the most used primitives are arcs and circles. As shown in our ablation study (cf. Tab. 5.3), using these two primitives alone can

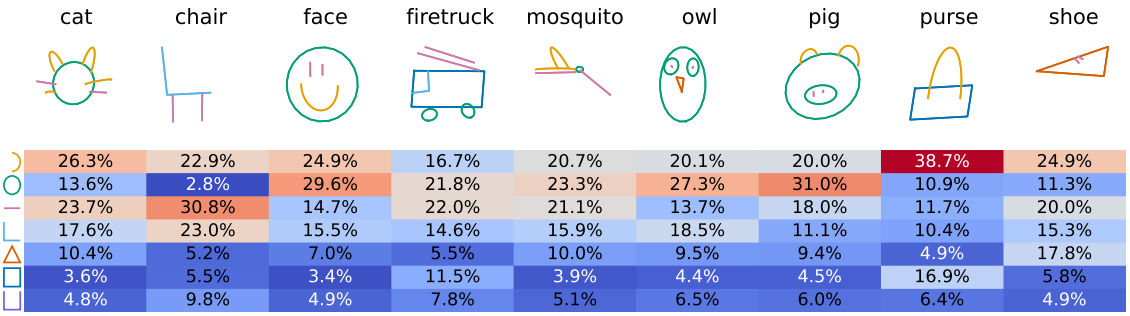



Figure 5.4: **Primitive analysis on Quickdraw**. Primitive representation of each Quickdraw class and the usage frequency of each primitive per class (in %).

already cover a lot of variation on human strokes. Common use cases for arcs include the ears in animals, smiles in faces and handles in purses. Circles most frequently represent heads in animals and faces and firetrucks' wheels. The body of the firetruck and the purse are often represented by rectangles. These correlations can be observed when comparing the average distribution of primitives per class, e.g. more frequent use of line and corner in chairs or rectangle and arc in purses than in other classes.

Fig. 5.4 also shows a limitation of our PMN model. PMN tries to match one primitive to each human stroke. However, when a stroke cannot be easily represented by a primitive, PMN may provide inaccurate representations. This is the case of the shoe class, where the main part of the shoe is usually drawn in a single stroke with a closed L-shape. In this case, PMN approximates this L-shape with a triangle (17.8% of shoe primitives are triangles, more than in any other class) that, despite driving the semantic of the sketch, provides a less accurate abstraction. In the future it would be interesting to address such cases by either learning to split/merge strokes and their parts into simpler shapes or by learning the primitives $\mathcal{P}$ together with PMN.

**Representations at different budgets.** We inspect some example qualitative results of our model in Fig. 5.5, showing sketch abstractions with varying compression budgets. We can see that partitioning the original strokes into three-point sub-strokes results in unrecognisable sketches even if GDSA is used to optimize the selection order. On the other hand, both SW and our PMN preserve the semantic much better given the same budget levels (e.g. shoe, bottom right). However, the additional flexibility allowed by PMN results in a much more faithful abstraction than SW, as exemplified by the body and ladder of the firefighting truck (top left), which are both represented by unnaturally rounded shapes by SW. Even when SW and PMN select the same shapes, PMN better aligns them to the original stroke, as can be seen from the circle used as seat of the chair (top right), or the arc used as left-ear of the cat (bottom left). This confirms the advantage of our self-supervised alignment objective w.r.t. the less flexible Least Squares solution of SW.

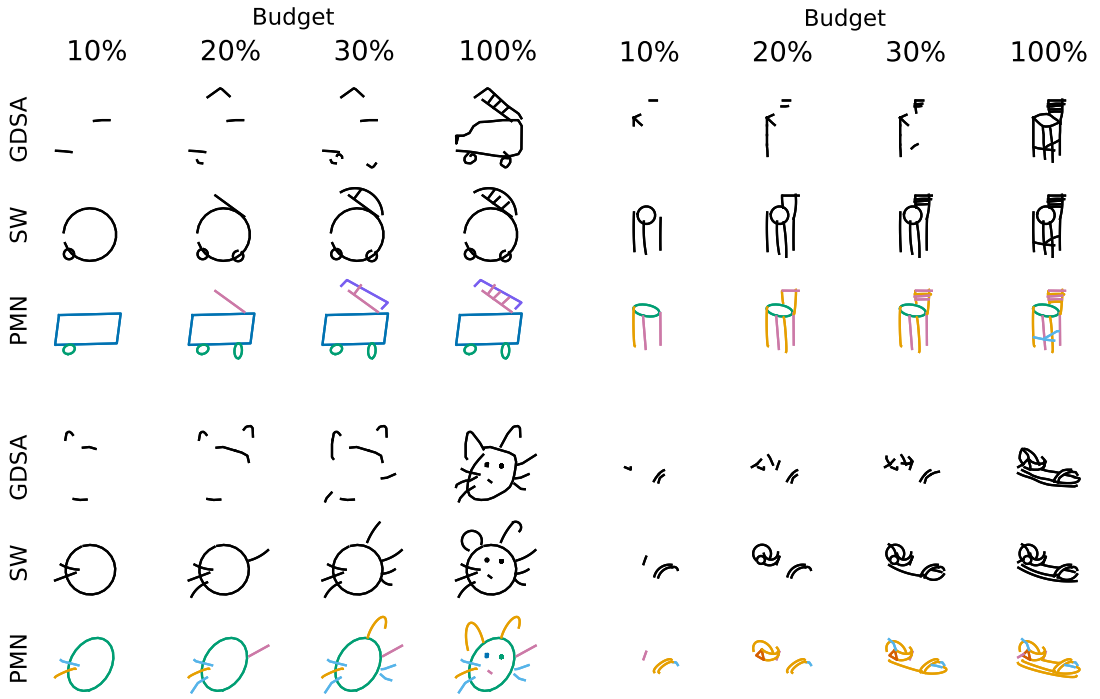



Figure 5.5: **Qualitative example of sketches at different budgets**. We show example sketches of Quickdraw (left), ChairV2 (top right) and ShoeV2 (bottom right) at 10%, 20%, 30% budgets when using GDSA, SW, PMN. Primitive color legend: ☽ ○ — ∟ △ □ ∪.

## 5.5 Conclusion

Motivated by how humans abstract object representations in interpretable messages when playing communication games, in this paper we proposed a new representation learning task, Primitive-based Sketch Abstraction, where the goal is to represent a sketch with a given set of simple drawing primitives. To address this task we proposed a model, Primitive-Matching Network, that maps each stroke of a sketch to its closest drawing primitive and predicts the affine transformation to align them. We overcome the lack of annotation for stroke abstractions by developing a self-supervised objective using the distance transforms of the primitives and the target strokes. Experiments show that our model surpasses standard sketch abstraction methods on sketch classification and sketch-based image retrieval at a given budget. Differently from hand-drawn strokes, our PMN abstraction is highly interpretable and leads to new types of sketch analyses, comparing sketches by means of their primitives.

# 6

# LANGUAGE-INSPIRED EXTENSIONS BEYOND COMMUNICATION

In this chapter, we present two lines of work that do not utilize communication directly, but extend the scope of this thesis by looking at aspects of language beneficial in communication scenarios.

Section 6.1 proposes a representation learning approach with the goal of transferring the compositionality inherent in language to the vision domain. We propose a probabilistic model of Gaussian mixtures that embed both images and textual labels into a joint latent space. Through aligning the two modalities by their components, our model learns to ground words in the image and generalize to unseen combination of objects in images.

Section 6.2 deals with the problem of semantic image synthesis by learning visual tokens that, when arranged together, produce natural images. These discrete tokens can be viewed as words from a vocabulary that are discovered from scratch from images. We propose to tie them closer together with semantic information to improve the synthesis process.

## 6.1 Compositional Mixture Representations for Vision and Text

Learning a common representation space between vision and language allows deep networks to relate objects in the image to the corresponding semantic meaning. We present a model that learns a shared Gaussian mixture representation imposing the compositionality of the text onto the visual domain without having explicit location supervision. By combining the spatial transformer with a representation learning approach we learn to split images into separately encoded patches to associate visual and textual representations in an interpretable manner. On variations of MNIST and CIFAR10, our model is able to perform weakly supervised object detection and demonstrates its ability to extrapolate to unseen combination of objects.

### 6.1.1 Introduction

For an artificial intelligence agent to gain an understanding of the world comparable to the one of humans', it should to be able to connect the visual world with its semantic meaning. There has been a substantial effort in learning image representations [Che+16; KW14] as well as text

representation [Dev+19; Mik+13; PSM14], capturing the semantic meaning and disentangling the variation factors in a way similar to how humans learn their surroundings. However, learning unsupervised visual representations from the image data, (e.g., through image reconstruction), can be challenging because there is no guidance towards an informative content (e.g., presence of objects) and uninformative content (e.g., the exact pixel location of the horizon), when there is no supervision involved or the downstream task is unknown [ZSE17]. [Ben17] hypothesises that language can be a good prior towards forming useful representations, i.e., things that are commonly described or talked about by people in visual data is information we would like to preserve in our representations.

Our goal is to build upon the idea of preserving the semantic information in the learned representations and learn an unified representation between the text and images, where we use the semantic structure of the text to disentangle the visual data. For instance, when a caption mentions a car, the corresponding representation of the image should not only include the same information, but should be able to report as well, which part of the image caused the representation of "car". Naturally, such a representation is not limited to hold only a single concept, but can be composed of several individual components, both on the text as well as on the image side. As text is already highly structural, its representation can be viewed as the aggregation of its words' meanings. Images could be then decomposed into patches associated with the building blocks that encode the semantics within the text.

By directly modeling the compositionality of the representation, it becomes possible to obtain several desirable properties. The first property is generalizability. In other words, learning a representation of its constituents is usually simpler than inferring a meaningful joint representation of a complex example. Hence, in novel scenarios, being able to robustly embed partial entities, which are well known from the training set, allow representations to generalize more easily. The second property is combinatorial extrapolation. Certain objects in the images might have high co-occurrence probability. Previously unknown object combinations during test time can cause a deep model to fail. Compositionality can overcome this bias as it allows to arbitrarily combine partial representations. The third and the final property is interpretability. Associating the visual components of the image to the semantics encoded by the textual components allows humans to develop a better understanding of how deep models form their complex representations and might also assist in identifying the causes behind possible failures.

In this work, we implement the idea of using textual guidance to learn compositional image and text representations. One goal of the representation learning is to create representations of the raw input data, which are useful for an a priori unknown downstream tasks. We choose to evaluate our approach on image retrieval and weakly supervised object detection. The latter task is particularly fitting as an aim to decompose images into their respective objects as given by the textual labels without having any bounding box supervision. Therefore, it allows to give a good indication about the compositionality and disentanglement of the image representation that is achieved by our model.

Our contributions are as follows: we present a novel model that learns a compositional Gaussian mixture representation for both the image and the text and matches them with a KL divergence loss.

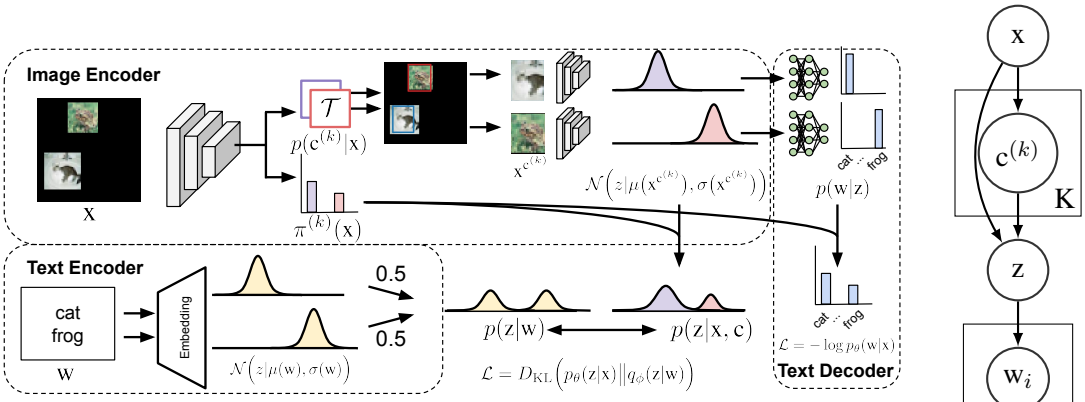



Figure 6.1: Compositional Mixture (CoMix) Model. The graphical model is shown in the (b) while (a) shows the architecture of our proposed CoMix model that learns two Gaussian mixtures, one is learned from the image and the other one is from the text data. The image encoder first uses a CNN to predict the Gaussian mixture weights $\pi^{(k)}(\mathrm{x})$ as well as the transformation parameters $p(\mathrm{c}^{(k)}|\mathrm{x})$ used by the spatial transformer module to extract image patches $\mathrm{x}^{c^{(k)}}$. Each patch is individually encoded by a second CNN into Gaussian distributions $\mathcal{N}(z|\mu(\mathrm{x}^{c^{(k)}}), \sigma(\mathrm{x}^{c^{(k)}}))$. A text decoder $p(\mathrm{w}|\mathrm{z})$ is learned with a negative log likelihood loss and ensures that textual information is contained in the representation. A text encoder embeds individual words into Gaussian components and then mixes them into the textual Gaussian mixure representation $p(\mathrm{z}|\mathrm{w})$. A KL-divergence loss allow to learn the correspondence between text tokens and image crops by matching the two representations. Without any bounding box supervision, our CoMix model learns to detect images.

The vision part of the model incorporates the spatial-transformer architecture to learn bounding boxes of the image parts corresponding to the different textual components. We evaluate our model on the altered MNIST and CIFAR10 datasets, where each image is composed of several images from the original dataset.

### 6.1.2 Related work

**Representation Learning.** We distinguish ourselves from most representation learning approaches on images in that we do not try to embed the complete image information in the latent code. Nonetheless, the goals of representation learning are shared across related work in this direction. MONet [Bur+19] also employs a compositional approach to scene understanding and reconstruction. It iteratively constructs a scene by its components such as objects and background elements and uses a VAE [KW14] architecture at each step. Similarly, IODINE [Gre+19] also employs a VAE iteratively to infer and reconstructs scene an object at a time. Related work has previously combined the spatial transformer architecture [Jad+15] with image representation learning both iteratively [Esl+16] and by processing the whole image at once [CP19].

**Multi-modal Learning.** When considering models working with multi-modal data, MVAE [WG18] applies the VAE setting to multiple modalities such as image and text and can be trained semi-supervised, but it does not employ a compositionality approach. Learning shared representations

between images and text is often done targeting a specific application such as retrieval [Zhe+20] or grounding sentences in images [Roh+16]. Work on associating objects with parts of language through visual-semantic alignment [KJL14; KL15] usually uses greater level of supervision at least on object detection. One such example is VisualBERT [Li+19] which combines the rich textual representations of BERT [Dev+19] with supervised object detectors to solve a variety of visual-language tasks. Similarly, [LZY20] uses natural language explanations to improve the performance of a visual classifier. LXMERT [TB19] and UNITER [Che+20b] demonstrate that rich representations learned from multi-modal vision and text data can benefit diverse tasks such as Visual Question Answering (VQA). The learned representations of CLIP [Rad+21] achieve zero-shot object recognition capabilities by training on a large dataset of unstructured image-text pairs. Recently, several works [Huo+21; Jia+21; Li+21; She+21] build upon the contrastive learning objective similar to CLIP to learned multi-modal representations in an unsupervised manner.

**Weakly Supervised Object Detection.** Following the advances in supervised object detection in models such as YOLO [Red+16], Faster-RCNN [Ren+15] and Mask-RCNN [He+17], more work is dedicated in solving the object detection task without bounding box labels, i.e. weakly supervised with only labels about which object are present in the image. WSDDN [BV16] works by pooling spatial regions on the last convolutional feature layer of a CNN and has been challenged by similar approaches such as C-MIDN [Gao+19] and PredNet [AJK19]. These approaches, however, do not offer the same level of introspection and interpretability as our CoMix.

### 6.1.3 Compositional Mixture Model

We present our proposed compositional mixture model (CoMix) in the following subsection. In our setting, we consider a joint data distribution $p(\mathrm{x}, \mathrm{w})$ of images x and text w with vocabulary $\mathcal{V}$. Our goal is to use a deep learning model to encode the visual signal coming from the images into a compositional representation, which entails all the semantic information that the components of the textual counterpart contains. At the same time, the representation should be interpretable in terms of which image region contributes to its components' representations.

#### 6.1.3.1 CoMix Overview

In our CoMix model, we choose to model the latent representation as a Gaussian mixture. By being a multimodal distribution, it is able to model complex data while having the desired property of being compositional as it consists of several simple Gaussian distributions. Hence, each textual component as well as inferred image patches are embedded into individual Gaussians before being mixed to form the final representation. Our CoMix model consists of three parts as depicted by Figure 6.1a. A text encoder $p(\mathrm{z}|\mathrm{w})$ takes as input the text tokens w and encodes each one of them into a latent mixture component.

Analogously, an image encoder maps the input image to another Gaussian mixture $p(\mathrm{z}|\mathrm{x})$, where each Gaussian component corresponds to a different spatial image region. By aligning the two mixture distributions with a KL-divergence loss term, CoMix learns to associate components of the text to the concrete image regions. Attending to separate image patches is learned end-to-end

by a spatial transformer module without requiring any extra supervision. Finally, a text decoder $p(\mathrm{w}|\mathrm{z})$ ensures that the learned Gaussian mixture representation retains all the textual information, effectively preventing degenerate solutions. Thus, CoMix is trained by supplying image-text paris $(\mathrm{x}, \mathrm{w})$ without any additional supervision as to how these two modalities are related, e.g. there is no bounding box supervision on where the text is grounded in the image.

#### 6.1.3.2 Text Encoder

A string of text w, such as a sentence, is composed of entities of interest, such as objects, that are also present in the image. Each word is embedded into a Gaussian latent component $\mathcal{N}(z|\mu(\mathrm{w}_i), \sigma(\mathrm{w}_i))$, where $\mu(\mathrm{w}_i)$ and $\sigma(\mathrm{w}_i)$ are determined by an embedding layer. The latent variables of all the words of the text are then combined into a Gaussian mixture model

$$\frac{1}{|\mathrm{w}|}\sum_i \mathcal{N}\Big(z|\mu(\mathrm{w}_i), \sigma(\mathrm{w}_i)\Big), \tag{6.1}$$

where each Gaussian of the mixture is weighted equally. This allows us to express a complex multi-modal representation as the sum of simple uni-modal Gaussian components representing its constituent words. Given a one-hot vector for each word, the text encoder is a single matrix that stores a trainable set of Gaussian parameters for each word.

#### 6.1.3.3 Image Encoder

A spatial transformer module [Jad+15] determines a total of $k$ image regions, which are embedded into the parameters of a Gaussian distribution. Such an architecture allows us to crop, translate, and scale portions of the original input image by defining an affine transformation $A_\theta$ that maps image pixel locations of the source and the target image:

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = A_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \tag{6.2}$$

where $(x^s, y^s)$ and $(x^t, y^t)$ are the source and target coordinates of pixel locations. The spatial transformer can be viewed as hard attention on the input image with differentiable transformation parameters $A_\theta$.

We employ a convolutional neural network (CNN) to learn $k$ different transformations $p(\mathrm{c}^{(k)}|\mathrm{x})$ from the input image. By applying the spatial transformer on each of these, we get $k$ different image crops $\mathrm{x}^{c^{(k)}}$, which are then individually encoded with another CNN that, in its turn, provides the parameters $\mu$ and $\sigma$ to a Gaussian distribution $\mathcal{N}(z|\mu(\mathrm{x}^{c^{(k)}}), \sigma(\mathrm{x}^{c^{(k)}}))$. Hence, each Gaussian component of the image is also associated to an image region by a hard-attention mechanism. The Gaussian components are combined through the categorical mixture distribution $\pi^{(k)}(\mathrm{x})$, which is anticipated from the original input image with the same network that predicts the $k$ transformations $\mathrm{c}^{(k)}$ to form the Gaussian mixture $\sum_k \pi^{(k)}\mathcal{N}(z|\mu(\mathrm{x}^{c^{(k)}}), \sigma(\mathrm{x}^{c^{(k)}}))$.

