TÉCNICO LISBOA

Visual Attention with Sparse and Continuous Transformations

António José Evaristo Farinhas

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. André Filipe Torres Martins
Prof. Pedro Manuel Quintas Aguiar

Examination Committee

Chairperson: Prof. Paulo Jorge Coelho Ramalho Oliveira
Supervisor: Prof. André Filipe Torres Martins
Member of the Committee: Prof. Alexandre José Malheiro Bernardino

January 2021

aos meus pais, por tudo

Acknowledgments

First of all, I would like to express my genuine appreciation to my incredible supervisors, André Martins and Pedro Aguiar, for their immeasurable guidance, motivation and support during all this time. Without their constant advice this work would not have been possible: they not only introduced me to this topic but also continuously presented me with new and exciting challenges. I can certainly say they are one of the best teams I have had the privilege of working with and I truly wish this is not the last time we work together.

During the last months, I was also fortunate to participate in the DeepSPIN/MAIA weekly meetings and get to know all the people involved. Even though I was more like a silent listener at first, I learned a lot from participating in their discussions.

My sincere thanks to my friends who are with me since I can remember. Thank you for all the talks, dinner parties and travels that made the last five years memorable, as they surely were. You make me understand what is really valuable in life.

I would also like to thank my loving family for their support and for sharing my accomplishments with the greatest enthusiasm I can imagine. In particular, I'm deeply grateful to my mother and my father who have always motivated me to learn and to my sister who is always interested in my work. Every accomplishment of mine, will always be yours.

Last but not least, I thank Tatiz for being on this journey with me. Everything seems easier by your side. Thank you for all the love and patience all these years.

This work was partially supported by the P2020 program MAIA (contract 045909).

Resumo

Os mecanismos de atenção visual são um componente importante das redes neuronais profundas com aplicação em visão computacional, permitindo-lhes identificar elementos relevantes em conjuntos finitos de objetos ou regiões. Para representar a pontuação indicativa da importância de cada *feature* no domínio probabilístico, estes mecanismos empregam uma função diferenciável – usualmente a função *softmax*, cujo resultado é estritamente denso, atribuindo probabilidade de massa a todos os elementos do conjunto. Esta densidade é, muitas vezes, um desperdício, porque não evita que *features* irrelevantes sejam consideradas, afetando negativamente a interpretabilidade dos modelos. Até agora, a atenção visual foi apenas aplicada a domínios discretos, o que pode levar a uma perda de foco, devido a uma dispersão excessiva da atenção sobre a imagem.

Nesta tese, exploramos alternativas de domínio contínuo aos modelos discretos, propondo soluções que se focam tanto na continuidade como na esparsidade das distribuições de atenção, sendo adequadas para selecionar regiões simultaneamente compactas e esparsas (*e.g.*, elipses). A primeira caraterística encoraja a seleção de regiões contínuas, enquanto a segunda permite destacar as *features* mais importantes, atribuindo uma probabilidade nula às partes irrelevantes. Utilizamos o facto de os Jacobianos destas transformações serem covariâncias generalizadas para derivar algoritmos de retropropagação eficientes, tanto para distribuições unimodais como multimodais. Experiências em *visual question answering* mostram que os nossos modelos contínuos permitem gerar mapas de atenção mais suaves (aparentemente mais próximos da perceção humana), conduzindo também a melhorias de precisão em relação a um modelo de base treinado com os mesmos dados.

Palavras-chave: aprendizagem profunda, mecanismos de atenção visual, transformações contínuas, esparsidade

Abstract

Visual attention mechanisms have become an important component of neural network models for Computer Vision applications, allowing them to attend to finite sets of objects or regions and identify relevant features. A key component of attention mechanisms is the differentiable transformation that maps scores representing the importance of each feature into probabilities. The usual choice is the softmax transformation, whose output is strictly dense, assigning a probability mass to every image feature. This density is wasteful, given that non-relevant features are still taken into consideration, making attention models less interpretable. Until now, visual attention has only been applied to discrete domains – this may lead to a lack of focus, where the attention distribution over the image is too scattered.

Inspired by the continuous nature of images, we explore continuous-domain alternatives to discrete attention models. We propose solutions that focus on both the continuity and the sparsity of attention distributions, being suitable for selecting compact and sparse regions such as ellipses. The former encourages the selected regions to be contiguous and the latter is able to single out the relevant features, assigning exactly zero probability to irrelevant parts. We use the fact that the Jacobian of these transformations are generalized covariances to derive efficient backpropagation algorithms for both unimodal and multimodal attention distributions. Experiments on Visual Question Answering show that continuous attention models generate smooth attention maps that seem to better relate with human judgment, while achieving improvements in terms of accuracy over grid-based methods trained on the same data.

Keywords: deep learning, visual attention mechanisms, continuous transformations, sparsity

Contents

List of Tables

List of Figures

List of Algorithms

Acronyms

AIC Akaike Information Criterion.

BIC Bayesian Information Criterion.

CNN Convolutional Neural Network.

CV Computer Vision.

DL Deep Learning.

EM Expectation Maximization.

GMM Gaussian Mixture Model.

MCA Multimodal Continuous Attention.

MDL Minimum Description Length.

ML Machine Learning.

NLP Natural Language Processing.

RBF Radial Basis Function.

RNN Recurrent Neural Network.

VQA Visual Question Answering.

Chapter 1

Introduction

1.1 Motivation

The human visual system selectively attends to the most relevant parts of visual stimuli for quick perception, being able to process large amounts of visual information in parallel: it becomes possible to interpret complex scenes in real time, despite the limited processing resources [1, 2]. Different ways to model human visual attention have long been studied in the Computer Vision (CV) and Machine Learning (ML) communities, trying to generate more biologically inspired results and liberating resources to focus on more important parts of the input [3–6].