#### 6.1.3.4 Representation Learning

Our requirements for learning a meaningful representation are twofold: 1) The data likelihood under our graphical model needs to be maximized. Since the prediction of w depends only on z, this forces z to capture all the required textual information from the corresponding images x; 2) The representation should discard image details that are irrelevant for text prediction to obtain a correspondence of text and image components.

To address the former, we train a text decoder $p(\mathrm{w}|\mathrm{z})$ on top of the image representation. This results in the full graphical model depicted in Figure 6.1b. According to the graphical model, we can now define the joint distribution of our data $p(\mathrm{x}, \mathrm{w}) = p(\mathrm{w}|\mathrm{x})p(\mathrm{x})$, where we are particularly interested in the conditional

$$p(\mathrm{w}|\mathrm{x}) = \prod_i p(\mathrm{w}_i|\mathrm{x}) \tag{6.3}$$

$$= \prod_i \iint p(\mathrm{w}_i|\mathrm{z})p(\mathrm{z}|\mathrm{x}, \mathrm{c})p(\mathrm{c}|\mathrm{x})d\mathrm{c}d\mathrm{z} \tag{6.4}$$

$$= \prod_i \iint p(\mathrm{w}_i|\mathrm{z}) \sum_k \pi^{(k)}(\mathrm{x})\mathcal{N}\Big(z|\mu(\mathrm{x}^{c^{(k)}}), \sigma(\mathrm{x}^{c^{(k)}})\Big) \tag{6.5}$$

$$p(\mathrm{c}^{(k)}|\mathrm{x})d\mathrm{c}^{(k)}d\mathrm{z} \tag{6.6}$$

$$= \prod_i \sum_k \pi^{(k)}(\mathrm{x})\mathbb{E}_{\mathrm{z}\sim\mathcal{N}(z|\mu(\mathrm{x}^{c^{(k)}}),\sigma(\mathrm{x}^{c^{(k)}}))}\Big[p(\mathrm{w}_i|\mathrm{z})\Big] \tag{6.7}$$

where the crop parameters $\mathrm{c}^{(k)}$ are deterministically determined as a parametric function of x, which is represented by the transformer architecture.

By parameterizing the conditional distribution with our model parameters, we can define our objective to minimize the negative log-likelihood of the joint occurrence of pictures x and text w:

$$\min_\theta \mathbb{E}_{\mathrm{x},\mathrm{w}\sim\tilde{p}}\Big[-\log p_\theta(\mathrm{x}, \mathrm{w})\Big] \tag{6.8}$$

$$= H(\mathrm{x}) + \min_\theta \mathbb{E}_{\mathrm{x},\mathrm{w}\sim\tilde{p}}\Big[-\log p_\theta(\mathrm{w}|\mathrm{x})\Big] \tag{6.9}$$

where $H(\mathrm{x})$ is the entropy of the image data and as a constant is irrelevant for the optimization procedure. Specifically, the second term on the right-hand side is the negative log-likelihood of the text data given the image under our model and will be denoted as NLL. By minimizing NLL, we are effectively capturing the information required to predict w since the predictive text distribution $p(\mathrm{w}|\mathrm{z})$ depends only on the representation z.

Since we want the representation z to focus only on the image details which have a textual counterpart, the second condition aims to discard any image-specific detail from z. This can be done by minimizing the distance between the conditional distributions $p(\mathrm{z}|\mathrm{x})$ and $p(\mathrm{z}|\mathrm{w})$ to enforce consistency between encoded images and corresponding text. While we model $p(\mathrm{z}|\mathrm{x})$ directly, $p(\mathrm{z}|\mathrm{w})$ is intractable under our graphical model. Hence, we introduce a variational distribution $q(\mathrm{z}|\mathrm{w})$ to approximate the representation induced by the observation of the text data. We model $q(\mathrm{z}|\mathrm{w})$ as a Gaussian mixture distribution in which each component corresponds to a single word to

capture the compositional nature of the text. By minimizing the Kullback-Leibler (KL) divergence between $p(\mathrm{z}|\mathrm{x})$ and $q(\mathrm{z}|\mathrm{w})$ we are effectively minimizing the amount of image-specific information embedded into z, fulfilling our second requirement:

$$D_{\mathrm{KL}}\Big(p_\theta(\mathrm{z}|\mathrm{x})\Big\|q_\phi(\mathrm{z}|\mathrm{w})\Big) \geq D_{\mathrm{KL}}\Big(p(\mathrm{z}|\mathrm{x})\Big\|p(\mathrm{z}|\mathrm{w})\Big) \tag{6.10}$$

$$= D_{\mathrm{KL}}\Big(p(\mathrm{z}|\mathrm{x},\mathrm{w})\Big\|p(\mathrm{z}|\mathrm{w})\Big) \tag{6.11}$$

$$= I(\mathrm{z};\mathrm{x}|\mathrm{w}). \tag{6.12}$$

When $I(\mathrm{z};\mathrm{x}|\mathrm{w})$ is minimal, the representation $\mathbf{z}$ must contain only the information that can be determined through the text (as z and x become conditionally independent), discarding picture specific nuisances not mentioned in the textual description w.

Combining the KL-divergence term with NLL, we arrive at the loss function

$$\min_{\theta,\phi}\mathcal{L} = \min_{\theta,\phi} E_{\mathrm{x},\mathrm{w}\sim\tilde{p}}\Big[-\log p_\theta(\mathrm{w}|\mathrm{x}) \tag{6.13}$$

$$+ D_{\mathrm{KL}}\Big(p_\theta(\mathrm{z}|\mathrm{x})\Big\|q_\phi(\mathrm{z}|\mathrm{w})\Big)\Big]. \tag{6.14}$$

The KL-divergence between the two mixture models $p(\mathrm{z}|\mathrm{x})$ and $q(\mathrm{z}|\mathrm{w})$ is approximated by sampling from each Gaussian component using the reparameterization trick [KW14] resulting in a stochastic estimate of the loss.

#### 6.1.3.5 Image Area Loss

In the previous subsections, we provided a detailed description of how a compositional Gaussian mixture representation for both text and image may be learned. Learning the representation is the key part that makes CoMix implicitly split the image into elements that match the textual components. However, the crops that are learned by the spatial transformer are not guaranteed to be minimally enclosing the objects they represent. Since neural networks are able to learn arbitrary mappings from the image region to representation, the network can learn to attend to a larger image part than just the object, effectively including unnecessary information without harming the modeling performance.

We are specifically interested in the smallest bounding box per object to be encoded into a Gaussian component because a precise localization greatly helps model inspection and interpretability. Hence, we introduce another tunable loss term that penalizes the size of the image crops learned by the spatial transformer. Our final loss is given by

$$\min_{\theta,\phi}\mathcal{L} = \min_{\theta,\phi} E_{\mathrm{x},\mathrm{w}\sim\tilde{p}}\Big[-\log p_\theta(\mathrm{w}|\mathrm{x}) \tag{6.15}$$

$$+ D_{\mathrm{KL}}\Big(p_\theta(\mathrm{z}|\mathrm{x})\Big\|q_\phi(\mathrm{z}|\mathrm{w})\Big) \tag{6.16}$$

$$+ \lambda\Big|\prod_i A_{\theta,ii}\Big|\Big]. \tag{6.17}$$

where the product of the diagonal elements of the affine transformation matrix $A_\theta$ equal the area of the patches by the transformer architecture. The coefficient $\lambda$ regulates to which degree we would like to penalize the area of the image crops. In the experiments subsection below, we investigate the effect of this hyperparameter.

### 6.1.4 Experiments

In this subsection, we describe the datasets and experimental tasks and provide quantitative and qualitative results evaluating the representation learned by CoMix. Furthermore, we demonstrate the performance of our model on classification, image-retrieval, and weakly supervised object detection.

#### 6.1.4.1 Datasets and Setup

**Datasets.** We conduct experiments on two datasets, MultiMNIST and MultiCIFAR10, which represent variations of MNIST [Lec+98] and CIFAR-10 [Kri09], respectively. MNIST consists of 60K/10K training/test examples from 10 handwritten digits and CIFAR-10 contains 50K/10K training/test natural image examples from 10 classes.

In contrast, MultiMNIST and MultiCIFAR10 combine several original images randomly sampled onto the same canvas. For each example, we sample between one to four original images, scale them with a factor drawn from $\mathcal{N}(1.3, 0.1)$, and place them at random locations on the canvas. The canvas size is $56 \times 56$ for MultiMNIST and $80 \times 80$ for MultiCIFAR10. The presence of an object class in the image is indicated with the help of labels, regardless of how many times the same object appears. We sample training and test data once according to the original data sizes and then keep the datasets fixed across all the experiments. See Figure 6.3 for the examples of images from these datasets.

**Experimental setting.** All the compared models consist of the same neural-network architectures with the similar or same number of parameters. For MultiMNIST, the image encoder is a 3-layer convolutional network with BatchNorm and ReLU after each layer. For MultiCIFAR10, we use ResNet18 as the image encoder. For both datasets, we use an embedding layer for the text encoder, a 2-layer MLP with ReLUs as the text decoder, and a spatial transformer network consisting of a 3-layer convolutional network with BatchNorm and ReLU.

The number of the mixture components $k$ learned by the spatial transformer network is set to 5 across all the datasets such that the model has to learn to actively choose to use or not to use components. The decision about using a particular component is regulated by the mixture weights $\pi^{(k)}$, which is a learned output of the same network. The area loss coefficient $\lambda$ is set to 4 for both datasets (see subsection 6.1.4.3). We randomly split 10% of the training data as a validation set to tune the remaining hyperparameters. The model's code, experiments, and data generation will be made publicly available.

#### 6.1.4.2 Classification and Object Detection

**Task.** We evaluate our model on its ability to detect and classify objects in an image. For the classification, since there are multiple target classes per image, we measure the mean average precision (mAP) of the predictions of our model's text decoder. While our model has a direct supervision on the classification task, there is no supervision signal or loss on object detection. Solely the learning of a compositional representation as well as the spatial-transformer architecture facilitates

| | MultiMNIST | | MultiCIFAR10 | |
|---|---|---|---|---|
| | Cls | Det | Cls | Det |
| CNN | 95.87 | n/a | 73.99 | n/a |
| WSDDN | **99.63** | **88.27** | 75.42 | 49.47 |
| CoMix | **99.39** | **87.04** | **90.74** | **75.83** |

Table 6.1: Classification and object detection performance measured in mean average precision (mAP, in percent) on MultiMNIST and MultiCIFAR10 datasets.

the fact that our model naturally learns to detect objects without the ground-truth bounding-box supervision. Thus, the task setup is identical to weakly supervised object detection (WSOD).

The predicted bounding boxes come from the transformer networks predictions that select a region of the image to be used as one component in our mixture model. We evaluate on the ground-truth bounding boxes that are known from the data generation process, i.e., location and boundary of each individual original image. Following [Eve+], object detection is measured by mAP, where a detection is considered to be a true positive whenever the intersubsection over union (IOU) of the predicted and the ground-truth bounding box is greater than 0.5.

**Baselines.** Two baselines are introduced for the classification and object-detection tasks. Firstly, we isolate the image encoder of our model to do the classification directly from the input image to the label output without any representation learning denoted as CNN. This baseline is exclusively trained on a binary cross-entropy loss to predict the presence of the object classes in the image. Our model should match the CNNs classification performance to ensure that the representation learning does not hinder the prediction performance.

Secondly, we compare against Weakly Supervised Deep Detection Network (WSDDN) [BV16] serving as an object-detection baseline. WSDDN is an established backbone for the weakly supervised object-detection networks that relies on adaptive pooling over a convolutional feature map similar to modern supervised object detection networks such as Faster-RCNN [Ren+15]. Contrary to our approach, bounding-box proposals are not learned, but generated algorithmically using the selective windows search strategy [San+11].

**Results.** We report the classification and object detection results in Table 6.1. On classification, our CoMix model outperforms both baselines on MultiCIFAR10 by a large margin (90.7% vs 75.4%/74%) and is on par with WSDDN on MNIST. The close results on MultiMNIST can most likely be explained by getting close to the perfect classification results rather than choices in the model. We believe, the considerably higher performance of our model on MultiCIFAR10 can be attributed to the compositional modelling of the individual objects in the image. For this reason, mAP approaches ResNet18's classification performance of around 93% on the individual CIFAR10 images. Since our model processes image parts separately, we can generalize from the single image CIFAR10 classification performance to our more difficult MultiCIFAR10 dataset. Both baseline models back this property and, therefore, fall short in classification.

Similarly on the object detection, CoMix obtains a comparable performance as WSDDN on MultiMNIST, but outperforms it on MultiCIFAR10 (75.8% vs. 49.5%). Being able to flexibly

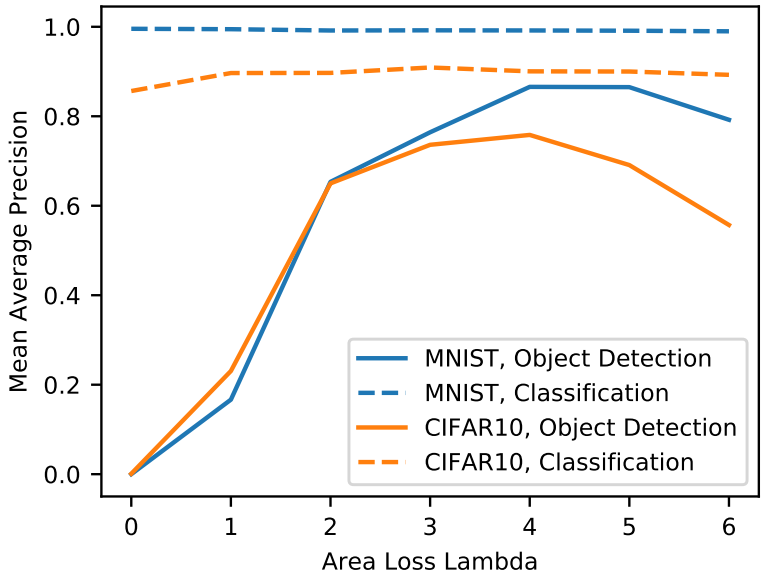



Figure 6.2: Mean average precision of our CoMix model on both classification as well as object detection with varying values of $\lambda$. The hyperparameter $\lambda$ enables an area loss term that tightens bounding box predictions around objects.

| | MultiMNIST | | | | MultiCIFAR10 | | | |
|---|---|---|---|---|---|---|---|---|
| | avg. Rank | R@1 | R@5 | R@10 | avg. Rank | R@1 | R@5 | R@10 |
| *Regular* | | | | | | | | |
| Gauss | 6.85 | 21.09 | 64.84 | 86.72 | 16.47 | 14.06 | 32.03 | 55.47 |
| CoMix | **2.33** | **64.06** | **91.41** | **96.88** | **2.5** | **46.09** | **93.75** | **97.66** |
| *Skewed* | | | | | | | | |
| Gauss | 15.96 | 31.25 | 40.63 | 53.13 | 22.35 | 7.81 | 35.16 | 49.22 |
| CoMix | **2.45** | **57.03** | **89.84** | **98.44** | **5.45** | **47.66** | **77.34** | **85.94** |

Table 6.2: Image retrieval results of our CoMix model compared to the Gauss baseline where we replace our composition Gaussian mixture latent representation with a single Gaussian distribution. *Regular* refers to the normal version of our datasets MultiMNIST and MultiCIFAR10, whereas *skewed* indicates a more difficult version where the combination of seen objects is highly correlated during training. Scores refer to the average retrieval rank and recall percentage at rank 1, 5 and 10.

learn the location of the objects instead of relying on the statically generated bounding boxes helps our model to more accurately pinpoint the image region responsible for a class label. Both results show the benefit of modelling image parts in isolation instead of the whole image as once (CNN) and of learning bounding boxes for our components with the spatial transformer module.

#### 6.1.4.3 Inspecting the Area Loss

We argue that our area loss is essential for our model to learn bounding boxes that enclose the object tightly, which is important for both the detection performance and model inspection. In order to validate this claim, we train our model on both datasets with varying values of $\lambda$. A $\lambda$ of 0 indicates that no area-loss term was applied while the bigger $\lambda$ gets, the more bounding boxes will be penalized for their size. In Figure 6.2, we report mAP for both the classification and object detection for different values of $\lambda$. Interestingly, the classification performance is largely unaffected with increasing $\lambda$ and it is even higher for MultiCIFAR10 for any $\lambda > 0$ compared to no area loss.

The mAP of the weakly supervised object-detection task steadily increases for both datasets until $\lambda$ reaches a value of 4. A value greater than 4 enforces bounding boxes that are too small and, thus, detection accuracy diminishes. Based on this analysis, we set $\lambda = 4$ for our experiments. Most importantly, introducing this loss term only benefits both the detection and classification in the range that we tested, so that its use can be easily justified.

#### 6.1.4.4 Image Retrieval

**Task.** To evaluate our model's representation learning capabilities, we introduce an image-retrieval task. Given a text sample, the task is to find the best matching image from a set of images. If our model learns a good shared representation of image-text pairs, it should be possible to retrieve the image matching a query text by finding the image representation closest to the inferred text representation. With the use of our model, we first encode both the text and all the images into Gaussian mixtures independently. Then, we score the text representation to all the image representations. Since these representations are multi-modal distributions, we resort to calculating the piece-wise distance between Gaussian components of all the possible text-image pairs. The distance is defined by the average euclidean distance of each text component's mean to the closest image component's mean.

Based on the distance score, we can rank the images from the high to low similarity. To evaluate retrieval performance, we report the average rank of the ground-truth matching image as well as the recall at ranks 1, 5, and 10.

**Ablation Study.** A key advantage of our model is that its representations are compositional. To study the impact of having a compositional representation, we create an ablation model (Gauss), where the latent variable z is a Gaussian instead of a mixture of Gaussians. In this ablation model, the image encoder directly infers the latent variable z from the input image without the compositional spatial transformer. Moreover, the text encoder's embedding layer is replaced by a 2-layer MLP such that the representation of the full sentence can be learned as it can no longer

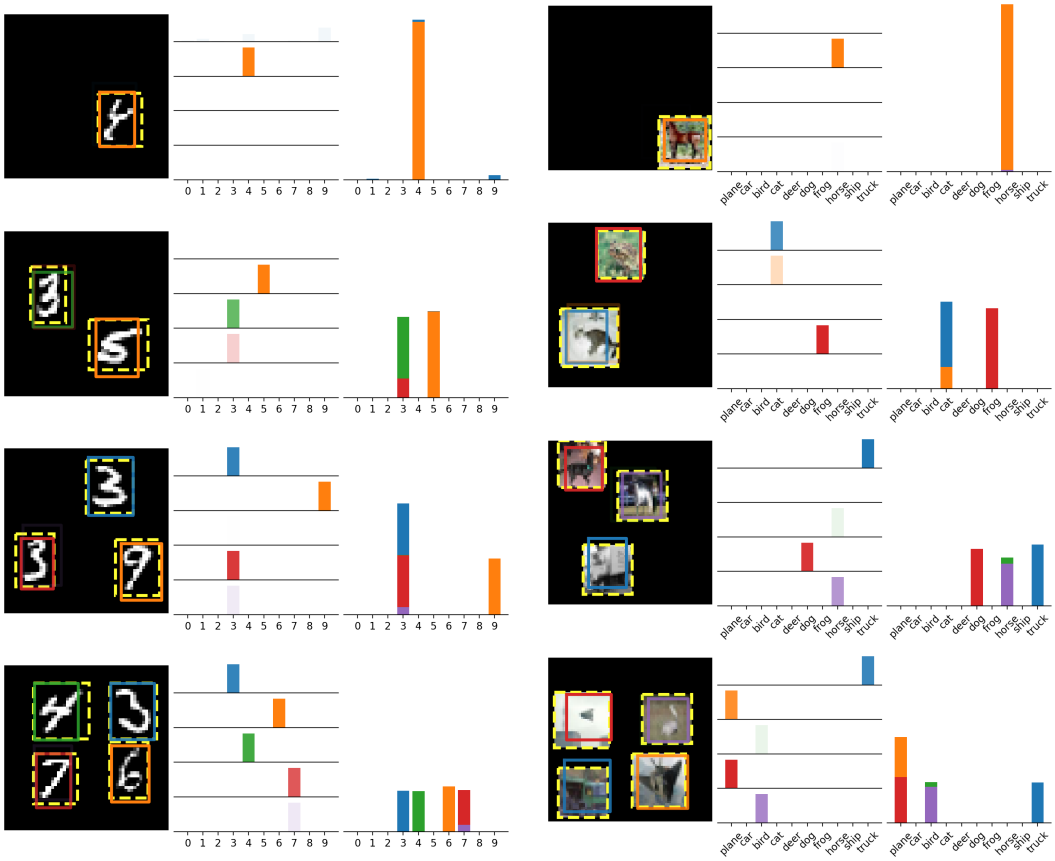



Figure 6.3: Qualitative results on MultiMNIST and MultiCIFAR10. For each dataset there are three columns: 1) data example visualiing the ground truth bounding boxes (yellow, dotted line) and the predicted bounding boxes, one for each component (various colors, solid line); 2) Categorical prediction of object for each individual component corresponding to bounding box with sample color; 3) Categorical mixture prediction of the whole image with colors indicating the contribution of each component.

be composed of simple embeddings. Since the ablation model only uses a single Gaussian, the retrieval distance is calculated by taking the euclidean distance between the means of the text and image latents.

**Skewed Datasets.** To better contrast two different representation-learning approaches, we conduct experiments on a skewed version of MultiMNIST and MultiCIFAR10. During the training, there are three groups of object classes, where each object only co-occurs with objects from the same group. For instance, for MultiMNIST, only the digits (0, 1, 2) would be seen together on the same canvas and similarly for digit groups (3, 4, 5) as well as (6, 7, 8, 9).

During the test phase, all the digits can co-occur with any other digits as in our initial MultiMNIST and MultiCIFAR10 definition. The purpose of this skewed dataset versions is to show how compositionality can help overcome dataset bias, where objects are highly correlated in the training data, but might occur in the novel combinations during the test time.

**Results.** Image-retrieval results are reported in Table 6.2. We observe, learning a Gaussian mixture representation achieves a lower retrieval rank on average and a higher recall at all the measured

levels as compared to a single Gaussian representation. There are two reasons for these results. Firstly, the Gaussian mixture of CoMix is multimodal and, thus, can model more complex latent distributions. Secondly, the compositionality of CoMix allows it to extrapolate to the unseen combination of objects during the test time. We observe this property when comparing the performance loss between the regular and skewed datasets. CoMix can maintain a low mean rank (2.3 $\rightarrow$2.5; 2.5 $\rightarrow$ 5.5), while average rank for the Gauss ablation gets considerably worse (6.9 $\rightarrow$ 16.0; 16.5 $\rightarrow$ 22.4) on both datasets. In conclusion, the explicit modelling of a compositional representation greatly benefits CoMix's generalization and extrapolation performance of the downstream tasks.

#### 6.1.4.5 Qualitative Results

To illustrate the datasets and inspect our model, we present qualitative example of CoMix applied to MultiMNIST and MultiCIFAR10 in Figure 6.3. For each dataset, we show an example for 1 to 4 objects in the image. In each data image, we visualize the ground-truth bounding boxes of the objects (dashed yellow line) as well as the object detections by our CoMix model (blue, orange, green, red, purple boxes).

Each image is accompanied by two additional columns. The first column displays the textual prediction of the text decoder applied onto the representation of the patch with the same color. The opaqueness of the bounding boxes and the bar plots is proportional to the categorical mixture distribution $\pi$ that is predicted by the spatial transformer network. In other words, CoMix predicts the number of objects in the image through the $\pi$ values of each component and sets a components' $\pi^{(k)}$ (close) to zero when less than the maximum number of components are needed. The second additional column is the joint categorical class prediction of the whole image that is obtained by mixing the individual component prediction by their respective $\pi$ weights.

CoMix transparently shows which image patch causes which label prediction as visualized by the color in the last column. As an interesting side effect, the joint classification prediction reflects the count of objects in the image. For example, in the third row of the MultiMNIST data, the joint prediction probability of number 3 is twice the probability of number 9. Naturally, this is caused by the number 3 occurring twice in the image. Hence, our model learns to count without ever having received the supervision in this regard because the textual labels only indicate the presence of objects and not the number of them.

Being able to backtrack exactly how predictions are composed and caused by the concrete image regions is another strength of our model that makes it more interpretable. Due to the hard-attention mechanism of the spatial transformer module, CoMix only uses pixel information inside the bounding box to form its representations and the predictions. In contrast, while WSDDN also predicts bounding boxes, there is no exclusive relationship to the part of the picture the bounding box captures. The relationship is on a feature level that often has a receptive field covering the whole image.

### 6.1.5 Conclusion

We introduced our CoMix model that learns a compositional Gaussian mixture representation for both images and text. It utilizes the spatial-transformer architecture to decompose the raw image and expose transparently, which patches are responsible for the Gaussian components of the representation. The information content of the representation is guided by the textual data by employing a likelihood and KL-divergence loss. Without any additional supervision, CoMix learns to detect objects solely facilitated by an additional area loss.

We demonstrate the advantages of learning a compositional representation on the tasks of weakly supervised object detection and image retrieval, where we can validate that our model can generalize and extrapolate to an unseen combination of objects while, at the same time, being easier to inspect and interpret. Although our current results focus on synthetically generated datasets, a plausible next step would be to scale to the natural images with the natural-language captions, where the employment of recent language models such as BERT [Dev+19], GPT-3 [Bro+20] could enable richer textual representations. Taking further advantage from advances in contrastive learning such as in CLIP [Rad+21], our model could extend these approaches to be more interpretable and inherently exhibit compositionality in their representations.

## 6.2 Semantic Image Synthesis with Semantically Coupled VQ-Model

Semantic image synthesis enables control over unconditional image generation by allowing guidance on what is being generated. We conditionally synthesize the latent space from a vector quantized model (VQ-model) pre-trained to autoencode images. Instead of training an autoregressive Transformer on separately learned conditioning latents and image latents, we find that jointly learning the conditioning and image latents significantly improves the modeling capabilities of the Transformer model. While our jointly trained VQ-model achieves a similar reconstruction performance to a vanilla VQ-model for both semantic and image latents, tying the two modalities at the autoencoding stage proves to be an important ingredient to improve autoregressive modeling performance. We show that our model improves semantic image synthesis using autoregressive models on popular semantic image datasets ADE20k, Cityscapes and COCO-Stuff.


Figure 6.4: A semantically coupled VQ-model together with a Transformer generator synthesizes images that follows the semantic guidance closer and has higher fidelity. For instance, the semantically coupled model correctly reproduces the lamp next to the bed and more accurately matches the shape of the store fronts.

### 6.2.1 Introduction

Semantic image synthesis allows for precise specification of the semantic content of an image. This enables applications such as artistic image creation, e.g. by outlining the scene and components in novel ways, or data augmentation [SFS20], e.g. creating similar images or changing objects or styles. In this work, semantic information refers to the class identity of objects (e.g. person, car, dog, chair) and scene concepts (e.g. sky, road, grass, lake), but also their locations, size and shape in the image. Advances in generative models have led the progress in semantic image synthesis methods mostly through improvements to GAN-based models [Hon+18; Iso+17; Nta+20; Par+19] and autoregressive generative models [Che+20a; Chi+19; ROV19]. Vector-quantized models (VQ-model) such as the VQGAN model [ERO21] combine the benefits of both GANs and autoregressive training into a single model. By building upon the VQVAE [OVK17; ROV19], the addition of a discriminator results in high-fidelity image generations similar to other GAN-based models.

In this work we propose improvements to the architecture of VQ-models, such as the VQGAN, that allows more effective usage of semantic conditioning information. Our model incorporates semantic information already at the autoencoding stage, allowing the VQ-model to combine the two modalities before the autoregressive training stage. We train a Transformer model [Vas+17] to generate latents which can subsequently be synthesized by the decoder of the VQ-model.

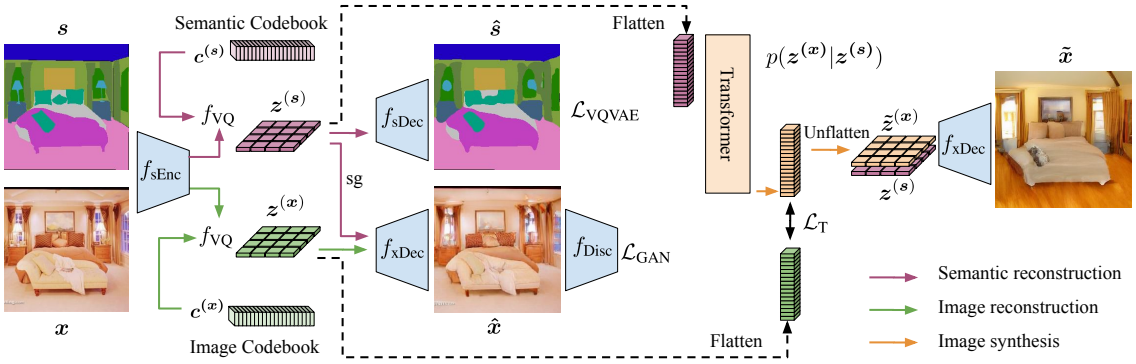



Figure 6.5: Overview of our semantically coupled VQ-model architecture. A single encoder produces both semantic and image latents. Two decoders reproduce semantics and image, while the image decoder also uses semantic information. The Transformer is trained to predict image latents conditioned on semantic latents and the image decoder synthesizes the latents.

To accomplish this, we make the following contributions. We propose an extension to the VQ-model auto-encoder that incorporates the semantic information used for image synthesis. By doing so, the decoder already learns the relationship between semantics and image content to make better image generations. An autoregressive Transformer model then only needs to act on the latent space of a single encoder model as opposed to requiring an auxiliary VQVAE which was proposed by [ERO21]. Our semantically coupled latents enable the Transformer model to create a better generative model of the data as seen in Figure 6.4, where semantic detail is better replicated in the synthesized images.

### 6.2.2 Model

We follow the methodology of [ERO21] to produce a generative model with semantic conditioning as a two-step approach. Instead of directly generating on the image space, we perform the task on the latent space. To do so, our image synthesis pipeline consists of two models parts: 1) an auto-encoder that learns a latent representation of the images; 2) an autoregressive model that learns to generate the latent code.

**Auto-Encoding with VQ-Models.** We consider both VQVAE [OVK17; ROV19] and VQGAN [ERO21] as the latent variable models and refer to them collectively as VQ-models. A VQ-model consists of an encoder $f_{\text{Enc}}$ that maps the input $x$ to a discrete latent space $z$ and a decoder $f_{\text{Dec}}$ reconstructing $x$ from $z$. The output of the encoder $f_{\text{Enc}}(x)$ is quantized to the closest vector of a learned codebook $\boldsymbol{c} \in \mathbb{R}^{K \times D}$ where $K$ is the number of codebook entries and $D$ the dimensionality. The quantization of the latent space allows using autoregressive generative models on shorter sequences than the full input data dimension. VQGAN sets itself apart from the VQVAE by introducing an additional discriminator CNN that is trained to distinguish between ground truth images and reconstructions from the decoder $f_{\text{Dec}}$.

**Autoregressive Modeling with Transformer.** After training the VQ-model on images $\boldsymbol{x}$ completes, an autoregressive model is typically trained to generate the image latents $\boldsymbol{z}^{(\boldsymbol{x})}$ of the training

| | Cityscapes | | | ADE20k | | | COCO-Stuff | | |
|---|---|---|---|---|---|---|---|---|---|
| | FID↓ | SSIM↑ | LPIPS↓ | FID↓ | SSIM↑ | LPIPS↓ | FID↓ | SSIM↑ | LPIPS↓ |
| VQVAE-T [OVK17; ROV19] | 190.22 | 0.3652 | 0.6406 | 142.11 | 0.0769 | 0.8043 | 111.85 | 0.0820 | 0.8127 |
| sVQVAE-T | 192.65 | **0.4000** | 0.6002 | 148.09 | **0.1343** | 0.7846 | 114.55 | **0.1382** | 0.7857 |
| VQGAN-T [ERO21] | **130.49** | 0.2797 | 0.4873 | 46.50 | 0.0667 | 0.6460 | 33.38 | 0.0638 | 0.6533 |
| sVQGAN-T | 131.37 | 0.3013 | **0.4034** | **38.36** | 0.0987 | **0.5534** | **28.80** | 0.0984 | **0.5583** |

Table 6.3: Semantic image synthesis results for VQ-Tansformer models on Cityscapes, ADE20K and COCO-Stuff datasets measuring SSIM, FID and LPIPS between generations and ground truth images with the same semantic map. (sVQVAE-T and sVQGAN-T uses $\lambda = 0.1$)

images, by maximizing the likelihood of the factorized joint distribution

$$p(\boldsymbol{z}^{(\boldsymbol{x})}) = p(z_1^{(x)}, ...z_n^{(x)}) = \prod_i^n p(z_i^{(x)}|z_1^{(x)}, \dots, z_{i-1}^{(x)}). \tag{6.18}$$

In order to perform semantic image synthesis, [ERO21] trains a separate VQVAE on auto-encoding the semantic map $\boldsymbol{s}$ to produce semantic latents $\boldsymbol{z}^{(\boldsymbol{s})}$. To condition the autoregressive model with this semantic information, we prepend the semantics latents $\boldsymbol{z}^{(\boldsymbol{s})}$ to the image latents $\boldsymbol{z}^{(\boldsymbol{x})}$ and train the autoregressive model on the conditional likelihood

$$p(\boldsymbol{z}^{(\boldsymbol{x})}|\boldsymbol{z}^{(\boldsymbol{s})}) = \prod_i^n p(z_i^{(x)}|z_1^{(x)}, \dots, z_{i-1}^{(x)}, \boldsymbol{z}^{(\boldsymbol{s})}) \tag{6.19}$$

which is done by minimizing the negative log-likelihood $\mathcal{L}_\text{T} = -\log p(\boldsymbol{z}^{(\boldsymbol{x})}|\boldsymbol{z}^{(\boldsymbol{s})})$. For the autoregressive model, we use a Transformer [Vas+17].

**Semantically Coupled VQ-Model.** We deem the existing approach of conditioning semantic information to the autoregressive Transformer suboptimal for several reasons. It requires training two independent VQ-models and the decoder only uses the image latents to produce the reconstruction, while more information in form of the semantic latents is available. This shifts the learning of correlations and dependencies between semantics and image entirely to the Transformer. We propose a semantically coupled VQ-model that incorporates the conditioning information already in the auto-encoding stage of a single VQ-model that jointly learns to reconstruct both images and semantics.

Figure 6.5 illustrates the joint model learning both latents at the same time. The encoder $f_\text{sEnc}(x, s)$ is a shared encoder that takes the concatenation of the image and semantic map as input to produce two latents, $\boldsymbol{z}^{(\boldsymbol{x})}$, and $\boldsymbol{z}^{(\boldsymbol{s})}$, respectively. The decoder is then split into two CNNs, one reconstructing the semantics using only the semantic latent $f_\text{sDec}(\boldsymbol{z}^{(\boldsymbol{s})}) = \hat{\boldsymbol{s}}$ and one reconstructing the image having access to both the semantic and the image latent $f_\text{xDec}(\boldsymbol{z}^{(\boldsymbol{x})}, \text{sg}\big[\boldsymbol{z}^{(\boldsymbol{s})}\big]) = \hat{\boldsymbol{x}}$. We stop the gradient flow from the image decoder to the semantic latent, such that each decoder is responsible for training exactly one of the latents, separating and focusing their training signal, while still allowing the image encoder to access the semantic latent and, thus, allowing it to encode complementary information in the image latents.


Figure 6.6: Qualitative results comparing multiple generations of an individually trained VQGAN-T with a semantically coupled sVQGAN-T on COCO-Stuff (top) and ADE20K (bottom). Semantic details are only retained by sVQGAN-T, e.g., bridge (top), airplane (bottom).

By restricting the gradient flow and using two decoders, we also induce the dependency structure of the two latents, i.e., the image latent depends on the semantic latent, but not vice versa, and the semantic latent is learned independently. Apart from this architectural change, the loss functions remain the same for both VQVAE and VQGAN. The semantic reconstruction is again trained with a cross-entropy term. Thus, the loss terms are combined into a single loss function

$$\mathcal{L}_{\text{sVQ}} = \mathcal{L}_{\text{VQ}}(\boldsymbol{x}, f_{\text{xDec}}(\boldsymbol{z}^{(\boldsymbol{x})}, \boldsymbol{z}^{(\boldsymbol{s})})) + \lambda \mathcal{L}_{\text{VQVAE}}(\boldsymbol{s}, f_{\text{sDec}}(\boldsymbol{z}^{(\boldsymbol{s})})) \tag{6.20}$$

where the second term comes from the VQVAE reconstructing the semantics and $\mathcal{L}_{\text{VQ}}$ concerns reconstructing the image and can be either $\mathcal{L}_{\text{VQGAN}}$ or $\mathcal{L}_{\text{VQVAE}}$. The hyperparameter $\lambda$ allows balancing the two loss terms.

After training the semantic VQ-model to auto-encode both images and semantics, we train the Transformer network with a cross-entropy loss to maximize the log-likelihood of Equation 6.19 where both the conditioning latents and the prediction latents come from the same VQ-model.

### 6.2.3 Experiments

**Experimental Setup.** To train and evaluate our models, we combine several semantic image datasets into one large dataset with dense semantic image annotations, namely Cityscapes [Cor+16], ADE20k [Zho+17] and COCO-Stuff [CUF18]. We create a unified semantic class mapping across the three datasets combining the 20, 150 and 183 object classes of Cityscapes, ADE20k and COCO-Stuff into a total of 243 classes by merging labels that occur across datasets such as person, car, building, etc. We evaluate our semantically coupled VQ-model in comparison to the traditional approach of training semantic latents and image latents separately. We use the VQVAE and VQGAN as base models and evaluate the VQ-models after the second stage when performing semantic image synthesis. The quality of image reconstructions and generations is evaluated using the Fréchet Inception Distance (FID) [Heu+17], the structural similarity index (SSIM) [Wan+04], and the Learned Perceptual Image Patch Similarity (LPIPS) [Zha+18].

**Semantic Image Synthesis using VQ-Transformer.** We find that semantically coupling the latents with our sVQVAE or sVQGAN model significantly improves the performance of the Transformer model in predicting the conditional sequence of latents. In Table 6.3, we observe that our semantically coupled VQ-model improves over the individually trained models across all datasets and metrics. For instance, the FID score of sVQGAN-T significantly improves over VQGAN-T with 38.4 vs. 46.5 on ADE20k and 28.8 vs. 33.4 on COCO-Stuff (lower is better). Thus, sVQGAN-T can better model the whole data distribution of the original datasets, which includes covering all semantic classes without distortions. For both VQVAE-T and VQGAN-T models, we find that our semantic variants achieve better SSIM and LPIPS scores, e.g., LPIPS of sVQVAE-T is 0.403 vs 0.487; 0.553 vs 0.646; 0.558 vs 0.653 (lower is better). These results indicate that our semantically coupled VQ-models better follow the semantic structure of the image as the semantic maps are the only source of information about the ground truth image which the metrics use for evaluation. We find that the improvements of our semantically coupled VQ-models stem from the complementary structure the semantic and image latents have learned during the auto-encoding training stage. In Figure 6.6, we illustrate synthesized images from the VQGAN-T and the sVQGAN-T model, sampling three times each. Some details of the semantic information is sometimes not properly replicated by the VQGAN-T model, e.g., the bridge in the first row or the plane in the second row. On the other hand, our sVQGAN-T model consistently generates these details provided by the semantics. These results show that training the latents of the two modalities together create stronger dependencies between them, which the Transformer model can leverage.

### 6.2.4 Conclusion

In this work, we present semantically coupled VQ-models that jointly learn latents of two modalities, images and semantic maps. We have shown that coupling the latents during training leads to dependencies that are easier to pick up by the Transformer model used to model their conditional distribution. For both VQVAE and VQGAN as the VQ-model, the semantic coupling improves the synthesis of images especially in following details of the semantic maps that is being conditioned on. Further investigation into understanding the cause of our findings could allow designing latent variable models with better synergies across data modalities beneficial for autoregressive modeling of Transformers. Currently, a reference image is required during inference as input to the VQ-Model. Further work will try to alleviate this dependency completely.