In the last few years, the increased amount of available training data and the development of more powerful computer hardware and software has led to the rise of Deep Learning (DL). An important architectural innovation, **attention mechanisms**, started to be integrated in neural networks, first in Natural Language Processing [7, 8] and then in Computer Vision (*e.g.*, image captioning [9], visual question answering [10], action recognition [11] and salient object detection [12]). Visual attention mechanisms appear as a way to mimic the human visual system – they can automatically learn the relevance of any element of the input by generating a set of weights and take them into account while performing the proposed task. Furthermore, since most commonly used neural network architectures are very complex, they remain black-box models: humans cannot easily understand their inner decision making process. In addition to boosting the performance of a model, attention mechanisms can partially provide insights into the model's reasoning behind its prediction [13, 14]. For both language and vision applications, the visualization of attention weights can help us analyze the outputs of a neural network and possibly understand some unpredictable outcomes [15].

In the context of Aerospace Engineering, visual attention mechanisms have been used for a wide range of tasks such as improving the performance and robustness of off-road robots [16], for tracking objects with unmanned aerial vehicles [17], in Earth Observation [18–20], among others. Specially in what concerns human computer interaction, where real applications of DL take place in safety-critical fields, intelligent agents are likely to be implemented and asked to perform autonomous vision-based tasks such as navigation, aerial mapping and object delivery [21–23].

Despite recent improvements in the quality of attention-based vision systems, there are limitations that motivate current work. In this thesis, we focus on the following ones:

- In state of the art models for many vision tasks, images are represented using *bounding box* features [24]: an object detector trained on a dataset annotated with ground truth bounding boxes is used to acquire those features, making the process computationally expensive, time consuming and dependent on external resources, becoming less suitable for practical and real-time applications.

- A key component of visual attention mechanisms is the differentiable transformation that maps scores representing the importance of each feature into probabilities. The usual choice is the softmax transformation, whose output is **strictly dense**, assigning a probability mass to **every image feature**. This density is wasteful, given that non-relevant features are still taken into consideration, making attention models less interpretable [25].

- Although image data is naturally continuous, visual attention mechanisms have only been applied to **discrete domains**, meaning that the input object is usually split into a finite set of pieces (*e.g.*, an image is split into a set of regions or pixels) and its continuity is not taken into account by attention models. In certain applications this may lead to a **lack of focus**, where the attention distribution over the image is too scattered.

In this thesis, we address the limitations above by considering continuous-domain alternatives to discrete attention models. We construct 2D continuous attention mechanisms that are able to increase focus on relevant image regions, leading to more interpretable predictions. Our solutions focus on both the **continuity** and the **sparsity** of attention distributions, being able to select compact and sparse regions in images. The first takes adjacency into account and encourages the selected regions to be contiguous, that is, to have a compact structure. The second is able to single out the relevant features, assigning exactly zero probability to irrelevant regions.

1.2 From discrete to continuous attention in 2D

Currently, a neural network with visual attention attends to finite sets of objects or regions and identifies relevant features (we refer to this as discrete attention, hereinafter). Alternatively, inspired by the continuous nature of images, we propose continuous attention mechanisms. These mechanisms attend to inherently continuous domains and are suitable for selecting compact regions such as ellipses. Furthermore, we introduce multimodal continuous attention mechanisms that efficiently model more complex attention distributions.

Discrete attention. Usual discrete attention mechanisms for images work as follows. Assume an input image split in L pieces: an image represented by L grid-level or object-level features. A discrete attention mechanism computes a *score vector* in which high scores should correspond to more relevant parts of

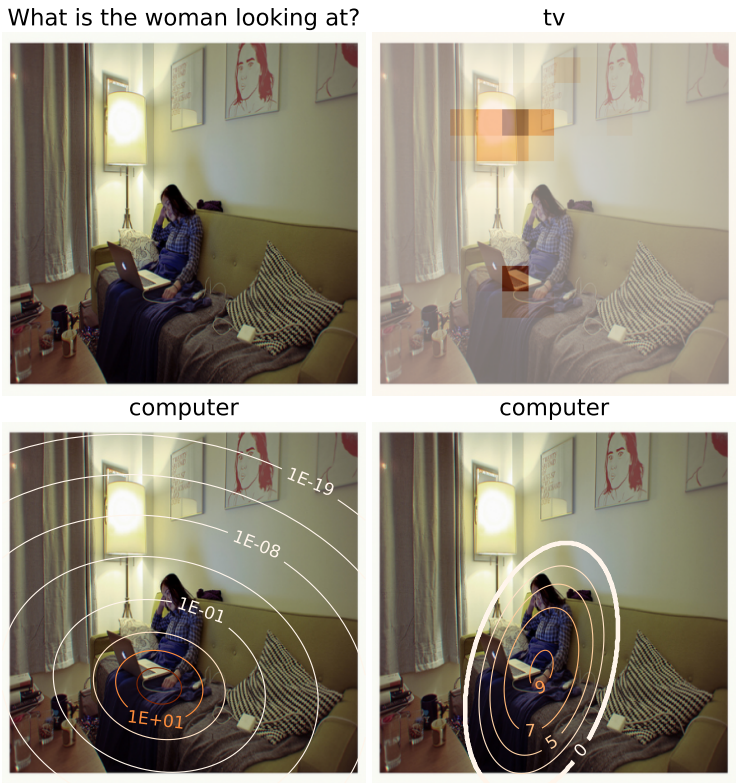

Figure 1.1: Examples of attention maps for VQA. Top left: original image. Top right: state-of-the-art with discrete attention. In the bottom, the results of our methods: 2D continuous softmax (left) and 2D continuous sparsemax (right, where the outer ellipse encloses all probability mass). The answers provided by each model are presented above the corresponding image.

the input. Then, a discrete transformation is applied to the score vector to produce a probability vector – the *attention weights*; softmax is the usual choice for this transformation, assigning a probability mass to every image feature. Finally, the probability vector is used to compute a weighted average of the input (known as the *context vector*), that is used to produce the network's decision.