# 7

# DISCUSSION AND CONCLUSION

This thesis deals with the problem setting of explainable AI in the computer vision domain. The purpose of our work is to expose an explanation about why and how our deep models made a certain prediction for a given task. To achieve this goal, we propose means of communication to enrich existing problem settings and facilitate interpretation of the deep models and their results.

In the previous chapters, a novel application of communication and language in the field of deep learning and explainable AI was established. To summarize, we

- exposed interpretable information of neural networks, thus, allowing introspection into their internal structure or intermediate representations,
- demonstrated how communication tasks help shaping and evaluating explainable systems,
- provided evidence why incorporating human-understandable semantics is important.

The following section reviews each contribution individually and collectively, discusses their strong sides as well as takes a look at their current limitations, proposing how the drawbacks could be overcome in the future.

## 7.1 Discussion of Results

**Learning Decision Trees Recurrently Through Communication.** We present RDTC, a framework to solve image-classification tasks by two communicating neural agents, in Chapter 3. The AbL agent holds an image while the RDT agent has to query the AbL agent, to obtain information about the image and make the classification.

In a standard CNN setup, the network would produce a prediction in a single step. Instead, our communication imposes a bottleneck that regulates the information flow and allows us to analyze and interpret each step of the question-answer loop. We specifically choose to make responses binary: asking yes/no questions is a common way for humans to learn new information and it is the smallest discrete unit of information, which makes explanation as intuitive as possible. Our goal is to transfer this idea to how our RDTC model presents its explanations to a human user. Our proposed framework integrates components for training the two agents end-to-end by making

the discrete communication differentiable with Gumbel-Softmax. Consequently, we can jointly optimize a classification loss and learn to effectively communicate between the two agents. This leads to two types of introspections into our model:

- each image is classified by following a sequence of binary decisions, where each question-answer pair contributes incrementally to the class prediction,
- our model learns to arrange questions globally, building a decision tree.

These contributions align with our goal of developing more interpretable model components for explainable AI. However, we find that human understanding is challenging when binary features, i.e., vocabulary, are discovered from scratch and are unrelated to semantic concepts known to the human in advance. We, therefore, incorporate a loss for aligning the binary features with a set of human-understandable attributes about the images. At this point, the questions asked by the RDT agent refer to clear concepts about the object in the image, e.g., "Does it have whiskers?". Our user study provides evidence that the combination of both using interpretable concepts in the communication and building decision trees improves the explainability of our RDTC model. Furthermore, with examples, we demonstrate, where we can pinpoint the exact binary decision leading to an error in the prediction. This enables model debugging and helps establishing trust for cases, when explanations are coherent with the output.

While we generally expect a trade-off between an explainable and a non-explainable model because we impose additional constraints through the communication, in the case of RDTC, we can retain the classification performance compared to a base model without our communication framework. This could be explained by the fact that we are still using a complex CNN backbone that can learn structured binary features but is ultimately not interpretable. Additionally, human-understandable attributes allow separating all the classes in AWA2 and CUB. Doing so is a requirement for achieving the same performance when they are used as an explainable vocabulary in the communication.

**Modeling Conceptual Understanding in Image Reference Games.** In Chapter 4, we investigate how differences of the conceptual understanding of the world affect interactions between agents when they communicate with one another.

Humans have the ability to tailor their explanations to the variations of the receiver's conceptual understanding over task-related concepts. For instance, different words have to be chosen when providing an explanation to a child than to an adult. Similarly, an expert terminology could make communication efficient given a suitable context and a communication partner, who understands its meaning.

With this work our goal is to have artificial agents obtain a similar ability of adjusting their communication behavior depending on how well the receiver understands the presented terms. Since the internal states of different actors in the environment are generally not available, the conceptual understanding of others needs to be inferred by observing their behavior. To study this idea, we formulate an image reference game between two agents, where modeling the conceptual understanding of other agents is key to performing well on the task.

Given a set of two images, the speaker agent communicates the presence of an attribute to the listener, who, in return, tries to determine, which image the speaker wants to identify. The listener, however, might not understand certain attributes and can only solve the task when the understood concepts are communicated. The goal of the speaker is to infer, which attributes are understood, by repeatedly playing the game and form a "mental model" of its communication partner to maximize reward. The proposed task contributes to better evaluation of the deep learning systems in more challenging environments, which is especially important to explainable AI, where communication of concepts is involved. We further develop a speaker-agent model that learns to quickly adapt to individual listeners based on the past experience, to avoid using misunderstood attributes in the future. After training, our agent is able to expose its inferred states for the listener agents, i.e., it provides details about which concepts it believes are understood and which are not. Our experiments illustrate that the learned agent embeddings can be used to cluster listeners based on their similarities in understanding concepts, e.g., if several agents misunderstand the same concepts because they are all color blind, these concepts would be clustered in the corresponding embedding space. Being able to expose this additional information allows us to further inspect, what the speaker agent has learned. This could potentially be used to discover biases in either the model or the data.

While these are promising results, we also notice that strong differences in the perceptual encoding of images, i.e., when we use different CNNs for the speaker and listener, degrade the effectiveness of our model. In such cases, the observation of other agents performing the presented task might not provide enough information to infer more complex differences in conceptual understanding.

**Abstracting Sketches through Simple Primitives.** In Chapter 5, we shift the focus from images to the sketch domain, to study abstractions through simple primitives and how they can benefit such scenarios, where communication of semantic concepts needs to be quick, i.e., when there is a limited budget on communicating the information. While sketches are already simplified representations of real-world concepts, the nuances of human drawing still need to be understood by a neural network. These nuances also make it difficult to computationally relate different sketches in an interpretable manner.

We develop PMN, a neural network that matches human strokes with a set of predefined primitives representing common shapes such as circle, line, arc, rectangle, in a self-supervised manner. By doing this, our model derives two key insights:

- Firstly, by transferring the raw sketch data into a sequence of primitive tokens, we are able to communicate the semantic content of the sketch more efficiently. This becomes possible due to the compression obtained from the translation.

- Secondly, the sketch representations become more interpretable as now all the sketches are made up from the same set of primitives and inter-sketch and inter-class relations can be inspected. Since the new representation still lies in the input space, it can be visualized just as any sketch, further contributing to its interpretability.

We evaluate PMN on a downstream task of sketch classification and sketch-based image retrieval through a communication bottleneck that defines a maximum information budget. This way, we can measure the effectiveness of both the primitive vocabulary and the compression achieved by our primitive encoding.

Our results demonstrate much higher task performance on low budget compared to communicating the original sketch data. Our result indicates that abstracting the data can lead to both increased explainability and improved task performance, when communication with a limited channel capacity is desired or is a requirement. Moreover, primitives let us relate different classes to one another in a self-supervised way, e.g., circles are used for heads of humans, cats, pigs, and owls, while arcs often represent the ears of different animals.

At the moment, we use a pre-defined and fixed vocabulary of primitives. This restricts the discovery of patterns that are not covered by the primitives and represents a trade-off, when it comes to interpretable compression and fine-grained details as our abstraction is not lossless. When the budget constraint is removed, using all the details from the raw data achieves better downstream performance. While we find that our selection covers most human strokes, an automated way of learning primitives could ensure a greater coverage of the input data alleviating the information loss. Nonetheless, with the goal of explainability in mind, it is generally preferred to have a small set of primitives that are distinct enough to ease making associations between sketches. This work contributes both in terms of creating more interpretable representations with PMN as part of its model design and with building upon the communication task, to evaluate this property.

**Compositional Mixture Representations for Vision and Text.** We develop an interpretable representation-learning approach called CoMix in Chapter 6.1.

Learning a representation of the data allows us to view it from a different perspective. We can make an attempt to regularize this representation, for the model to become more explainable. CoMix achieves this by building a compositional latent space that explicitly separates concepts from the input space, e.g., objects in an image, to different parts of the embedding. We do this by modeling the latent as a Gaussian mixture distribution, where each mixture component corresponds to a different region in the image. The region in the image is transparently exposed through the use of a spatial-transformer module isolating an enclosed part of the image for encoding. To further improve the explainability of CoMix, the latent space of images is aligned with a latent space from textual labels. Through this multi-modal alignment, we impose the compositionality of language onto the vision domain, i.e., images. Equally, a weak supervision of the text acts as a transfer of the semantic information such that CoMix can locate objects by their label and location in images. We show these properties on the weakly supervised object-detection task as well as on image retrieval given an unordered set of text labels, for which we want to find a matching image. The compositionality in the latent space allows CoMix to generalize to an unseen combination of labels at test time. Being a desirable property directly exposed in natural language, i.e., composing two words to form new phrases, this is hard to achieve in computer vision: in this domain, concepts are not always clearly separated.

While we do not employ communication directly in this work, compositionality and multi-modal alignment are important aspects for a communication model, when expressing visual concepts in a language. To bridge this gap further, the textual data could be extended to be from natural language such that the model creates associations between objects and all their synonyms.

**Semantic Image Synthesis with Semantically Coupled VQ-Model.** In Chapter 6.2, we explore vector-quantized (VQ) models in the context of semantic image synthesis.

VQ-models learn a discrete latent space for images, where each image is encoded to a grid of latent codes. Each individual latent code comes from a learned codebook that is similar to vocabularies in the language domain. Due to discretization, images can be treated as a sequence of visual tokens such that learning a generative model of images corresponds to learning a sequence model of the image tokens.

The vocabulary is learned from scratch; therefore, it is not easily interpretable as there is no direct correspondence to semantic concepts for each token. Nonetheless, each token could potentially encode unique image features. We examine how semantic conditioning influences the generative model of visual tokens. In the context of VQ-models, conditioning information in the form of semantic maps is first also encoded into discrete tokens and then prepended to the sequence the generative model is tasked to produce.

Our contribution revolves around a tighter integration between semantic and image token by encoding them jointly with a single encoder and using the semantic tokens as additional information while decoding the image. Our experiments demonstrate that with such a semantic coupling, generative image follows the semantic guidance better, i.e., objects are generated more consistently, where the semantic mask indicates them to be.

While this work mostly focuses on the task performance for image synthesis, the results suggest visual tokens correspond to the semantic information better through our semantic coupling. Our contribution underlines the importance of integrating semantic information into the model's architecture, and we could potentially use these observations to learn if visual tokens have the same properties as language tokens for them to be used to communicate information or explanations about the content of images.

## 7.2 Conclusion and Future Directions

Research on computer vision and deep learning can be characterized with an extraordinary progress: the pace at which new methods and disciplines are established continues to grow. Alongside more sophisticated models, arises the necessity for building more explainable models as well. With more explainability, these novel technologies would find a faster and easier way into everyday life and encourage the research community to better understand, learn from, and be more interested in further advancements in the deep learning field in general.

The goal of this thesis is to contribute to the research in the field of explainable AI by proposing means of communication to enhance the explainability of deep models. With our RDTC, PMN, and CoMix models, we present new building blocks and learning objectives that make parts of the

neural network inherently interpretable and that can be included into existing model architectures for tackling common tasks in the field.

One aspiration is to have deep models that are inherently interpretable throughout, from input to output. RDTC and CoMix use standard CNN backbones that by themselves are not interpretable. At the same time, their contributions are orthogonal to the research on post-hoc explanations that aim at explaining deep models after the fact. One could analyze the non-interpretable part of our models with a perturbation-based attribution method such as RISE [PDS18] or obtain input saliency maps with activation or gradient-based approaches like GradCAM [Sel+17]. In RDTC, these post-hoc explanations would allow to inspect whether the learned attributes are grounded in the correct regions of the image, further establishing trust in case of coherence, and enabling informed decisions in conjunction with humans in case of misalignment.

The recently proposed dynamic alignment networks [BFS21; BFS22] are a promising direction towards inherently interpretable neural networks. The model's explanations guarantee faithfulness, making them a great choice as a backbone. Yet, we can observe a trade-off between model's interpretability and a task performance. Similarly, we discussed that the lossy compression of our PMN greatly benefits explainability but can lead to a trade-off with achieving state-of-the-art scores in specific scenarios. While the current conclusion is that inherently interpretable models still come with a trade-off at least in some situations, it is an interesting research direction, to attempt to close this gap.

To better analyze cases where explainability is of interest, it is important to come up with new tasks reflecting real-world situations. Determining evaluation criteria is equally important as they would help explainable models' advancement. We propose a communication task in the context of an image-reference game, to study the influence of conceptual understanding on communicating explanations. This represents a specific use case for explanations and helps covering a width of desirable properties when developing new explainable AI systems.

The research community has established sanity checks for saliency maps [Ade+18; YG21] to uncover failure cases of post-hoc methods. At the same time, the evaluation of explainability remains a challenge.

For RDTC, we conducted a user study to measure the effectiveness of our explanation and observed positive results. User studies can be advantageous for measuring partially subjective effects of explanations but are far from being perfect. We currently lack a unified methodology and designing user studies to assess the task-specific influence of an explanation is non-trivial. In our communication setup, we tried to develop multi-agent scenarios, where the interaction between neural agents could act as a surrogate target for quantifying explainability, e.g., when our sketch primitives convey more semantic information for a downstream task. Nonetheless, the ultimate effect on a human can only be studied together with a human, so, additional efforts are required to find a fitting way to evaluate explainable AI.

Incorporating semantics from a natural language guarantees to some extent that explanations align with human-understandable concepts. Some related work takes the approach to generate full natural language explanations [Her+22; KT20; LZY20; Maj+21]. Natural language models such as GPT-3 [Bro+20] and the recently presented PaLM [Cho+22] have become so powerful that the

latter can explain complex jokes simply by prepending "Explain this joke:" to a natural-language prompt. Despite that, they cannot be used to make internal representations and structure of a neural network more explainable. For this purpose, we propose to align internal aspects of the models with semantic side information, to make model parts of the neural networks more explainable, e.g., when we use attributes as the vocabulary in the RDTC communication or when CoMix learns mixture representations, where image and text latents are located in the same space. With a similar idea, representation-learning approaches have been proposed to learn a shared feature space between image and language features using a contrastive loss [Huo+21; Jia+21; Li+21; She+21] building upon CLIP [Rad+21].

These approaches currently lack desirable properties of CoMix such as the compositionality of the latent features and the granularity in grounding latent components in the image. A logical next step would be to combine these efforts and integrate more general language models into the architectures from this thesis. In CoMix, this could be done at the encoding stage of the text data while at the same time scaling to both natural images and language pairs.

One clear advantage of a natural language is that the textual data is available in abundance. When we align the discrete vocabulary of our communication models with human-understandable words, human annotations are required as supervised information. Therefore, CoMix and RDTC currently rely on the availability of such a human-annotated data, so that we can train them with explainability in mind. Related work has explored the emergence of language in communication [Cao+18; CLF18; Evt+18; Far+22; Foe+16; HT17; Kot+17; LB20; Laz+18; LPT20], but it remains non-trivial to connect a learned neural language with a natural language without human intervention. RDTC can learn an emergent language with a vocabulary corresponding to binary features relating to input images. These features clearly correspond to visual concepts but naming them is challenging to do computationally.

Analogously, our semantic VQ-models learn visual tokens in an unsupervised task but cannot expose the meaning of each token in their vocabulary. In the vision domain, the discovery of visual concepts and subsequent association with semantic words is arguably more difficult than in more structured environments studied in reinforcement learning. Therefore, we believe the sketch domain could be a promising starting point to research this direction further.

Instead of pre-defining sketch primitives for our PMN, the shape of primitives could be discovered end-to-end with our reconstruction objective. Since our primitives lie in the input space, even learned ones can be visually inspected and remain interpretable. This contrasts with the situation when the communication vocabulary is used at an intermediate layer of the model such as for RDTC and sVQ. With the sketch domain as a test bed, it would be interesting to see whether such an idea could be ultimately transferred to natural images, where visual tokens lie in the input space and compose images when arranged together to communicate the semantic content of a scene.

# BIBLIOGRAPHY


[Ade+18] J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim. "Sanity Checks for Saliency Maps". In: *NeurIPS*. 2018 (cit. on p. 80).

[Aga+21] S. Agarwal, O. Iqbal, S. A. Buridi, M. Manjusha, and A. Das. "Reinforcement Explanation Learning". In: *NeurIPS*. 2021 (cit. on p. 11).

[Aka+13] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. "Label-embedding for attribute-based classification". In: *CVPR*. 2013 (cit. on p. 34).

[AFA22] S. Alaniz, M. Federici, and Z. Akata. "Compositional Mixture Representations for Vision and Text". In: *CVPR L3DIVU Workshop*. 2022 (cit. on p. 6).

[AHA22] S. Alaniz, T. Hummel, and Z. Akata. "Semantic Image Synthesis with Semantically Coupled VQ-Model". In: *ICLR DGM4HSD Workshop*. 2022 (cit. on p. 6).

[Ala+22] S. Alaniz, M. Mancini, D. Marcos, A. Dutta, and Z. Akata. "Abstracting Sketches through Simple Primitives". In: *ECCV*. 2022 (cit. on p. 6).

[Ala+21] S. Alaniz, D. Marcos, B. Schiele, and Z. Akata. "Learning Decision Trees Recurrently Through Communication". In: *CVPR*. 2021 (cit. on p. 5).

[Arr+20] A. B. Arrieta, N. D. Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Inf. Fusion* 58 (2020) (cit. on p. 2).

[AJK19] A. Arun, C. V. Jawahar, and M. P. Kumar. "Dissimilarity Coefficient Based Weakly Supervised Object Detection". In: *CVPR*. 2019 (cit. on p. 58).

[AE13] A. T. Azar and S. M. El-Metwally. "Decision tree classifiers for automated medical diagnosis". In: *Neural Computing and Applications* 23.7-8 (2013) (cit. on p. 16).

[Bac+15] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation". In: *PLoS ONE*. 2015 (cit. on p. 11).

[BST11] C. Baker, R. Saxe, and J. Tenenbaum. "Bayesian theory of mind: Modeling joint belief-desire attribution". In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. Vol. 33. 2011 (cit. on p. 30).

[Bak+17] C. L. Baker, J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum. "Rational quantitative attribution of beliefs, desires and percepts in human mentalizing". In: *Nature Human Behaviour* 1 (2017) (cit. on p. 30).

[BL99] B. Barquero and R. Logie. "Imagery constraints on quantitative and qualitative aspects of mental synthesis". In: *European Journal of Cognitive Psychology* 11.3 (1999) (cit. on p. 42).

[Ben+92] S. Bengio, Y. Bengio, J. Cloutier, and J. Gecsei. "On the optimization of a synaptic learning rule". In: *Preprints Conf. Optimality in Artificial and Biological Neural Networks*. 1992 (cit. on p. 31).

[Ben17] Y. Bengio. "The Consciousness Prior". In: *CoRR* abs/1709.08568 (2017) (cit. on pp. 18, 56).

[Ber+13] I. Berger, A. Shamir, M. Mahler, E. Carter, and J. Hodgins. "Style and abstraction in portrait sketching". In: *ACM Transactions on Graphics (TOG)* 32.4 (2013) (cit. on p. 43).

[Bhu+21] A. K. Bhunia, P. N. Chowdhury, A. Sain, Y. Yang, T. Xiang, and Y.-Z. Song. "More Photos Are All You Need: Semi-Supervised Learning for Fine-Grained Sketch Based Image Retrieval". In: *CVPR*. 2021 (cit. on p. 43).

[Bhu+20] A. K. Bhunia, Y. Yang, T. M. Hospedales, T. Xiang, and Y. Song. "Sketch Less for More: On-the-Fly Fine-Grained Sketch-Based Image Retrieval". In: *CVPR*. 2020 (cit. on pp. 44, 48, 111, 114, 115).

[Bie87] I. Biederman. "Recognition-by-components: a theory of human image understanding." In: *Psychological review* 94.2 (1987) (cit. on p. 44).

[BV16] H. Bilen and A. Vedaldi. "Weakly Supervised Deep Detection Networks". In: *CVPR*. 2016 (cit. on pp. 58, 63).