Continuous attention. Instead of assuming a finite set, we assume a continuous measure space such as the $\mathbb{R}^2$ plane and represent the image as a continuous function. Then, a *score function* is mapped to a probability vector by using an extension of *regularized prediction maps*, originally defined on finite domains by Blondel et al. [26], to arbitrary measure spaces [27]. By choosing a regularization function based on the Tsallis α-entropies [28], we construct 2D continuous α-entmax attention mechanisms. For $\alpha \in \{1, 2\}$ this is called 2D continuous softmax and 2D continuous sparsemax attention, respectively. The attention density corresponds to a bivariate Gaussian, in the first case, and to a truncated paraboloid distribution, whose support corresponds to an ellipse, in the second.

Attention visualization. Figure 1.1 shows an example of the attention maps generated by different attention models for Visual Question Answering (VQA), whose goal is to provide an answer, given an image and a question about it. In the discrete model, the attention is too scattered, possibly mistaking the lamp with a TV screen; contrarily, our continuous attention models focus on the right region and answer the question correctly, with 2D continuous sparsemax enclosing all the relevant information in its supporting ellipse.

Multimodal continuous attention. As a natural follow-up, we propose multimodal continuous attention by using mixtures of unimodal attention densities. These mechanisms are able to generate more flexible attention maps (*e.g.*, attention distributions with multiple *peaks*), while enjoying some of the properties of their unimodal counterparts.

1.3 Contributions

In this thesis, we present **continuous** alternatives to the usual visual attention mechanisms, some of which being simultaneously **sparse**. Our transformations are able to improve focus on the relevant image regions, while providing higher interpretability. We identify the main contributions of this thesis as follows:

1. We propose a framework for using continuous attention with image data, in which the parameters of continuous attention densities are obtained from discrete attention weights using *moment matching*. This is done for both softmax and sparsemax attention.

2. We derive efficient algorithms for the evaluation and gradient computation of 2D α-entmax continuous attention mechanisms, for $\alpha \in \{1, 2\}$. For $\alpha = 1$, we obtain closed-form expressions valid for any number of dimensions (including the 2D case). For $\alpha = 2$, we reduce the problem to a univariate integral, easy to compute numerically.

3. We introduce novel multimodal continuous attention mechanisms: we generalize continuous attention to multimodal distributions by using mixtures of unimodal attention densities. We parametrize the attention density as a mixture of Gaussians, adapting the Expectation Maximization algorithm to deal with weighted data (*e.g.*, discrete attention weights) by changing the way the parameters of the mixture are estimated and computing a weighted likelihood function. Even though we focus on the 2D case, this approach is also valid for 1D data (*e.g.*, speech or text segments).

4. We plug our 2D continuous attention mechanisms in a VQA attention-based model in order to improve focus and possibly provide better explanations via smoother attention maps. In terms of accuracy, we obtain small improvements over grid-based methods trained on the same data. Moreover, we open-sourced our pytorch implementation; our code is available at https://github.com/antonio-farinhas/mcan-vqa-continuous-attention, along with additional information on how to reproduce our experimental results.

Part of this work was published as a spotlight paper at the Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS 2020) [27] and a journal paper is to be submitted [29].

1.4 Thesis Outline

This thesis is organized as follows. In Chapter 2, we start by reviewing the Machine Learning concepts that our work builds upon; then, we explain how attention mechanisms have become a crucial component in current Deep Learning models, first, in NLP and then in CV applications. We focus on fully-differentiable attention mechanisms that can be plugged in neural networks and trained end-to-end. In Chapter 3, we explain how these attention mechanisms can be generalized to continuous spaces, focusing on naturally continuous domains such as image data.

In Chapter 4, we derive efficient algorithms for the evaluation and gradient computation of 2D α-entmax continuous attention mechanisms, for $\alpha \in \{1, 2\}$, with Gaussian RBFs. The reader is referred to Appendices A.1 and A.2 for detailed derivations when $\alpha = 2$. In Chapter 5, we construct multimodal continuous attention mechanisms with mixtures of attention densities as a natural follow-up of the previous chapter. We point out the difference between these novel multimodal continuous attention mechanisms and the usual multi-head attention for discrete domains. Finally, in Chapter 6, we describe in detail the experiments in VQA and present attention maps generated by our continuous transformations (see Appendix B.1 for details about our computing infrastructure).

Chapter 2

Background

In this chapter we discuss key topics that will be used throughout this work. We begin by reviewing the Machine Learning concepts that are the foundations of our work, in Section 2.1. This is not intended to be an extensive explanation of recent neural network architectures but only to serve as an overview of some topics our work rely on. Then, we explain the reason why attention mechanisms are important for both language and vision applications in current Deep Learning models, in Section 2.2. We pave the way for developing continuous attentions mechanisms by presenting an overview of popular methods and using them to motivate our own approach.

2.1 Machine Learning

Supervised Learning. Consider an input x from a set of possible inputs $\mathcal{X}$ and an output $y \in \mathcal{Y}$. In supervised learning problems we assume we are provided a *dataset* containing N training inputs paired with its correspondent outputs,

$$\mathcal{D}_{\mathcal{N}} = \{(x_1, y_1), ..., (x_N, y_N)\} \subseteq \mathcal{X} \times \mathcal{Y}. \tag{2.1}$$

The goal is to maximize some performance measure defined with respect to the test set, based on training data. We use the training data to learn a *classifier* $f : \mathcal{X} \to \mathcal{Y}$ that generalizes well to arbitrary inputs. To do so, we can define a parametrized prediction model as a mapping $f(x; \theta)$ from a given input x to an output y by learning the value of the parameters θ that lead to the best fit. At test time, we predict $\hat{y} = f(x)$ that should be as similar to y as possible.

Since we just have access to a finite training set of examples, we don't know the true underlying distribution of the data p_{data}. To solve this as an optimization problem we can represent the discrepancy between the predicted and the actual outputs by means of a loss function L and try to find the parameters θ that minimize the value of L evaluated on the training set. This is known as *empirical risk minimization*,

$$\text{minimize} \quad \frac{1}{N} \sum_{i=1}^{N} L(f(x_i; \theta), y_i), \tag{2.2}$$

and should not be confused with the risk minimization in optimization problems, where p_{data} is known. The choice of the loss function L in (2.2) is an important aspect when training Machine Learning models and it depends on the problem we are considering. For instance, for multiclass classification we can use the principle of *maximum likelihood* and choose the *cross-entropy* between the training data and the model's predictions as the loss function, *i.e.*, consider a loss function that is simply the negative log-likelihood function, given by

$$L(f(x;\theta),y) = -\sum_{i=1}^{N} y_i \log(f(x_i;\theta)). \tag{2.3}$$

Deep Learning. Deep Learning (DL) is a very powerful tool for supervised learning. Deep models appeared as an alternative to linear models, being able to deal efficiently with nonlinear data by learning complex functions through multiple layers of computation. Recall that a linear model outputs y through an affine transformation

$$f_{\text{linear}}(x;w,b) = x^\top w + b, \tag{2.4}$$

where w is a vector of weights and b is a bias term. We can choose an appropriate loss function and minimize it with respect to the parameters w and b. To overcome the limitations of linear models, we can consider a classic DL model – a deep feedforward network – by choosing a nonlinear transformation or a mapping ϕ and, instead of applying the linear model to x itself, applying it to a transformed input $\phi(x)$. For a feedforward neural network with one *hidden layer* defined by ϕ, we can consider a model

$$f_{\text{nonlinear}}(x;\theta,w) = \phi(x;\theta)^\top w, \tag{2.5}$$

where θ are parameters used to learn ϕ from a broad class of functions and w are parameters that map $\phi(x)$ to the desired output. In this case, the representation is parametrized as $\phi(x,\theta)$ and the optimization algorithm is used to find a θ that leads to a good representation. By adding more hidden layers or more units within a layer, a deep network can represent functions of increasing complexity.

Deep models are not limited to deep feedforward neural networks where the information flows from x to y, passing through the intermediate computations of f, without feedback connections from y to x. In fact, when these networks include feedback connections, they originate the so called Recurrent Neural Networks (RNNs) that are commonly used in natural language applications. They can also have a known grid-like topology, leading to Convolutional Neural Networks (CNNs) that are extremely successful in computer vision (Goodfellow et al. [30]).

Backpropagation. Optimizing Equation (2.2) for DL models, *i.e.*, update the network's parameters, can be a difficult task. A usual approach is to use stochastic gradient-based optimization algorithms that require the computation of the gradient of the loss function with respect to the network's parameters,

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \mathcal{L}_i(\theta), \tag{2.6}$$

where $\mathcal{L}_i(\theta) = L(f(x_i;\theta), y_i)$ in the simpler unregularized case.[1] This can be done using the gradient backpropagation algorithm [32], also known as *backprop*.[2] Backpropagation can be used to evaluate (2.6) at a given point, using the chain rule of calculus by performing Jacobian-gradient products, *i.e.*, multiplying the Jacobian matrix by the gradient, for each node in the graph.

2.2 Attention mechanisms

Based on the idea that we need to *attend to* a certain part of a large input (*e.g.*, some words in a sentence or regions in an image) when processing it, attention mechanisms became one of the most powerful concepts in the Deep Learning field. Each element composing the input may have different relevance to the task we are solving: for instance, in machine translation, each word in the source sentence can be more or less relevant for translating the next word; in image captioning, the background regions of an image can be irrelevant to describe an object but crucial to characterize the landscape. To solve this problem, the prevailing solution consists in using *attention mechanisms* by automatically learning the relevance of any element of the input, *i.e.*, by generating a set of weights (one per element of the input) and take them into account while performing the proposed task.

2.2.1 Sequence-to-sequence learning

Consider a sequence-to-sequence task, *e.g.*, neural machine translation, where the goal is to find the output sequence y that maximizes the conditional probability of y given a input sequence x and note that it is unmanageable to enumerate all y to maximize $p(y|x)$. Generally, these problems are performed using an encoder-decoder architecture: the encoder takes a variable-length sequence, converts it to an intermediate fixed-length vector representation and then passes it to the decoder that produces the output variable-length sequence. The two components of these models are jointly trained to maximize the likelihood of generating the correct output sequence, usually using a gradient-based algorithm to estimate the model parameters, as discussed in Section 2.1.