[BFS21] M. Böhle, M. Fritz, and B. Schiele. "Convolutional Dynamic Alignment Networks for Interpretable Classifications". In: *CVPR*. 2021 (cit. on pp. 11, 80).

[BFS22] M. Böhle, M. Fritz, and B. Schiele. "B-cos Networks: Alignment is All We Need for Interpretability". In: *CVPR*. 2022 (cit. on pp. 11, 80).

[Bre+84] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984 (cit. on p. 23).

[BB19] W. Brendel and M. Bethge. "Approximating CNNs with Bag-of-local-Features models works surprisingly well on ImageNet". In: *ICLR*. 2019 (cit. on p. 11).

[Bro+20] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. "Language Models are Few-Shot Learners". In: *NeurIPS*. 2020 (cit. on pp. 10, 68, 80).

[BD20] C. A. Brust and J. Denzler. "Integrating Domain Knowledge: Using Hierarchies to Improve Deep Classifiers". In: *Pattern Recognition*. Springer, 2020 (cit. on p. 17).

[Bur+19] C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner. "MONet: Unsupervised Scene Decomposition and Representation". In: *CoRR* abs/1901.11390 (2019) (cit. on p. 57).

[But+09] J. Butterfield, O. C. Jenkins, D. M. Sobel, and J. Schwertfeger. "Modeling aspects of theory of mind with Markov random fields". In: *International Journal of Social Robotics* 1 (2009) (cit. on p. 30).

[CUF18] H. Caesar, J. Uijlings, and V. Ferrari. "Coco-stuff: Thing and stuff classes in context". In: *CVPR*. 2018 (cit. on p. 72).

[Cao+18] K. Cao, A. Lazaridou, M. Lanctot, J. Z. Leibo, K. Tuyls, and S. Clark. "Emergent Communication through Negotiation". In: *ICLR*. 2018 (cit. on pp. 12, 17, 81).

[Cao+11] Y. Cao, C. Wang, L. Zhang, and L. Zhang. "Edgel index for large-scale sketch-based image search". In: *CVPR*. 2011 (cit. on p. 44).

[Cao+10] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, and L. Zhang. "Mindfinder: interactive sketch-based image search on millions of images". In: *ACM MM*. 2010 (cit. on p. 44).

[Cha+17] A. Chandrasekaran, D. Yadav, P. Chattopadhyay, V. Prabhu, and D. Parikh. "It Takes Two to Tango: Towards Theory of AI's Mind". In: *CoRR* abs/1704.00717 (2017) (cit. on p. 29).

[Che+19a] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. Su. "This Looks Like That: Deep Learning for Interpretable Image Recognition". In: *NeurIPS*. 2019 (cit. on p. 11).

[CGG13] H. Chen, A. Gallagher, and B. Girod. "What's in a Name: First Names as Facial Attributes". In: *CVPR*. 2013 (cit. on p. 17).

[CGG12] H. Chen, A. Gallagher, and B. Girod. "Describing Clothing by Semantic Attributes". In: *ECCV*. 2012 (cit. on p. 17).

[Che+20a] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever. "Generative Pretraining From Pixels". In: *ICML*. 2020 (cit. on p. 69).

[Che+19b] R. Chen, H. Chen, G. Huang, J. Ren, and Q. Zhang. "Explaining Neural Networks Semantically and Quantitatively". In: *ICCV*. 2019 (cit. on p. 17).

[Che+16] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets". In: *NeurIPS*. 2016 (cit. on p. 55).

[Che+20b] Y. Chen, L. Li, L. Yu, A. E. Kholy, F. Ahmed, Z. Gan, Y. Cheng, and J. Liu. "UNITER: UNiversal Image-TExt Representation Learning". In: *ECCV*. 2020 (cit. on p. 58).

[Chi+19] R. Child, S. Gray, A. Radford, and I. Sutskever. "Generating Long Sequences with Sparse Transformers". In: *CoRR* abs/1904.10509 (2019) (cit. on p. 69).

[CLF18] E. Choi, A. Lazaridou, and N. de Freitas. "Compositional Obverter Communication Learning from Raw Visual Input". In: *ICLR*. 2018 (cit. on pp. 31, 81).

[Cho+22] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. "PaLM: Scaling Language Modeling with Pathways". In: *CoRR* abs/2204.02311 (2022) (cit. on pp. 10, 80).

[Cog+19] M. Cogswell, J. Lu, S. Lee, D. Parikh, and D. Batra. "Emergence of Compositional Language with Deep Generational Transmission". In: *CoRR* abs/1904.09067 (2019) (cit. on p. 31).

[Cor+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *(CVPR)*. 2016 (cit. on p. 72).

[CAA19] R. Corona, S. Alaniz, and Z. Akata. "Modeling Conceptual Understanding in Image Reference Games". In: *NeurIPS*. 2019 (cit. on pp. 6, 17).

[CDK19] A. Cortese, B. De Martino, and M. Kawato. "The neural and cognitive architecture for learning from a small sample". In: *Current opinion in neurobiology* 55 (2019) (cit. on p. 18).

[Cra+20] M. D. Cranmer, A. Sanchez-Gonzalez, P. W. Battaglia, R. Xu, K. Cranmer, D. N. Spergel, and S. Ho. "Discovering Symbolic Models from Deep Learning with Inductive Biases". In: *NeurIPS*. 2020 (cit. on p. 18).

[CP19] E. Crawford and J. Pineau. "Spatially Invariant Unsupervised Object Detection with Convolutional Neural Networks". In: *AAAI*. 2019 (cit. on p. 57).

[Das+19] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau. "TarMAC: Targeted Multi-Agent Communication". In: *ICML*. 2019 (cit. on p. 17).

[Das+17] A. Das, S. Kottur, J. M. Moura, S. Lee, and D. Batra. "Learning cooperative visual dialog agents with deep reinforcement learning". In: *ICCV*. 2017 (cit. on p. 31).

[Dee22] DeepMind. *AlphaGo*. 2022. URL: https://www.deepmind.com/research/highlighted-research/alphago (visited on 2022-03-30) (cit. on p. 1).

[Den+22] H. Deng, Q. Ren, X. Chen, H. Zhang, J. Ren, and Q. Zhang. "Discovering and Explaining the Representation Bottleneck of DNNs". In: *ICLR*. 2022 (cit. on p. 11).

[Den+09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *CVPR*. 2009 (cit. on p. 34).

[Dev+19] J. Devlin, M. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *NAACL-HLT*. 2019 (cit. on pp. 56, 58, 68).

[DK17] F. Doshi-Velez and B. Kim. "Towards a rigorous science of interpretable machine learning". In: *CoRR:1702.08608* (2017) (cit. on p. 17).

[Dos+21] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR*. 2021 (cit. on p. 10).

[Eit+10] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. "Sketch-based image retrieval: Benchmark and bag-of-features descriptors". In: *IEEE transactions on visualization and computer graphics* 17.11 (2010) (cit. on p. 43).

[Esl+16] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. "Attend, Infer, Repeat: Fast Scene Understanding with Generative Models". In: *NeurIPS*. 2016 (cit. on p. 57).

[ERO21] P. Esser, R. Rombach, and B. Ommer. "Taming Transformers for High-Resolution Image Synthesis". In: *CVPR*. 2021 (cit. on pp. 69–71).

[Eve+] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results* (cit. on p. 63).

[Evt+18] K. Evtimova, A. Drozdov, D. Kiela, and K. Cho. "Emergent Communication in a Multi-Modal, Multi-Step Referential Game". In: *ICLR*. 2018 (cit. on pp. 31, 81).

[Far+22] A. Farinhas, W. Aziz, V. Niculae, and A. F. T. Martins. "Sparse Communication via Mixed Distributions". In: *ICLR*. 2022 (cit. on pp. 12, 81).

[Fed+20] M. Federici, A. Dutta, P. Forré, N. Kushman, and Z. Akata. "Learning Robust Representations via Multi-View Information Bottleneck". In: *ICLR*. 2020 (cit. on p. 44).

[FZ08] V. Ferrari and A. Zisserman. “Learning Visual Attributes”. In: *NeurIPS*. 2008 (cit. on p. 17).

[FS88] R. A. Finke and K. Slayton. “Explorations of creative visual synthesis in mental imagery”. In: *Memory & cognition* 16.3 (1988) (cit. on pp. 41, 42).

[FAL17] C. Finn, P. Abbeel, and S. Levine. “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *ICML*. 2017 (cit. on p. 31).

[Foe+16] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson. “Learning to Communicate with Deep Multi-Agent Reinforcement Learning”. In: *NeurIPS*. 2016 (cit. on pp. 12, 15, 17, 31, 81).

[Fre14] A. A. Freitas. “Comprehensible classification models: a position paper”. In: *ACM SIGKDD Explorations Newsletter* 15 (2014) (cit. on p. 29).

[FB97] M. A. Friedl and C. E. Brodley. “Decision tree classification of land cover from remotely sensed data”. In: *Remote Sensing of Environment* 61.3 (1997) (cit. on p. 16).

[FH17] N. Frosst and G. Hinton. “Distilling a neural network into a soft decision tree”. In: *CoRR:1711.09784* (2017) (cit. on p. 17).

[Gan+21] Y. Ganin, S. Bartunov, Y. Li, E. Keller, and S. Saliceti. “Computer-aided design as language”. In: *NeurIPS* (2021) (cit. on p. 44).

[Gao+19] Y. Gao, B. Liu, N. Guo, X. Ye, F. Wan, H. You, and D. Fan. “C-MIDN: Coupled Multiple Instance Detection Network With Segmentation Guidance for Weakly Supervised Object Detection”. In: *ICCV*. 2019 (cit. on p. 58).

[Gil+18] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. “Explaining Explanations: An Overview of Interpretability of Machine Learning”. In: *DSAA*. 2018 (cit. on p. 29).

[Goo+15] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. “Generative Adversarial Networks”. In: *NIPS*. 2015 (cit. on p. 10).

[Gre+19] K. Greff, R. L. Kaufman, R. Kabra, N. Watters, C. Burgess, D. Zoran, L. Matthey, M. Botvinick, and A. Lerchner. “Multi-Object Representation Learning with Iterative Variational Inference”. In: *ICML*. 2019 (cit. on p. 57).

[GA19] D. Gunning and D. W. Aha. “DARPA’s Explainable Artificial Intelligence Program”. In: *AI Magazine* 40.2 (2019) (cit. on p. 15).

[HE18] D. Ha and D. Eck. “A Neural Representation of Sketch Drawings”. In: *ICLR*. 2018 (cit. on p. 47).

[HP11] T. Hammond and B. Paulson. “Recognizing sketched multistroke primitives”. In: *ACM Transactions on Interactive Intelligent Systems (TiiS)* 1.1 (2011) (cit. on p. 44).

[Han+00] M. C. Hansen, R. S. DeFries, J. R. Townshend, and R. Sohlberg. “Global land cover classification at 1 km spatial resolution using a classification tree approach”. In: *International Journal of Remote Sensing* 21.6-7 (2000) (cit. on p. 16).

[HT17] S. Havrylov and I. Titov. “Emergence of Language with Multi-agent Games: Learning to Communicate with Sequences of Symbols”. In: *NeurIPS*. 2017 (cit. on pp. 12, 15, 17, 31, 81).

[He+17] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. “Mask R-CNN”. In: *ICCV*. 2017 (cit. on pp. 10, 58).

[He+16] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *CVPR*. 2016 (cit. on pp. 10, 22, 23, 34).

[Hen+16] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. “Generating Visual Explanations”. In: *ECCV*. 2016 (cit. on pp. 11, 29).

[Hen+18] L. A. Hendricks, R. Hu, T. Darrell, and Z. Akata. “Grounding Visual Explanations”. In: *ECCV*. 2018 (cit. on pp. 11, 28).

[HG16] D. Hendrycks and K. Gimpel. “Gaussian Error Linear Units (GELUs)”. In: *CoRR* abs/1606.08415 (2016) (cit. on p. 111).

[Her+22] E. Hernandez, S. Schwettmann, D. Bau, T. Bagashvili, A. Torralba, and J. Andreas. “Natural Language Descriptions of Deep Features”. In: *ICLR*. 2022 (cit. on p. 80).

[Heu+17] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *NeurIPS*. 2017 (cit. on p. 72).

[HVD15] G. Hinton, O. Vinyals, and J. Dean. “Distilling the knowledge in a neural network”. In: *CoRR:1503.02531* (2015) (cit. on pp. 17, 19).

[HS97] S. Hochreiter and J. Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997) (cit. on pp. 10, 19, 33).

[Hon+18] S. Hong, X. Yan, T. S. Huang, and H. Lee. “Learning hierarchical semantic image manipulation through structured representations”. In: *NeurIPS*. 2018 (cit. on p. 69).

[HSW89] K. Hornik, M. B. Stinchcombe, and H. White. “Multilayer feedforward networks are universal approximators”. In: *Neural Networks* 2 (1989) (cit. on p. 11).

[Huk+18] D. Huk Park, L. Anne Hendricks, Z. Akata, A. Rohrbach, B. Schiele, T. Darrell, and M. Rohrbach. “Multimodal explanations: Justifying decisions and pointing to the evidence”. In: *CVPR*. 2018 (cit. on p. 29).

[HPM19] K. D. Humbird, J. L. Peterson, and R. G. Mcclarren. “Deep Neural Network Initialization With Decision Trees”. In: *IEEE Trans. Neural Netw. Learn. Syst* 30.5 (2019) (cit. on p. 17).

[Huo+21] Y. Huo, M. Zhang, G. Liu, H. Lu, Y. Gao, G. Yang, J. Wen, H. Zhang, B. Xu, W. Zheng, Z. Xi, Y. Yang, A. Hu, J. Zhao, R. Li, Y. Zhao, L. Zhang, Y. Song, X. Hong, W. Cui, D. Y. Hou, Y. Li, J. Li, P. Liu, Z. Gong, C. Jin, Y. Sun, S. Chen, Z. Lu, Z. Dou, Q. Jin, Y. Lan, W. X. Zhao, R. Song, and J. Wen. "WenLan: Bridging Vision and Language by Large-Scale Multi-Modal Pre-Training". In: *CoRR* abs/2103.06561 (2021) (cit. on pp. 58, 81).

[Huy+11] J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens. "An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models". In: *Decision Support Systems* 51.1 (2011), pp. 141–154 (cit. on p. 16).

[Iso+17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. "Image-to-Image Translation with Conditional Adversarial Networks". In: *CVPR* (2017) (cit. on p. 69).

[Jad+15] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. "Spatial Transformer Networks". In: *NeurIPS*. 2015 (cit. on pp. 57, 59).

[JGP17] E. Jang, S. Gu, and B. Poole. "Categorical Reparameterization with Gumbel-Softmax". In: *ICLR*. 2017 (cit. on p. 19).

[Jia+21] C. Jia, Y. Yang, Y. Xia, Y. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig. "Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision". In: *ICML*. 2021 (cit. on pp. 58, 81).

[Jia+20] Q. Jia, X. Fan, M. Yu, Y. Liu, D. Wang, and L. J. Latecki. "Coupling Deep Textural and Shape Features for Sketch Recognition". In: *ACM MM*. 2020 (cit. on p. 43).

[JL18] J. Jiang and Z. Lu. "Learning Attentional Communication for Multi-Agent Cooperation". In: *NeurIPS*. 2018 (cit. on p. 17).

[Jon+16] J. Jongejan, H. Rowley, T. Kawashima, J. Kim, and N. Fox-Gieg. "The Quick, Draw! - A.I. Experiment". In: *https://quickdraw.withgoogle.com* (2016) (cit. on p. 43).

[JKG16] E. Jorge, M. Kågebäck, and E. Gustavsson. "Learning to play guess who? and inventing a grounded language as a consequence". In: *CoRR* abs/1611.03218 (2016) (cit. on p. 31).

[JN20] A. Jung and P. H. J. Nardelli. "An Information-Theoretic Approach to Personalized Explainable Machine Learning". In: *IEEE Signal Process. Lett.* 27 (2020) (cit. on p. 11).

[KJL14] A. Karpathy, A. Joulin, and F. Li. "Deep Fragment Embeddings for Bidirectional Image Sentence Mapping". In: *NeurIPS*. 2014 (cit. on p. 58).

[KL15] A. Karpathy and F. Li. "Deep visual-semantic alignments for generating image descriptions". In: *CVPR*. 2015 (cit. on p. 58).

[KLA19] T. Karras, S. Laine, and T. Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *CVPR*. 2019 (cit. on p. 10).

[KW52] J. Kiefer and J. Wolfowitz. “Stochastic Estimation of the Maximum of a Regression Function”. In: *Ann. Math. Statist.* Vol. 23. 1952 (cit. on p. 10).

[Kim+18] B. Kim, M. Wattenberg, J. Gilmer, C. J. Cai, J. Wexler, F. B. Viégas, and R. Sayres. “Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)”. In: *ICML*. 2018 (cit. on p. 11).

[KW14] D. P. Kingma and M. Welling. “Auto-Encoding Variational Bayes”. In: *ICLR*. 2014 (cit. on pp. 19, 55, 57, 61).

[Kle+18] J. Kleinberg, H. Lakkaraju, J. Leskovec, J. Ludwig, and S. Mullainathan. “Human decisions and machine predictions”. In: *The Quarterly Journal of Economics* 133.1 (2018) (cit. on p. 16).

[Kon01] I. Kononenko. “Machine learning for medical diagnosis: history, state of the art and perspective”. In: *Artificial Intelligence in Medicine* 23.1 (2001) (cit. on p. 16).

[Kon+16] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò. “Deep Neural Decision Forests”. In: *IJCAI*. 2016 (cit. on pp. 17, 22, 23).

[KHW19] W. Kool, H. van Hoof, and M. Welling. “Attention, Learn to Solve Routing Problems!” In: *ICLR*. 2019 (cit. on p. 12).

[Kot+17] S. Kottur, J. Moura, S. Lee, and D. Batra. “Natural Language Does Not Emerge ‘Naturally’ in Multi-Agent Dialog”. In: *EMNLP*. 2017 (cit. on pp. 31, 81).

[Kri09] A. Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. University of Toronto, 2009 (cit. on p. 62).

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *NIPS*. 2012 (cit. on p. 10).

[KBN08] N. Kumar, P. Belhumeur, and S. Nayar. “FaceTracer: A Search Engine for Large Collections of Images with Faces”. In: *ECCV*. 2008 (cit. on p. 17).

[KT20] S. Kumar and P. P. Talukdar. “NILE : Natural Language Inference with Faithful Natural Language Explanations”. In: *ACL*. 2020 (cit. on p. 80).

[Lag+18] I. Lage, A. Ross, S. J. Gershman, B. Kim, and F. Doshi-Velez. “Human-in-the-loop interpretability prior”. In: *NeurIPS*. 2018 (cit. on p. 17).

[LBL16] H. Lakkaraju, S. H. Bach, and J. Leskovec. “Interpretable decision sets: A joint framework for description and prediction”. In: *ACM SIGKDD*. 2016 (cit. on p. 29).

[LNH14] C. H. Lampert, H. Nickisch, and S. Harmeling. “Attribute-Based Classification for Zero-Shot Visual Object Categorization”. In: *IEEE TPAMI* 36.3 (2014) (cit. on pp. 17, 22).

[LB20] A. Lazaridou and M. Baroni. “Emergent Multi-Agent Communication in the Deep Learning Era”. In: *CoRR:2006.02419* (2020) (cit. on pp. 12, 17, 81).

[Laz+18] A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark. "Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input". In: *ICLR*. 2018 (cit. on pp. 12, 17, 31, 81).

[LPB17] A. Lazaridou, A. Peysakhovich, and M. Baroni. "Multi-agent cooperation and the emergence of (natural) language". In: *ICLR*. 2017 (cit. on p. 31).

[LPT20] A. Lazaridou, A. Potapenko, and O. Tieleman. "Multi-agent Communication meets Natural Language: Synergies between Functional and Structural Language Learning". In: *ACL*. 2020 (cit. on pp. 12, 17, 81).

[LeC+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. "Back-propagation applied to handwritten zip code recognition". In: *Neural Computation*. Vol. 1. 1989 (cit. on p. 10).

[Lec+98] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998) (cit. on p. 62).

[Let+15] B. Letham, C. Rudin, T. H. McCormick, D. Madigan, et al. "Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model". In: *The Annals of Applied Statistics* 9 (2015) (cit. on p. 29).

[Li+19] L. H. Li, M. Yatskar, D. Yin, C. Hsieh, and K. Chang. "VisualBERT: A Simple and Performant Baseline for Vision and Language". In: *CoRR* abs/1908.03557 (2019) (cit. on p. 58).

[Li+21] Y. Li, F. Liang, L. Zhao, Y. Cui, W. Ouyang, J. Shao, F. Yu, and J. Yan. "Supervision Exists Everywhere: A Data Efficient Contrastive Language-Image Pre-training Paradigm". In: *CoRR* abs/2110.05208 (2021) (cit. on pp. 58, 81).

[LSG13] Y. Li, Y. Song, and S. Gong. "Sketch Recognition by Ensemble Matching of Structured Features". In: *BMVC*. 2013 (cit. on p. 43).

[LZY20] W. Liang, J. Zou, and Z. Yu. "ALICE: Active Learning with Contrastive Natural Language Explanations". In: *EMNLP*. 2020 (cit. on pp. 58, 80).

[Liu+18a] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L. Li, L. Fei-Fei, A. L. Yuille, J. Huang, and K. Murphy. "Progressive Neural Architecture Search". In: *ECCV*. 2018 (cit. on p. 34).

[Liu+21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *ICCV*. 2021 (cit. on p. 10).

[Liu+18b] Z. Liu, W. T. Freeman, J. B. Tenenbaum, and J. Wu. "Physical primitive decomposition". In: *ECCV*. 2018 (cit. on p. 44).

[Liu+22] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. "A ConvNet for the 2020s". In: *CVPR*. 2022 (cit. on p. 10).

[Lom12] T. Lombrozo. *Explanation and abductive inference.* The Oxford handbook of thinking and reasoning, 2012 (cit. on p. 11).

[Lou+13] G. Louppe, L. Wehenkel, A. Sutera, and P. Geurts. "Understanding variable importances in forests of randomized trees". In: *NeurIPS*. 2013 (cit. on p. 24).

[MHB14] W. J. Ma, M. Husain, and P. M. Bays. "Changing concepts of working memory". In: *Nature Neuroscience* 17.3 (2014) (cit. on p. 17).

[MMT17] C. J. Maddison, A. Mnih, and Y. W. Teh. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *ICLR*. 2017 (cit. on p. 19).

[MSN11] D. K. Mahajan, S. Sellamanickam, and V. Nair. "A joint learning framework for attribute models and object descriptions". In: *ICCV*. 2011 (cit. on p. 17).

[Maj+21] B. P. Majumder, O. Camburu, T. Lukasiewicz, and J. J. McAuley. "Rationale-Inspired Natural Language Explanations with Commonsense". In: *CoRR* abs/2106.13876 (2021) (cit. on p. 80).

[Mar+20] D. Marcos, R. Fong, S. Lobry, R. Flamary, N. Courty, and D. Tuia. "Contextual Semantic Interpretability". In: *ACCV*. 2020 (cit. on p. 17).

[Mar20] G. Marcus. "The next decade in ai: four steps towards robust artificial intelligence". In: *CoRR:2002.06177* (2020) (cit. on p. 18).

[Mar+04] R. Marée, P. Geurts, J. Piater, L. Wehenkel, K.-S. Hong, and Z. Zhang. "A generic approach for image classification based on decision tree ensembles and local subwindows". In: *ACCV*. 2004 (cit. on p. 15).

[MKR21] A. F. Markus, J. A. Kors, and P. R. Rijnbeek. "The role of explainability in creating trustworthy artificial intelligence for health care: A comprehensive survey of the terminology, design choices, and evaluation strategies". In: *J. Biomed. Informatics* 113 (2021) (cit. on p. 2).

[Mei03] M. Meilă. "Comparing clusterings by the variation of information". In: *Learning theory and kernel machines*. Springer, 2003 (cit. on p. 36).

[Mik+13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. "Distributed Representations of Words and Phrases and their Compositionality". In: *NeurIPS*. 2013 (cit. on p. 56).

[Mil19] T. Miller. "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial Intelligence* 267 (2019) (cit. on p. 11).

[MA18] I. Mordatch and P. Abbeel. "Emergence of grounded compositional language in multi-agent populations". In: *AAAI*. 2018 (cit. on p. 31).

[Muh+19] U. R. Muhammad, Y. Yang, T. M. Hospedales, T. Xiang, and Y. Song. "Goal-Driven Sequential Data Abstraction". In: *ICCV*. 2019 (cit. on pp. 42, 43, 47–51, 114, 115).

[Muh+18] U. R. Muhammad, Y. Yang, Y. Song, T. Xiang, and T. M. Hospedales. "Learning Deep Sketch Abstraction". In: *CVPR*. 2018 (cit. on pp. 42–44, 47–51, 111, 113–115).

[Mur+16] V. N. Murthy, V. Singh, T. Chen, R. Manmatha, and D. Comaniciu. "Deep Decision Network for Multi-class Image Classification". In: *CVPR*. 2016 (cit. on p. 17).

[NBT16] R. Nakahashi, C. L. Baker, and J. B. Tenenbaum. "Modeling human understanding of complex intentional action with a bayesian nonparametric subgoal model". In: *AAAI*. 2016 (cit. on p. 30).

[NHH15] H. Noh, S. Hong, and B. Han. "Learning Deconvolution Network for Semantic Segmentation". In: *ICCV*. 2015 (cit. on p. 10).

[Nta+20] E. Ntavelis, A. Romero, I. Kastanis, L. Van Gool, and R. Timofte. "SESAME: Semantic Editing of Scenes by Adding, Manipulating or Erasing Objects". In: *ECCV*. 2020 (cit. on p. 69).

[OVK17] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. "Neural Discrete Representation Learning". In: *NeurIPS*. 2017 (cit. on pp. 69–71).

[OKB11] V. Ordonez, G. Kulkarni, and T. L. Berg. "Im2Text: Describing Images Using 1 Million Captioned Photographs". In: *NeurIPS*. 2011 (cit. on p. 17).

[Pan+19] K. Pang, K. Li, Y. Yang, H. Zhang, T. M. Hospedales, T. Xiang, and Y.-Z. Song. "Generalising Fine-Grained Sketch-Based Image Retrieval". In: *CVPR*. 2019 (cit. on p. 44).

[Pan+20] K. Pang, Y. Yang, T. M. Hospedales, T. Xiang, and Y.-Z. Song. "Solving Mixed-Modal Jigsaw Puzzle for Fine-Grained Sketch-Based Image Retrieval". In: *CVPR*. 2020 (cit. on pp. 43, 44).

[Par+21] W. Para, S. Bhat, P. Guerrero, T. Kelly, N. Mitra, L. J. Guibas, and P. Wonka. "Sketchgen: Generating constrained cad sketches". In: *NeurIPS* (2021) (cit. on p. 44).

[Par+18] D. H. Park, L. A. Hendricks, Z. Akata, A. Rohrbach, B. Schiele, T. Darrell, and M. Rohrbach. "Multimodal Explanations: Justifying Decisions and Pointing to the Evidence". In: *CVPR*. 2018 (cit. on pp. 11, 15).

[Par+19] T. Park, M. Liu, T. Wang, and J. Zhu. "Semantic Image Synthesis With Spatially-Adaptive Normalization". In: *CVPR*. 2019 (cit. on p. 69).

[PM14] S. Parui and A. Mittal. "Similarity-Invariant Sketch-Based Image Retrieval in Large Databases". In: *ECCV*. 2014 (cit. on p. 43).

[Pat+14] G. Patterson, C. Xu, H. Su, and J. Hays. "The SUN Attribute Database: Beyond Categories for Deeper Scene Understanding". In: *International Journal of Computer Vision* 108 (2014) (cit. on p. 34).

[Ped+20] T. Pedapati, A. Balakrishnan, K. Shanmugam, and A. Dhurandhar. "Learning Global Transparent Models consistent with Local Contrastive Explanations". In: *NeurIPS*. 2020 (cit. on p. 11).

[Ped+19] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini. "Meaningful Explanations of Black Box AI Decision Systems". In: *AAAI*. 2019 (cit. on p. 11).

[PSM14] J. Pennington, R. Socher, and C. D. Manning. "Glove: Global Vectors for Word Representation". In: *EMNLP*. 2014 (cit. on p. 56).

[PDS18] V. Petsiuk, A. Das, and K. Saenko. "RISE: Randomized Input Sampling for Explanation of Black-box Models". In: *BMVC*. 2018 (cit. on pp. 11, 80).

[Rab+18] N. Rabinowitz, F. Perbet, F. Song, C. Zhang, S. M. A. Eslami, and M. Botvinick. "Machine Theory of Mind". In: *ICML*. 2018 (cit. on p. 30).

[Rad+21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *ICML*. 2021 (cit. on pp. 58, 68, 81).

[RS04] L. E. Raileanu and K. Stoffel. "Theoretical Comparison between the Gini Index and Information Gain Criteria". In: *Annals of Mathematics and Artificial Intelligence* 41.1 (2004) (cit. on p. 23).

[Ram+20] K. N. Ramamurthy, B. Vinzamuri, Y. Zhang, and A. Dhurandhar. "Model Agnostic Multilevel Explanations". In: *NeurIPS*. 2020 (cit. on p. 11).

[ROV19] A. Razavi, A. van den Oord, and O. Vinyals. "Generating Diverse High-Fidelity Images with VQ-VAE-2". In: *NeurIPS*. 2019 (cit. on pp. 69–71).

[Red+21] P. Reddy, M. Gharbi, M. Lukac, and N. J. Mitra. "Im2vec: Synthesizing vector graphics without vector supervision". In: *CVPR*. 2021 (cit. on p. 44).

[Red+16] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *CVPR*. 2016 (cit. on p. 58).

[RF17] J. Redmon and A. Farhadi. "YOLO9000: Better, Faster, Stronger". In: *CVPR*. 2017 (cit. on p. 10).

[Ren+15] S. Ren, K. He, R. B. Girshick, and J. Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *NeurIPS*. 2015 (cit. on pp. 10, 58, 63).

[RMW14] D. J. Rezende, S. Mohamed, and D. Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *ICML*. 2014 (cit. on p. 19).

[Rib+20] L. S. F. Ribeiro, T. Bui, J. Collomosse, and M. Ponti. "Sketchformer: Transformer-based representation for sketched structure". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 14153–14162 (cit. on p. 43).

[RSG16] M. T. Ribeiro, S. Singh, and C. Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *ACM SIGKDD*. 2016 (cit. on p. 11).

[RM51] H. Robbins and S. Monro. “A Stochastic Approximation Method”. In: *The Annals of Mathematical Statistics*. Vol. 22. 1951 (cit. on p. 10).

[Roh+16] A. Rohrbach, M. Rohrbach, R. Hu, T. Darrell, and B. Schiele. “Grounding of Textual Phrases in Images by Reconstruction”. In: *ECCV*. 2016 (cit. on p. 58).

[RP68] A. Rosenfeld and J. L. Pfaltz. “Distance functions on digital pictures”. In: *Pattern recognition* 1.1 (1968) (cit. on p. 45).

[RIA93] B. Roskos-Ewoldsen, M. J. Intons-Peterson, and R. E. Anderson. *Imagery, creativity, and discovery: A cognitive perspective*. Elsevier, 1993 (cit. on p. 42).

[Rud19] C. Rudin. “Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead”. In: *Nature Machine Intelligence* 1 (2019) (cit. on p. 29).

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning internal representations by error propagation”. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Vol. 1. 1986 (cit. on p. 10).

[Rus+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”. In: *IJCV* 115.3 (2015) (cit. on p. 22).

[Saa14] J. M. Saavedra. “Sketch based image retrieval using a soft computation of the histogram of edge local orientations (S-HELO)”. In: *ICIP*. 2014 (cit. on p. 44).

[Saa17] J. M. Saavedra. “Rst-shelo: sketch-based image retrieval using sketch tokens and square root normalization”. In: *Multimedia Tools and Applications* 76.1 (2017), pp. 931–951 (cit. on p. 44).

[SBO15] J. M. Saavedra, J. M. Barrios, and S. Orand. “Sketch based Image Retrieval using Learned KeyShapes (LKS).” In: *BMVC*. 2015 (cit. on p. 44).

[San+11] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. “Segmentation as selective search for object recognition”. In: *ICCV*. 2011 (cit. on p. 63).

[Sch87] J. Schmidhuber. “Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-... hook”. Diplomarbeit. München: Technische Universität München, 1987 (cit. on p. 31).

[ST14] R. G. Schneider and T. Tuytelaars. “Sketch classification and classification-driven analysis using fisher vectors”. In: *ACM Transactions on graphics (TOG)* 33.6 (2014) (cit. on p. 43).

[Sch+20] J. Schrittwieser, I. Antonoglou, T. Hubert, S. Karen, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. P. Lillicrap, and D. Silver. “Mastering Atari, Go, chess and shogi by planning with a learned model”. In: *Nature.* 588 (2020) (cit. on p. 1).

[SK19] P. Schwab and W. Karlen. “CXPlain: Causal Explanations for Model Interpretation under Uncertainty”. In: *NeurIPS*. 2019 (cit. on p. 11).

[Sef+21] A. Seff, W. Zhou, N. Richardson, and R. P. Adams. “Vitruvion: A Generative Model of Parametric CAD Sketches”. In: *International Conference on Learning Representations*. 2021 (cit. on p. 44).

[SCH66] M. H. Segall, D. T. Campbell, and M. J. Herskovits. “The influence of culture on visual perception.” In: (1966) (cit. on p. 42).

[Sel+17] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: *ICCV*. 2017 (cit. on pp. 11, 80).

[Sha48] C. E. Shannon. “A mathematical theory of communication”. In: *Bell system technical journal* 27 (1948) (cit. on p. 13).

[SQL12] V. Sharmanska, N. Quadrianto, and C. H. Lampert. “Augmented Attribute Representations”. In: *ECCV*. 2012 (cit. on p. 17).

[She+21] S. Shen, L. H. Li, H. Tan, M. Bansal, A. Rohrbach, K. Chang, Z. Yao, and K. Keutzer. “How Much Can CLIP Benefit Vision-and-Language Tasks?” In: *CoRR* abs/2107.06383 (2021) (cit. on pp. 58, 81).

[SFS20] R. Shetty, M. Fritz, and B. Schiele. “Towards Automated Testing and Robustification by Semantic Adversarial Data Generation”. In: *ECCV*. 2020 (cit. on p. 69).

[Shi+14] Z. Shi, Y. Yang, T. M. Hospedales, and T. Xiang. “Weakly Supervised Learning of Objects, Attributes and Their Associations”. In: *ECCV*. 2014 (cit. on p. 17).

[SGK17] A. Shrikumar, P. Greenside, and A. Kundaje. “Learning Important Features Through Propagating Activation Differences”. In: *ICML*. 2017 (cit. on p. 11).

[ST19] T. Shu and Y. Tian. “M$^3$RL: Mind-aware Multi-agent Management Reinforcement Learning”. In: *ICLR*. 2019 (cit. on p. 30).

[Shu+18] T. Shu, C. Xiong, Y. N. Wu, and S. Zhu. “Interactive Agent Modeling by Learning to Probe”. In: *CoRR* abs/1810.00510 (2018) (cit. on p. 30).

[SFD11] B. Siddiquie, R. Feris, and L. Davis. “Image ranking and retrieval based on multi-attribute queries.” In: *CVPR*. 2011 (cit. on p. 17).

[Sil+16] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature.* 529 (2016) (cit. on p. 1).

[SZ15] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *ICLR*. 2015 (cit. on p. 10).

[Siu19] C. Siu. "Transferring Tree Ensembles to Neural Networks". In: *ICONIP*. 2019 (cit. on p. 17).

[SBS21] D. Smirnov, M. Bessmeltsev, and J. Solomon. "Deep sketch-based modeling of man-made shapes". In: *ICLR*. 2021 (cit. on p. 44).

[Son+17] J. Song, Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales. "Deep Spatial-Semantic Attention for Fine-Grained Sketch-Based Image Retrieval". In: *ICCV*. 2017 (cit. on pp. 44, 48, 111).

[SF19] S. Srinivas and F. Fleuret. "Full-Gradient Representation for Neural Network Visualization". In: *NeurIPS*. 2019 (cit. on p. 11).

[SF21] S. Srinivas and F. Fleuret. "Rethinking the Role of Gradient-based Attribution Methods for Model Interpretability". In: *ICLR*. 2021 (cit. on p. 11).

[Sut+00] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. "Policy gradient methods for reinforcement learning with function approximation". In: *NeurIPS*. 2000 (cit. on pp. 12, 34).

[SB98] R. S. Sutton and A. G. Barto. *Reinforcement learning - an introduction*. MIT Press, 1998 (cit. on p. 12).

[Sze+16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the Inception Architecture for Computer Vision". In: *CVPR*. 2016 (cit. on pp. 10, 111).

[TB19] H. Tan and M. Bansal. "LXMERT: Learning Cross-Modality Encoder Representations from Transformers". In: *EMNLP-IJCNLP*. 2019 (cit. on p. 58).

[Tan+19] R. Tanno, K. Arulkumaran, D. Alexander, A. Criminisi, and A. Nori. "Adaptive neural trees". In: *ICML*. 2019 (cit. on p. 17).

[TM12] D. Taubman and M. Marcellin. *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*. Vol. 642. Springer Science & Business Media, 2012 (cit. on p. 44).

[TC17] G. Tolias and O. Chum. "Asymmetric Feature Maps with Application to Sketch Based Retrieval". In: *CVPR*. 2017 (cit. on p. 43).

[Tou+21] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. "Training data-efficient image transformers & distillation through attention". In: *ICML*. 2021 (cit. on p. 10).

[Vas+17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. "Attention is All you Need". In: *NeurIPS*. 2017 (cit. on pp. 10, 48, 69, 71, 111).

[Vin+19] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, Ç. Gülçehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. P. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature.* 575 (2019) (cit. on p. 1).

[Wah+11] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011 (cit. on pp. 22, 34).

[WB13] C. Wah and S. Belongie. "Attribute-Based Detection of Unfamiliar Classes with Humans in the Loop". In: *CVPR*. 2013 (cit. on p. 17).

[Wal92] G. K. Wallace. "The JPEG still picture compression standard". In: *IEEE transactions on consumer electronics* 38.1 (1992) (cit. on p. 44).

[Wan+21] A. Wan, L. Dunlap, D. Ho, J. Yin, S. Lee, H. Jin, S. Petryk, S. A. Bargal, and J. E. Gonzalez. "NBDT: Neural-Backed Decision Trees". In: *ICLR*. 2021 (cit. on p. 17).

[Wan+04] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE transactions on image processing* (2004) (cit. on p. 72).

[War+12] M. Warnier, J. Guitton, S. Lemaignan, and R. Alami. "When the robot puts itself in your shoes. managing and exploiting human and robot beliefs". In: *IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. 2012 (cit. on p. 30).

[Wil92] R. J. Williams. "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". In: *Mach. Learn.* 8 (1992) (cit. on p. 12).

[Wri+10] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan. "Sparse representation for computer vision and pattern recognition". In: *Proceedings of the IEEE* 98.6 (2010) (cit. on p. 17).

[WM19] J. Wu and R. Mooney. "Faithful Multimodal Explanation for Visual Question Answering". In: *Proceedings of the ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. 2019 (cit. on p. 29).

[Wu+15] J. Wu, C. Wang, L. Zhang, and Y. Rui. "Offline sketch parsing via shapeness estimation". In: *AAAI*. 2015 (cit. on p. 44).

[WG18] M. Wu and N. Goodman. "Multimodal Generative Models for Scalable Weakly-Supervised Learning". In: *NeurIPS*. 2018 (cit. on p. 57).

[Wu+18] M. Wu, M. C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez. “Beyond sparsity: Tree regularization of deep models for interpretability”. In: *AAAI*. 2018 (cit. on p. 17).

[Xia+18] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. “Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly”. In: *IEEE transactions on pattern analysis and machine intelligence* (2018) (cit. on p. 34).