The simplest approach is to consider that both the encoder and the decoder are RNNs [34]. The encoder converts the input sentence into a context vector c,

$$c = q(\{h_1, ..., h_{T_x}\}), \tag{2.7}$$

where the hidden state at time t is given by $h_t = f(h_{t-1}, x_t) \in \mathbb{R}^n$ and both q and f are nonlinear functions. The decoder is trained to predict the next symbol $y_{t'}$ given the context c and all the previously

[1] Since some complex deep models are able to "memorize" the training set, a slightly different approach is usually used in DL. Different techniques (*e.g.* regularization) can be used to reduce the so called *overfitting*. See Goodfellow et al. [30, Chapter 7] or Bishop [31, Chapter 5] for an overview of these methods.

[2] Although this term only refers to the procedure for computing the gradients, it is often confused as the learning algorithm itself, *e.g.* the classic stochastic gradient descent or Adam optimizer [33].

predicted symbols $\{y_1, ..., y_{t'-1}\}$, defining a probability over $y = (y_1, ..., y_{t'-1})$,

$$p(y) = \prod_{t=1}^{T} p(y_t | \{y_1, ..., y_{t'-1}\}, c), \tag{2.8}$$

with

$$p(y_t | \{y_1, ..., y_{t'-1}\}, c) = g(y_{t-1}, s_t, c), \tag{2.9}$$

where g is a nonlinear function that must produce valid probabilities and s_t is the hidden state of the RNN.

One problem with this approach is that of compressing all the necessary information of an input sequence into a fixed-length vector (specially difficult when dealing with large sequences). Bahdanau et al. [7] addresses this problem introducing an attention mechanism that develops a context vector that is filtered in a proper way for each time step to improve the information the decoder has available. Now, each conditional probability in (2.8) depends on a different context vector c_i (unlike the first approach with a single c),

$$p(y_i | \{y_1, ..., y_{i-1}\}, x) = g(y_{i-1}, s_i, c_i), \tag{2.10}$$

where $s_i = f(s_{i-1}, y_{i-1}, c_i)$. The context vector c_i is obtained as a weighted sum of the hidden states of the encoder:

$$c_i = \sum_{j=1}^{L} \alpha_{ij} h_j, \tag{2.11}$$

where L is the number of symbols of the input sequence and α is a normalized vector of attention scores that can be seen as probability distribution over the input sequence - whose values reflect the importance of h_j with respect to the previous hidden state s_{i-1} in deciding the next state s_i and the output y_i - and that can be computed, for instance, as:

$$\alpha_{ij} = \text{softmax}(e) = \frac{\exp(e_{ij})}{\sum_{k=1}^{L} \exp(e_{ik})}, \tag{2.12}$$

where $e_{ij} = a(s_{i-1}, h_j)$ is an **alignment model** that measures how well the inputs around position j and the output at i match and that can be defined in multiple ways. For instance, Bahdanau et al. [7] parametrizes the alignment model as a feedforward neural network which is jointly trained with all the other components of the system. Other approaches will be discussed in Section 2.2.3.

2.2.2 The Transformer

The transformer was first introduced by Vaswani et al. [8] as an attention mechanism based alternative to the usual models for sequence-to-sequence tasks that use the encoder-decoder architecture introduced before (possibly connected through an attention mechanism), dispensing with recurrence and convolutions at all. This model builds upon the concepts of scaled dot-product attention, multi-head attention and self-attention, which we now explain. See Figure 2.1 to understand how these modules are connected in this architecture.