[Xia+19] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. “Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 41.9 (2019) (cit. on pp. 10, 22).

[Xia+15] C. Xiao, C. Wang, L. Zhang, and L. Zhang. “Sketch-based image retrieval via shape words”. In: *ACM ICMR*. 2015 (cit. on pp. 44, 49–51, 114, 115).

[XJB21] P. Xu, C. K. Joshi, and X. Bresson. “Multigraph Transformer for Free-Hand Sketch Recognition”. In: *IEEE TNNLS* (2021) (cit. on p. 43).

[Yan+21] L. Yang, K. Pang, H. Zhang, and Y.-Z. Song. “SketchAA: Abstract Representation for Abstract Sketches”. In: *ICCV*. 2021 (cit. on p. 43).

[Yao+11] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. J. Guibas, and F. Li. “Human action recognition by learning bases of action attributes and parts”. In: *ICCV*. 2011 (cit. on p. 17).

[Yig+20] T. Yigitcanlar, L. Butler, E. Windle, K. C. Desouza, R. Mehmood, and J. M. Corchado. “Can Building "Artificially Intelligent Cities" Safeguard Humanity from Natural Disasters, Pandemics, and Other Catastrophes? An Urban Scholar’s Perspective”. In: *Sensors* 20 (2020) (cit. on p. 1).

[YG21] G. Yona and D. Greenfeld. “Revisiting Sanity Checks for Saliency Maps”. In: *NeurIPS*. 2021 (cit. on p. 80).

[Yu+17a] Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales. “SketchX! - Shoe/Chair fine-grained SBIR dataset”. In: (2017) (cit. on p. 47).

[Yu+17b] Q. Yu, Y. Yang, F. Liu, Y.-Z. Song, T. Xiang, and T. M. Hospedales. “Sketch-a-Net: A Deep Neural Network that Beats Humans”. In: *IJCV* (2017) (cit. on p. 43).

[Zen+20] H. Zeng, K. Joseph, A. Vest, and Y. Furukawa. “Bundle pooling for polygonal architecture segmentation problem”. In: *CVPR*. 2020 (cit. on p. 44).

[Zha+17a] Q. Zhang, R. Cao, Y. N. Wu, and S. Zhu. “Growing Interpretable Part Graphs on ConvNets via Multi-Shot Learning”. In: *AAAI*. Ed. by S. P. Singh and S. Markovitch. 2017 (cit. on p. 17).

[Zha+17b] Q. Zhang, R. Cao, Y. N. Wu, and S. Zhu. “Mining Object Parts from CNNs via Active Question-Answering”. In: *CVPR*. 2017 (cit. on p. 17).

[ZNZ18] Q. Zhang, Y. Nian Wu, and S.-C. Zhu. “Interpretable Convolutional Neural Networks”. In: *CVPR*. 2018 (cit. on p. 11).

[Zha+19] Q. Zhang, Y. Yang, H. Ma, and Y. N. Wu. "Interpreting cnns via decision trees". In: *CVPR*. 2019 (cit. on p. 11).

[Zha+18] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. "The unreasonable effectiveness of deep features as a perceptual metric". In: *CVPR*. 2018 (cit. on p. 72).

[Zha+21] Y. Zhang, P. Tiňo, A. Leonardis, and K. Tang. "A Survey on Neural Network Interpretability". In: *IEEE Trans. Emerg. Top. Comput. Intell.* 5 (2021) (cit. on p. 2).

[ZLJ16] Y. Zhang, J. D. Lee, and M. I. Jordan. "L1-regularized neural networks are improperly learnable in polynomial time". In: *ICML*. 2016 (cit. on p. 17).

[ZSE17] S. Zhao, J. Song, and S. Ermon. "Towards Deeper Understanding of Variational Autoencoding Models". In: *CoRR* (2017) (cit. on p. 56).

[Zhe+20] Z. Zheng, L. Zheng, M. Garrett, Y. Yang, M. Xu, and Y. Shen. "Dual-path Convolutional Image-Text Embeddings with Instance Loss". In: *ACM Trans. Multim. Comput. Commun. Appl.* (2020) (cit. on p. 58).

[Zho+17] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. "Scene parsing through ade20k dataset". In: *CVPR*. 2017 (cit. on p. 72).

[ZW12] Q. Zhou and G. Wang. "Atomic Action Features: A New Feature for Action Recognition". In: *ECCVW*. 2012 (cit. on p. 17).

[Zin+17] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling. "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis". In: *ICLR*. 2017 (cit. on p. 11).

# A

# SUPPLEMENTARY MATERIAL - LEARNING DECISION TREES RECURRENTLY THROUGH COMMUNICATION

## A.1 Ablating Loss and Maximum Steps T

When training our RDTC model with a cross-entropy loss for image classification, we found that training is more efficient and yields better results when applying the loss term at every step $t$ in the communication loop up to the maximum step $T$.

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{CE}(y, \hat{y}^{(t)}) \tag{A.1}$$

One natural alternative to this approach is to apply the loss only at step $T$, the leaf node, essentially removing the sum Equation A.1. In Figure A.1, we show the difference of applying the loss at every time step (full loss) or only at the end (leaf loss) on CUB and AWA2. The final performance of the decision tree is the same, however, applying the loss at every time step produces a tree that has a better performance when evaluated at intermediate steps and results in a smaller tree after pruning, i.e., fewer tree nodes are used for the final tree to obtain best performance.

Moreover, we found that when hyperparameter $T$ is chosen sufficiently high, we are able to reach this maximum performance while our tree distillation process ensures that the tree size does not increase past the point where the classifier achieves the highest accuracy. Figure A.2 shows classification accuracy with increasing tree depth. Accuracy does not decrease past some value for $T$ where the model performs best, and choosing any value bigger results in an equally explainable tree after pruning.

## A.2 Decision Trees and Explanations of CUB and AWA2

Illustrating the decision making process helps the user get an explainable overview of the internal decision process of the whole classifier. We point to the tree branch into which a certain class (indicated by an example image from this class) falls along with the attribute associated with that

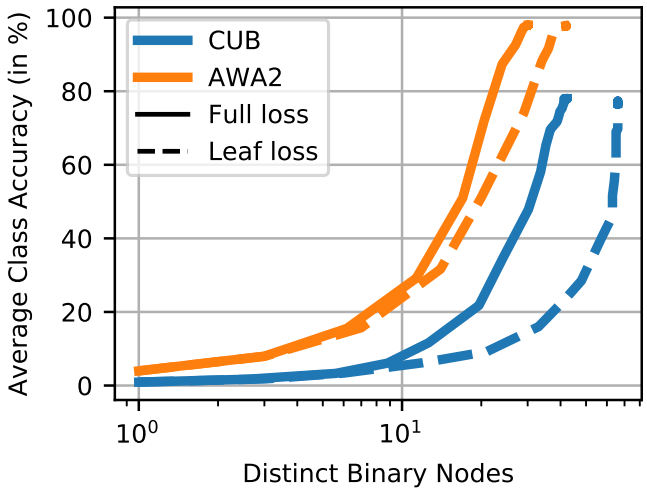



Figure A.1: Accuracy and number of distinct nodes in RDTC on AWA2, CUB comparing our full loss at each time step (solid line) with a loss only applied at leaf nodes (dashed line). The full loss uses fewer nodes, i.e., a smaller tree, to achieve the same accuracy.

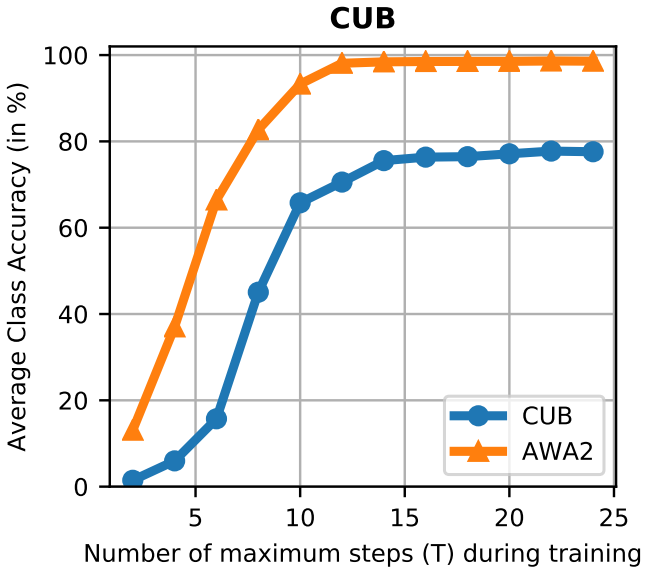



Figure A.2: Training RDTC while varying hyperparameter $T$. As $T$ increases, the model achieve a better accuracy up to a value of $T$ where a plateau is reached. When increasing $T$ further, the final tree size of RDTC does not increase due to pruning during tree distillation.

branch. We inspect the learned structure of the decision tree by illustrating the splits from our aRDTC model on CUB in Figure A.4, and on AWA2 in Figure A.6. Here, the left and right sub-tree indicates that the attribute is present or absent respectively. For instance, on CUB, the first decision deals with identifying bird with white underparts, separating these from birds with any other color. These categories get further refined with each binary split via a hierarchical clustering that reveals the decision tree structure of our aRDTC framework. These serve as additional examples of introspection, showing that our model allows to make a more informed decision about the

trustworthiness of the network's prediction.

In Figure A.5, we illustrate a qualitative example of the classification of two images of Scarlet Tanagers made by our model trained on CUB. Both images follow the same path for the first decisions, before diverging when it comes to the decision whether the bird has black wings. The top bird actually does not have black wings and, thus, is classified as a Summer Tanager, a bird species with the same appearance as Scarlet Tanager except for having red instead of black wings.

Equivalently in Figure A.7, we illustrate a qualitative example of the classification of two images of tigers made by our model trained on AWA2. Again, both images follow the same decision until, for the white tiger, our model wrongly predicts "no stripes" and incorrectly classifies it as a lion. Together with the full decision trees, these explanations allow for detailed introspections into the global decision process our `aRDTC` model.

## A.3 Explanations without Attributes

When working with datasets that do not provide annotated attributes, we can train our `RDTC` with $\lambda = 0$, which still exposes the decision tree structure. This allows introspection into the intermediate class splits of the model revealing a hierarchy that can reveal semantics. When applies on CIFAR-10, our `RDTC` model not only retains ResNet performance ($93.1\%$ vs. $93.3\%$), it also semantically clusters the data even though there is no attribute guidance. Figure A.3 shows the resulting decision tree of `RDTC` on CIFAR-10. In the first binary split, we observe that `RDTC` separates the animal classes from the vehicles. Subsequently, vehicles are clustered into motor vehicles (car, truck) and the rest (airplane, ship). For animals, our model also finds reasonable clusters such as grouping cat and dog, as well as grouping horse and deer. ImageNet is a more challenging dataset, where we observe similar behaviour. In Figure A.8, we show the decision tree of the first decisions on ImageNet with a randomly selected subset of classes, each represented by one representative image. Our model separates animals from inanimate objects in the first tree split following the data semantics. In the later decisions of the tree, there are clusters of dogs/cats, birds, monkeys on one side of the tree and clusters of furniture and electrical appliances on the other. These example show that, even when no additional attribute information is given, tree splits often follow semantics that are exposed by the decision tree learned by our `RDTC`.

## A.4 RDTC Training Algorithm

For a concise representation of the `RDTC` training algotithm, we present a summary in Algorithm 2 including both components, RDT and AbL, iterative loss calculation and gradient updates using the terminology of the main paper.

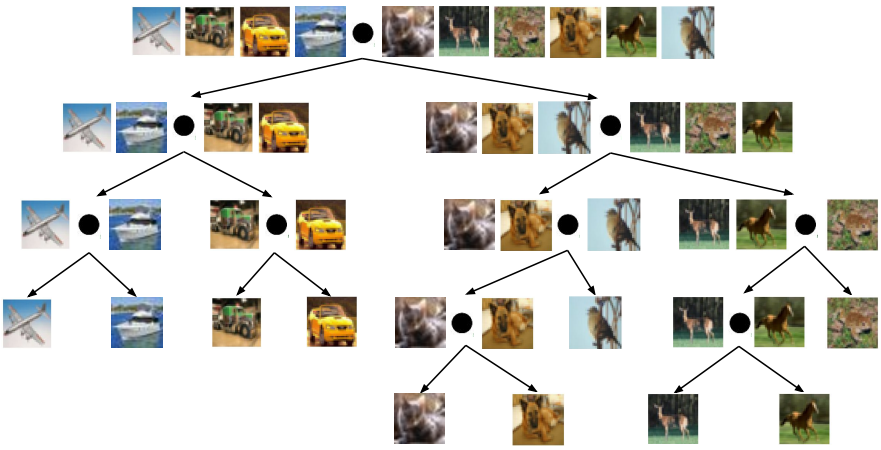


Figure A.3: Our RDTC learns the decision tree on CIFAR10 without attribute data. While decision nodes do not have ground truth attributes, we can still interpret the decision, e.g., the first node separates animals from vehicles.

**Algorithm 2** RDTC training

**Input:** Image $x$, label $y$
Max # of decisions $T$, Attribute data $\alpha$
**Output:** Predicted label $\hat{y}$
Binary decision sequence $d^{(1)}, \ldots, d^{(T)}$

$z = \text{CNN}(x)$
$\hat{\boldsymbol{a}} = \text{TempSoftmax}(f_{\text{AttrMLP}}(z))$
init $\mathcal{M}^0, h_0$
$\mathcal{L} = 0$
**for** $t = 1$ to $T$ **do**
  $c^{(t)} = \text{GumbelSoftmax}(f_{\text{QuestMLP}}(h^{(t-1)}))$
  $d^{(t)} = \hat{a}_{c^{(t)}}$
  $\mathcal{M}^{(t)} = \mathcal{M}^{(t-1)} \oplus (c^{(t)}, d^{(t)})$
  $h^{(t)} = \text{LSTM}(h^{(t-1)}, \mathcal{M}^{(t)}, c^{(t)}, d^{(t)})$
  $\hat{y}^{(t)} = f_{\text{ClassMLP}}(\mathcal{M}^{(t)})$
  $\mathcal{L}^{(t)} = \frac{1}{T}\Big[(1-\lambda)\mathcal{L}_{CE}(y, \hat{y}^{(t)}) + \lambda\mathcal{L}_{CE}(\alpha_{y,c^{(t)}}, \hat{a}_{c^{(t)}})\Big]$
  $\mathcal{L} = \mathcal{L} + \mathcal{L}^{(t)}$
**end for**
gradient update with $\mathcal{L}$
**return** $\hat{y}^{(T)}; d^{(1)}, \ldots, d^{(T)}$

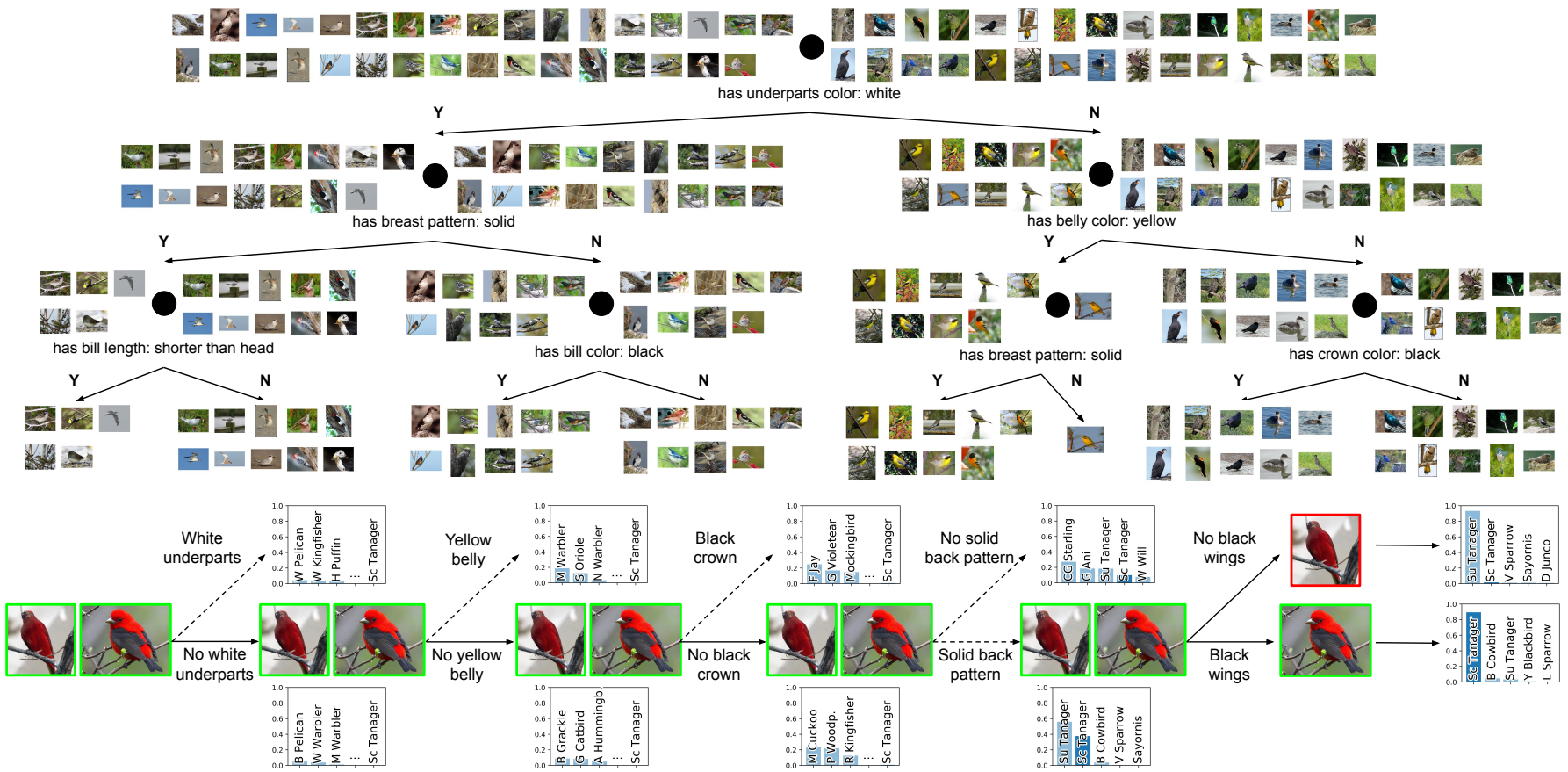



Figure A.5: Our aRDTC points to the reasoning behind a wrong decision. Here we illustrate two images from the "Green Kingfisher" class. The lower path lead to a correct classification. Both images follow the same path except for the decision of "black wings". The flying bird gets classified as a "Belted Kingfisher" incorrectly because the black wings are not visible.