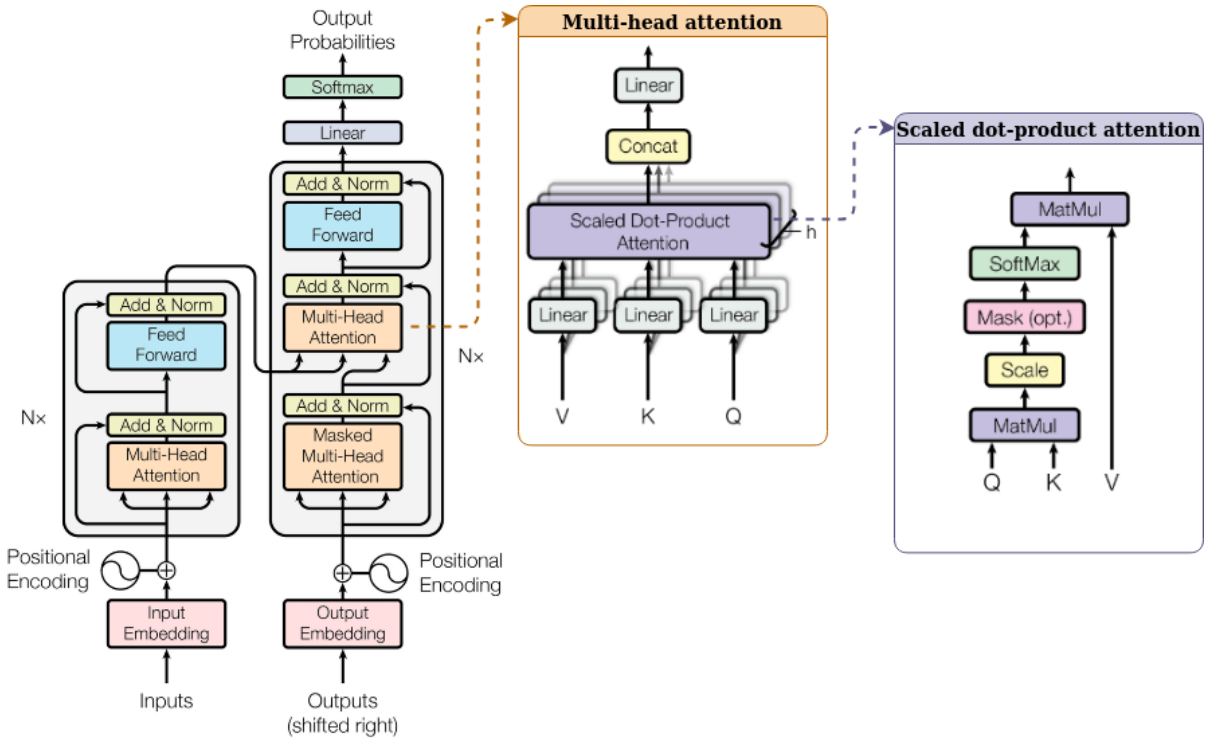

Figure 2.1: The transformer: model architecture including scaled dot-product and multi-head attention. Adapted from Vaswani et al. [8, Figure 1 and 2]

Scaled dot-product attention. Consider a set of key-values pairs (K, V) and a query Q, where both K, V and Q are vectors[3]. In general, an attention function can be defined as a mapping from Q and (K, V) to an output as a weighted sum of V, where the weight associated with each V depends on both Q and K. It can be computed as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{2.13}$$

where d_k is the dimension of the queries and the keys. Note that this construction is the same as the one presented in Subsection 2.2.1 – see Expression (2.11) and (2.12) – considering that $K = V = h_t$ and $Q = s_{i-1}$.

Multi-head attention. As the name suggests, multi-head attention consists in obtaining a different representation of (Q, K, V) per head h in parallel, computing scaled dot-product attention (2.13) for each representation, concatenating the results and projecting them through a feedforward layer. Formally, we can write:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \tag{2.14}$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O, \tag{2.15}$$

[3]In practice, both K, V and Q can be packed as matrices so that it is possible to compute the attention function on a set of queries simultaneously.

where the projections W_i and W^O are parameter matrices. Unlike with a single attention head, with multi-head attention the model is able to attend to information from different representations, meaning that the ability of the model to focus on different positions is expanded.

Self-attention. Self-attention is an attention mechanism that consists in relating different positions in the same sequence, regardless of their relative distance, in order to compute a representation of that sequence. An intuitive way of thinking of self-attention is as some inputs interacting with each other and learning what parts are more relevant or, in other words, what they should pay more attention to. In practice, this is achieved by considering that both Q, K and V in Expression (2.13) are equal. See, for instance, the masked multi-head attention block in Figure 2.1.

Architecture. The overall architecture of the transformer is similar to the ones described in Subsection 2.2.1 with the difference that it is composed of a stack of $N = 6$ encoder and decoder layers. Each encoder has 2 sub-layers: a multi-head self-attention mechanism as described in the last paragraph, followed by a simple feedforward neural network, introduced in subsection 2.1. Each decoder is composed by 3 sub-layers: the first and the last are identical to the ones in the encoder; the second sub-layer consists of a multi-head attention over the output of the encoder stack.

2.2.3 Attention mechanisms in Natural Language Processing

In subsections 2.2.1 and 2.2.2 we introduced attention mechanisms using two different formulations. In the former, we followed Bahdanau et al. [7] approach and explained how to introduce attention mechanisms in an encoder-decoder architecture; in the latter, attention was described as a mapping from query vectors to output vectors via a mapping table of key-value pairs [8]. In fact, both formulations are equivalent and materialize the same principle: from (2.11) we can obtain a context vector as a weighted sum of values $c_i = \sum_{j=1}^{L} \alpha_{ij} h_j$, where $\sum_{j=1}^{L} \alpha_{ij} = 1$: α_{ij} is a probability vector (attention scores/weights) and h_j is the *value* that in the first formulation corresponds to the encoder hidden state. By going back to the previous subsections it is possible to note that the only difference in the two approaches lies in how α_{ij} is computed. Different lines of work introduced distinct attention mechanisms by choosing different alignment models or using different distribution functions.

Choosing the alignment model. The alignment model defines how keys and queries are matched or combined. Table 2.1 contains a summary of several popular attention mechanisms in NLP and corresponding alignment models.

Choosing the distribution function. To obtain normalized attention weights from the computed alignment scores we need to choose a distribution function. The most common choice is the *softmax* normalizing transform, leading to dense alignments and strictly positive output probabilities, *i.e.*, a non-zero attention weight is assigned to all the parts of the input. Arguably, some parts might be completely

ATTENTION	ALIGNMENT MODEL	REFERENCE
Additive	$v_a^\top \tanh(W_a s_{i-1} + U_a h)$	Bahdanau et al. [7]
Dot-product	$h^\top s_{i-1}$	Luong et al. [35]
General	$h^\top W_a \ s_{i-1}$	Luong et al. [35]
Scaled dot-product	$(1/\sqrt{d}) \ h^\top s_{i-1}$	Vaswani et al. [8]

Table 2.1: Summary table of different attention mechanisms and correspondent alignment models. Note that v_a, W_a and U_a are weight matrices and d is a constant that corresponds to the dimension of the queries and the keys. We simplified the notation by writing h instead of h_j, as in Subsection 2.2.1.

irrelevant for the matter. This problem has been recently addressed by choosing sparse normalizing transforms such as *sparsemax* [36],

$$\text{sparsemax}(z) = \underset{p \in \triangle^L}{\arg\min} ||p - z||^2, \tag{2.16}$$

where $\triangle^L$ is the L-dimensional probability simplex or α-*entmax* [37] as the distribution function - this allows us to produce sparse alignments and to assign exactly zero probability to some of the possible outputs.