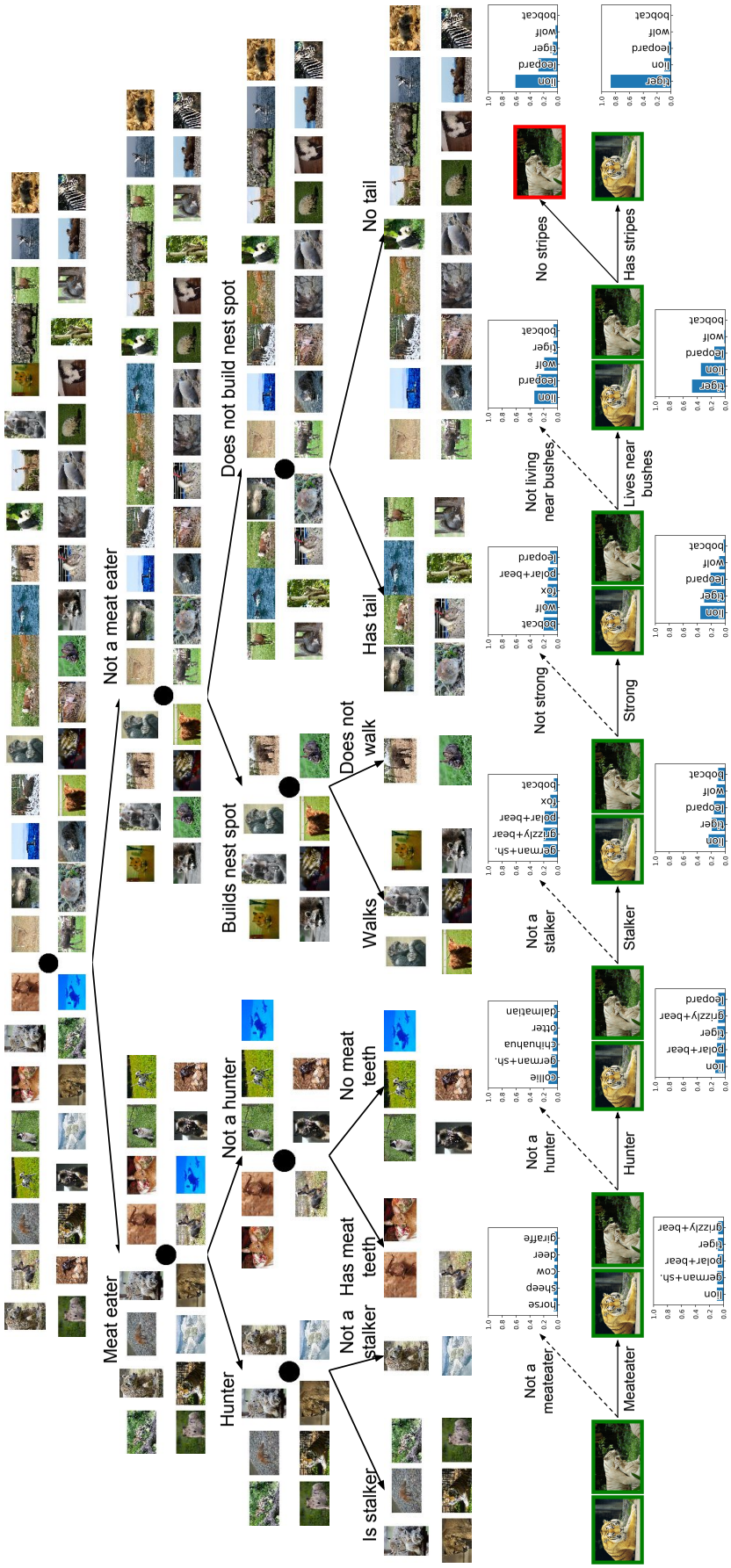



Figure A.7: Decision process for two tiger images in AWA2 along with the current label prediction at each step. The lower (upper) path is taken when the attribute is present (absent) for a given class. Both images follow the same path except for the last decision, "has stripes". Since these are absent in the white tiger, it gets classified incorrectly as a lion.

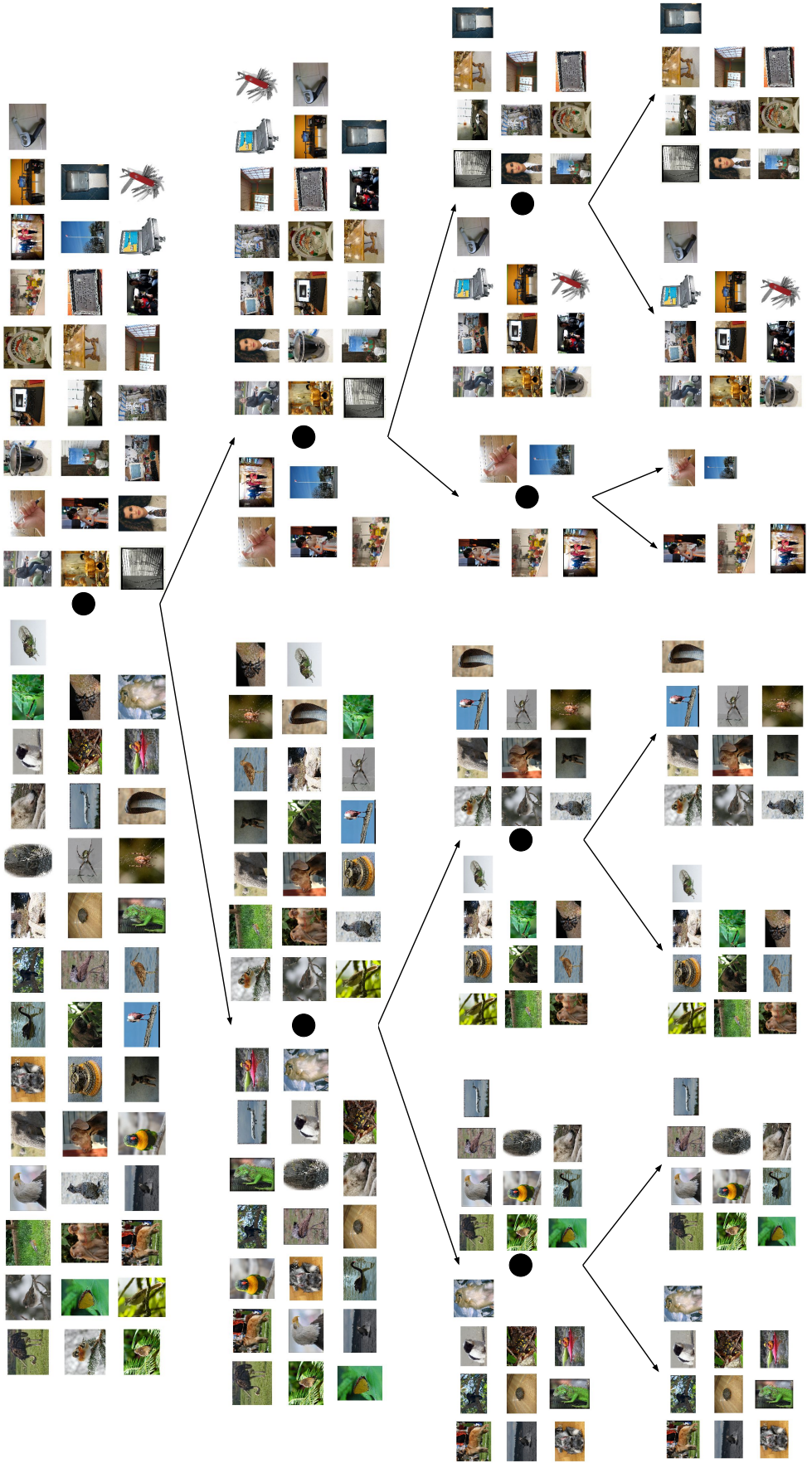


Figure A.8: Learned binary decisions on ImageNet by our RDTC model. A subset of randomly chosen classes are shown by one representative image of each class. Our tree reveals a clustering as decision splits narrow down towards a specific subset of classes.

# B

# SUPPLEMENTARY MATERIAL - ABSTRACTING SKETCHES THROUGH SIMPLE PRIMITIVES

## B.1 Network Architecture Details

For all our networks that take sketches as input, we use the Transformer [Vas+17] architecture with GeLU activations [HG16], 8 self-attention heads and a layer size of 128 for the self-attention layers and 512 for the fully-connected layers. Our network $f$ has 6 Transformer layers and embeds strokes by adding a single embedding token to the input sequence of points. We find that adding positional embeddings to encode the order of points does not improve the performance, so we only use the coordinates as input for each point.

We train a sketch classifier on Quickdraw with the same architecture as our stroke encoder, i.e. we use the 6-layer Transformer architecture. The first three layers embed strokes and the last three layers embed the full sketch by taking the sequence of stoke embeddings as input. First, we pass the sequence of points of individual strokes (or primitives in PMN) through the first three Transformer layers without positional embedding. To integrate stroke relative positions, the global position and scale are mapped linearly and added to the stroke embeddings. We then pass this stroke embedding sequence to the last three Transformer layers. A linear layer maps the final sketch embedding to the class logits before taking the cross entropy loss on the ground truth class label. When partial sketches are evaluated, we mask out unused strokes or sub-strokes at the input to obtain the predicted class logits.

For FG-SBIR, we train a Siamese network [Son+17] on the original training sketches with the same architecture of [Bhu+20; Muh+18], based on the InceptionV3 [Sze+16] architecture with an embedding size of 128. Since the network acts on natural images and images of sketches, we render our primitive reconstructions to images when evaluating them on the task. Similarly, partial sketches are rendered and fed to the CNN to obtain the retrieval scores for different budget and when applying DSA and GDSA.

| Batch size | with $\phi$ | without $\phi$ | factor |
|---|---|---|---|
| 32 | 13.14 ms | 118.82 ms | 9.04× |
| 64 | 14.28 ms | 226.46 ms | 15.85× |
| 128 | 20.16 ms | 449.22 ms | 22.28× |
| 256 | 34.79 ms | 905.66 ms | 26.03× |
| 512 | 65.23 ms | out-of-memory | - |

Table B.1: Average time is milliseconds for a forward pass of PMN using the compatibility function (with $\phi$) or not using it (without $\phi$) on a V100 GPU.

## B.2 Compatibility function

Our proposed PMN model uses a compatibility function $\phi$ to choose which primitive to match to a human stroke. In theory, the model can also be trained without $\phi$, where the loss functions is applied to each transformed primitive independently, regardless of how well a primitive fits a human stroke. At inference time, without $\phi$ the distance transform needs to be calculated for each transformed primitive to determine which one best fits the human stroke.

Using $\phi$ brings two main advantages. The first is reducing the inference time, without requiring the computation of the distance transform, and the second is providing a better definition of the training loss. Firstly, at inference time, we do not require the distance transform to be computed which is the most computationally expensive operation in our pipeline. To quantify this speed-up obtained by using $\phi$, we measured the time a forward pass takes on a V100 GPU (in milliseconds) with and without $\phi$ in Table B.1. We see a speed-up of an order of magnitude at small batch sizes of 32 to up to 26 times faster inference time at a batch size of 256. Using a batch size of 512 is not possible without $\phi$ as the 32GB of memory of the V100 is not sufficient to calculate all required distance transforms. On the other hand, we can use a batch size of up to 16384 when using $\phi$ (tested at powers of two).

Secondly, without $\phi$, the loss cannot reach zero, due to the distance transform between target strokes and their most different primitives (e.g. a circle-like human stroke vs the "line" primitive). With $\phi$ instead, the loss can become close to zero since only the most compatible primitives will be used to compute the loss. While we found no clear difference between the two strategies in terms of overall performance on downstream tasks, with $\phi$ the training loss becomes more expressive when comparing varying configurations and it avoids eventual loss spikes caused by matching primitives to strokes of very different shape.

## B.3 Affine Transformation

The affine transformation applied on primitives to reconstruct human strokes differs when computing the loss and when recreating a whole sketch. During training, human strokes and primitives are normalized to the range [-1, 1] while retaining the their aspect ratio by subtracting the mean of its points $\mu$ and then dividing by the size of the longest side $w$. The function $h(z_p^h, z_s^h)$ predicts the transformation $T_s^p$ to align $p$ with $s$ on this normalized scale. When reconstructing full sketches,

| Transformation | Budget 10% | 20% | 30% |
|---|---|---|---|
| rotate | 44.05 | 64.35 | 73.12 |
| scale | 57.39 | 69.59 | 73.92 |
| shear | 57.38 | 74.52 | 80.93 |
| scale, rotate | 64.75 | 80.08 | 85.46 |
| shear, rotate | 64.69 | 81.74 | 87.86 |
| shear, scale | 65.99 | 82.12 | 87.69 |
| shear, scale, rotate | **67.11** | **83.73** | 88.86 |
| rotate, scale, rotate | 67.08 | 83.69 | **89.15** |

Table B.2: Classification accuracy on Quickdraw at budgets of 10%, 20% and 30% for different types of transformations learned by $h$.

primitives are first transformed by $T_s^p$ followed by denormalizing based on the mean and size of the human stroke to obtain the transformed primitive $pT_s^p * w_s + \mu_s$. In practice, we combine the scaling factor $w_s$ and the translations $\mu_s$ into the transformations matrix $T_s^p$ before applying it to $p$ as it also assures that we always use at most six floating point values (maximum number of parameters of a 2D transformation matrix) for our fixed budget communication messages.

The affine transformation predicted by $h$ can be defined in several different ways. We do not allow arbitrary affine transformations in order to retain similarity with the original shape. For instance, if the scaling factors are not controlled, any shape can be collapsed into a line by applying a small factor on one of the axis. Therefore, we restrict the scaling transformation to be a proportional scale where one axis is scaled by a value between 0.05 and 1 while the other is fixed (at 1). Since scaling alone does not provide enough flexibility to fit primitives to strokes, we experiment with combining scaling with rotation and shear transformations. As reported in Table B.2, the composite transformation of rotate-scale-rotate works best and is chosen for all of our experiments. Notably, these transformations are applied in order, but except for rotate-scale-rotate, changing the order of the transformations, does not have a significant impact on the performance.

## B.4 Additional Quickdraw Results

**All budget levels.** Figure B.1 shows the performance of all evaluated methods at different budget levels between 0% and 100%. It illustrates the difference between selection-based and shape-based methods. While selection-based methods steadily increase in classification accuracy as the budget increases, shape-based methods have a more steep increase in the beginning and flatten off afterwards, making them favorable in low-budget regimes. As PMN performs lossy compression of the sketches, it requires at most a budget of 70% to abstract the whole sketch. The intersection of PMN with GDSA is at around 55% budget.

**Quickdraw-345.** In the main paper, we follow [Muh+18] and we use Quickdraw with 9 classes. Here, we also train and evaluate our PMN model and the compared methods on the Quickdraw dataset with 345 classes. Table B.3 shows the results. The trend is consistent with Quickdraw-9,

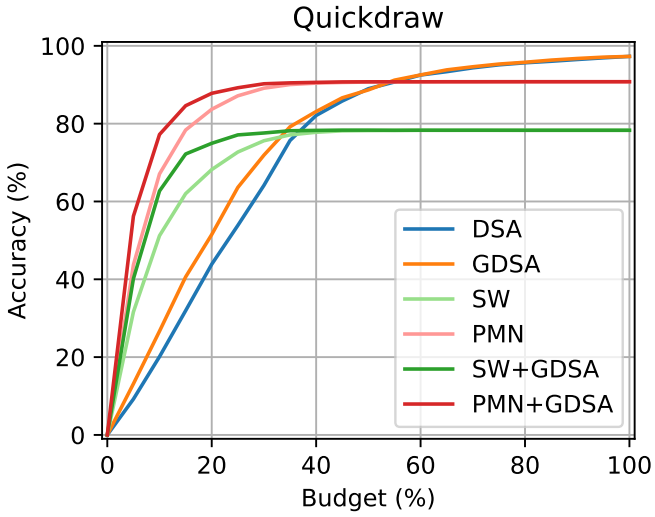



Figure B.1: Classification accuracy on Quickdraw at varying budgets between 0% and 100% evaluated with a classifier trained on the original human-drawn sketches.

with PMN+GDSA performing the best, and shape-based abstraction outperforming selection-based strategies at low budgets.

| Abstraction method | | Budget (%) | | | |
|---|---|---|---|---|---|
| Type | Name | 10 | 20 | 30 | 100 |
| Selection | DSA [Muh+18] | 1.18 | 2.78 | 7.22 | **70.12** |
| | GDSA [Muh+19] | 1.39 | 3.45 | 9.04 | |
| Shape | SW [Xia+15] | 5.56 | 13.51 | 18.95 | 23.17 |
| | PMN | 11.45 | 25.50 | 33.33 | 38.55 |
| Selection +Shape | SW+GDSA | 5.92 | 14.82 | 20.12 | 23.17 |
| | PMN+GDSA | **13.43** | **27.80** | **34.87** | 38.55 |

Table B.3: Classification accuracy on the Quickdraw345 dataset at budgets of 10%, 20% and 30% evaluated with a classifier trained on the original human-drawn sketches.

## B.5 Additional FG-SBIR Results

In the main paper we report the results of sketch-based image retrieval on top-10 accuracy, following previous works [Muh+18]. For completeness, Table B.4 shows the results for top-1 retrieval accuracy. With this metric, we observe the same trend in all datasets, with PMN+GDSA achieving the best results at low budgets.

Additionally, apart from the *Selection*-based and *Shape*-based abstraction methods discussed in the main paper, *On-the-Fly Fine-Grained Sketch Based Image Retrieval* (OTF) [Bhu+20] proposes a *Finetuning*-based approach that can be employed specifically for the FG-SBIR. Such a method does not learn to abstract, but finetunes the embedding network with partial sketches, optimizing the FG-SBIR ranking to better retrieve their respective images.

Results for OTF are added to Table B.4. Since shape-based abstraction is orthogonal to finetuning, we also evaluate the combination of SW/PMN with OTF for FG-SBIR. OTF generally performs well when there is a shift in the data distribution fed through the sketch embedding network. For instance, at 10% budget on ShoeV2 and ChairV2, OTF outperforms GDSA as finetuning the embedding works better than selecting more relevant parts of the sketch. This gap closes as we increase the budget, and at 30% GDSA already performs better than OTF on both datasets. Similarly, OTF boosts SW more than our PMN when combined, as the sketches reproduced by SW less accurately resemble the original sketches while the reconstructions of our model stay closer to the original data distribution.

| | | ShoeV2, Budget (%) | | | | ChairV2, Budget (%) | | | |
|---|---|---|---|---|---|---|---|---|---|
| Type | Name | 10 | 20 | 30 | 100 | 10 | 20 | 30 | 100 |
| Finetuning | OTF [Bhu+20] | 2.40 | 3.45 | 5.11 | **36.49** | 3.38 | 5.56 | 8.98 | **53.56** |
| Selection | DSA[Muh+18] | 1.35 | 2.40 | 4.05 | **36.49** | 2.48 | 6.19 | 10.22 | **53.56** |
| | GDSA [Muh+19] | 1.86 | 3.45 | 6.46 | | 2.79 | 7.43 | 12.07 | |
| Shape | SW [Xia+15] | 3.30 | 6.16 | 7.96 | 9.11 | 8.98 | 14.24 | 16.72 | 17.85 |
| | PMN | 6.76 | 16.07 | 18.17 | 20.04 | 16.41 | 31.89 | 35.91 | 37.53 |
| Shape +Finetuning | SW+OTF | 4.80 | 8.41 | 10.66 | 11.92 | 9.91 | 17.03 | 18.89 | 20.31 |
| | PMN+OTF | 9.16 | 17.17 | 18.92 | 20.77 | 16.72 | 31.89 | 35.15 | 38.04 |
| Shape +Selection | SW+GDSA | 4.20 | 7.21 | 8.56 | 9.11 | 9.29 | 14.86 | 17.03 | 17.85 |
| | PMN+GDSA | **9.61** | **17.37** | **19.22** | 20.0 | **20.74** | **33.75** | **36.84** | 37.53 |

Table B.4: Top-1 accuracy for FG-SBIR on ShoeV2 and ChairV2.

# C

# PUBLICATIONS

This thesis is based on the following publications. An overview of contributions can be found in Section 1.2. Asterisks (*) indicate shared first author publications.

R. Corona*, S. Alaniz*, Z. Akata. "Modeling Conceptual Understanding in Image Reference Games". In: *Advances in Neural Information Processing Systems 32 (NeurIPS), Spotlight*. 2019

S. Alaniz, D. Marcos, B. Schiele, Z. Akata, "Learning Decision Trees Recurrently Through Communication". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021

S. Alaniz, M. Mancini, D. Marcos, A. Dutta, Z. Akata. "Abstracting Sketches through Simple Primitives". In: *European Conference on Computer Vision (ECCV)*. 2022

S. Alaniz, M. Federici, Z. Akata. "Compositional Mixture Representations for Vision and Text". In: *CVPR Workshop on Learning with Limited Labelled Data for Image and Video Understanding*. 2022

S. Alaniz*, T. Hummel*, Z. Akata. "Semantic Image Synthesis with Semantically Coupled VQ-Model". In: *ICLR Workshop on Deep Generative Models for Highly Structured Data*. 2022

I contributed to the following publication, which is not part of this thesis.

Z. Akata, L.A. Hendricks, S. Alaniz, T. Darrell. "Generating Post-Hoc Rationales of Deep Visual Classification Decisions". In: *Explainable and Interpretable Models in Computer Vision and Machine Learning. The Springer Series on Challenges in Machine Learning*. 2018