2.2.4 Visual attention mechanisms in Deep Learning

Although attention mechanisms were first introduced in NLP for machine translation [7], previous work by Larochelle and Hinton [3] in Computer Vision had already proposed an object recognition model that learns where to look from scratch using *glimpses*, inspired by the idea that biological vision systems need to sequentially fixate relevant parts for a specified task because their retina has a limited resolution that falls very quickly with eccentricity. Later, when the work by Mnih et al. [4] - a novel neural network model capable of processing only a selected sequence of regions in an image or video - outperformed the state of the art in both static vision tasks (*e.g.,* image classification) and dynamic visual environments (*e.g.*, object tracking), visual attention mechanisms gained popularity. Nowadays, this is a trending research topic and alternatives to these *hard* attention mechanisms are constantly being presented, including *soft* and, more recently, *sparse* visual attention approaches.

Hard vs soft attention. *Hard attention mechanisms* [4–6] rely on a controller to select the relevant parts of the input; mechanisms like these are not differentiable end-to-end and, for that reason, cannot be trained with the standard backpropagation algorithm - instead, they require the use of *reinforcement learning* techniques. Although these models perform well on simple datasets, it has been difficult to use them in real-world applications [38]. To make training easier, *soft attention mechanisms* commonly used for natural language tasks [7, 8, 35] were introduced in tasks that also require vision (*e.g.*, image captioning [9] and visual question answering [10]). These mechanisms are fully differentiable: they can be plugged in neural networks and trained end-to-end with gradient backpropagation algorithm. Whereas models that use hard attention make decisions on only a subset of pixels in the input image,

with soft attention the model has access to the entire image but certain areas may be more attended than others.

Sparse and structured visual attention. Soft attention mechanisms can be used to encourage models to look to the most relevant part of an input image. To enable end-to-end training with gradient backpropagation they require a differentiable mapping from scores $z \in \mathbb{R}^L$ (that represent the importance of each feature) to a probability distribution $p \in \triangle^L$, where $\triangle^L$ is the L-dimensional probability simplex. As explored in Subsection 2.2.3, the usual choice is the softmax transformation, whose output is strictly dense, assigning a probability mass to every image feature. Very recent work by Martins et al. [25] addressed this problem by introducing *selective visual attention mechanisms*: sparse alternatives that are able to select only the relevant features. A first approach consists in replacing softmax with sparsemax [36]. Then, in order to select contiguous regions of an image and to encourage the weights of related adjacent spatial locations to be the same, they introduce *Total-Variation Sparse Attention* (TVMAX). These alternatives allow us to select sparse and compact regions in images; they can better relate to human attention, conducing to higher interpretability.

Chapter 3

Continuous attention mechanisms

In this chapter, we provide an overview of the work done on continuous attention mechanisms. Although its definition is fully general concerning the dimension of the input, continuous attention had only been applied to text data before this work. We begin by introducing Ω-regularized prediction maps (Ω-RPM), in Section 3.1. We then discuss possible choices of regularization functions that lead to different Ω-RPM, in Section 3.2. Finally, we use the concepts introduced in the previous sections to explain how to construct continuous attention mechanisms, in Section 3.3, focusing on naturally continuous domains such as image data.

Notation. Consider a measure space $(S, \mathcal{A}, \nu)$, where S is an underlying set, $\mathcal{A}$ is a σ-algebra on S and ν is a measure on $(S, \mathcal{A})$. We denote the set of ν-absolutely continuous probability measures as $\mathcal{M}_+^1(S)$. From the Radon-Nikodym theorem [39, §31], each element of $\mathcal{M}_+^1(S)$ is identified (up to equivalence within measure zero) with a probability density function $p : S \to \mathbb{R}_+$, with $\int_S p(t)\, d\nu(t) = 1$. We refer to the support of a density $p \in \mathcal{M}_+^1(S)$ as $\operatorname{supp}(p) = \{t \in S \mid p(t) > 0\}$. Moreover, we denote as $\triangle^{|S|}$ the $|S|$-dimensional probability simplex. For some $A \in \mathcal{A}$, when the base measure is clear enough from the context, we often drop $d\nu(t)$ from $\int_A p(t)\, d\nu(t)$, simply writing $\int_A p(t)$ and we denote the measure of A as $|A| = \nu(A) = \int_A 1$. Furthermore, given functions $\phi : S \to \mathbb{R}^m$ and $\psi : S \to \mathbb{R}^n$, we write expectations and covariances as $\mathbb{E}_p[\phi(t)] := \int_S p(t)\, \phi(t)$ and $\operatorname{cov}_p(\phi(t), \psi(t)) := \mathbb{E}_p[\phi(t)\psi(t)^\top] - \mathbb{E}_p[\phi(t)]\, \mathbb{E}_p[\psi(t)]^\top$. Finally, we refer to $\max\{a, 0\}$ as $[a]_+$.

3.1 Regularized prediction maps

The work by Blondel et al. [40] introduced Ω-regularized prediction maps for finite domains. Consider an input vector $x \in \mathcal{X}$ and a parametrized model $f : \mathcal{X} \to \mathbb{R}^{|S|}$, producing a score vector $\theta = f(x) \in \mathbb{R}^{|S|}$. For instance, θ can be label scores computed by a neural network model, f. A particular case of this general framework that is very relevant for our work consists in assuming a regularization function Ω, with $\operatorname{dom}(\Omega) \subseteq \triangle^{|S|}$. In this case, this framework allows us to map vectors $\theta \in \mathbb{R}^{|S|}$ into probability

vectors in the simplex $\triangle^{|S|}$. The prediction function can be written as

$$\hat{y}_\Omega(\theta) \in \underset{p \in \triangle^{|S|}}{\arg\max} \langle \theta, p \rangle - \Omega(p), \tag{3.1}$$

where p is a discrete probability distribution. Intuitively, the first term captures the affinity between x and y and the latter is a confidence term that encourages uniform distributions. Furthermore, when $\mathrm{dom}(\Omega)$ is the probability simplex, generalized negative entropies are an important class of Ω; in fact, a natural choice of regularization function is $\Omega = -\mathrm{H}$, where H is a generalized entropy, a function used to measure the *uncertainty* in a probability distribution [26]. Particular choices of H recover well-known transformations such as argmax, softmax, sparsemax [36] and others.

Martins et al. [27] extended the Ω-regularized prediction maps defined as in (3.1) to arbitrary measure spaces $\mathcal{M}_+^1(S)$, assuming that $\Omega : \mathcal{M}_+^1(S) \to \mathbb{R}$ is a lower semicontinuous, proper and strictly convex function. The **Ω-regularized prediction map** (Ω-RPM) $\hat{p}_\Omega : \mathcal{F} \to \mathcal{M}_+^1(S)$ is defined as

$$\hat{p}_\Omega[f] = \underset{p \in \mathcal{M}_+^1(S)}{\arg\max}\, \mathbb{E}_p[f(t)] - \Omega(p), \tag{3.2}$$

where $\mathcal{F}$ is the set of functions for which the maximizer exists and is unique. Again, the regularizer Ω in (3.2) can be chosen in order to recover transformations such as softmax and sparsemax, when S is finite. For the case where S is continuous, more interesting examples of regularizational functionals are shown in Section 3.2.

3.2 Choosing the regularization function Ω

The choice of the regularization function Ω is very important and leads to different regularized prediction maps – depending on its properties, it can lead to distributions with fixed support within the same family (*e.g.* distributions in the exponential family) (§ 3.2.1) or to alternatives with varying and sparse support, assigning zero probability mass to some entries (§ 3.2.2). In this section, similarly to Blondel et al. [40], we consider regularization functions in the class of the generalized negative entropies.

Tsallis [28] proposed a generalization of the well known Shannon's negentropy $\int_s p(t) \log p(t)$, using the notions of **β-exponential** and **β-logarithm** [41]. Suppose that β is a fixed positive number; the β-exponential function is defined as

$$\exp_\beta(u) = \begin{cases} [1 + (1-\beta)u]_+^{1/(1-\beta)}, & \beta \neq 1 \\ \exp u, & \beta = 1 \end{cases} \tag{3.3}$$

and the β-logarithm function as

$$\log_\beta(u) = \begin{cases} \frac{u^{1-\beta}-1}{1-\beta}, & \beta \neq 1 \\ \log u, & \beta = 1 \end{cases}. \tag{3.4}$$

Considering the limit $\beta \to 1$, the β-exponential and the β-logarithm recover the standard exponential and

the standard logarithm, respectively. Note that the $\exp_\beta : \mathbb{R} \to \mathbb{R}$ is the inverse of the $\log_\beta : \mathbb{R}_{\geq 0} \to \mathbb{R}$.

The α-**Tsallis negentropy** is then defined as

$$\Omega_\alpha(p) = \begin{cases} \frac{1}{\alpha(\alpha-1)} \left(\int_S p(t)^\alpha - 1 \right), & \alpha \neq 1 \\ \int_S p(t) \log p(t), & \alpha = 1 \end{cases}, \tag{3.5}$$

where $\Omega_1(p)$ is the Shannon's negentropy and $\Omega_2(p) = \frac{1}{2} \int_S p(t)^2 - \frac{1}{2}$ is known as Gini-Simpson index [42]. These 2 choices of α are of particular interest in this thesis.

Let $\alpha > 0$ and $f \in \mathcal{F}$. Martins et al. [27, Proposition 1] shows that the Ω_α-RPM in (3.2) can be simply written as

$$\hat{p}_{\Omega_\alpha}[f](t) = \exp_{2-\alpha}(f(t) - A_\alpha(f)), \tag{3.6}$$

where $A_\alpha : \mathcal{F} \to \mathbb{R}$ is a normalizing function,

$$A_\alpha(f) = \frac{\frac{1}{1-\alpha} + \int_S p_\theta(t)^{2-\alpha} f(t)}{\int_S p_\theta(t)^{2-\alpha}} - \frac{1}{1-\alpha}. \tag{3.7}$$

Furthermore, it is possible to show (see Martins et al. [27, Proposition 2] or Amari and Ohara [41, Theorem 5] for a proof) that the normalizing function, A_α (3.7), is a convex function and its gradient is given by

$$\nabla_\theta A_\alpha(\theta) = \mathbb{E}_{\tilde{p}_\theta^{2-\alpha}}[\phi(t)] = \frac{\int_S p_\theta(t)^{2-\alpha} \phi(t)}{\int_S p_\theta(t)^{2-\alpha}}, \tag{3.8}$$

where $\tilde{p}^\beta(t)$ is the β-escort distribution [28]:

$$\tilde{p}^\beta(t) = \frac{p(t)^\beta}{\|p\|_\beta^\beta}, \qquad \text{with } \|p\|_\beta^\beta = \int_S p(t')^\beta d\nu(t') \text{ and } \tilde{p}^1(t) = p(t). \tag{3.9}$$

The expression for the gradient of A_α (3.8) will be used in the Chapter 4 to compute the Jacobian of continuous α-entmax attention mechanisms.

3.2.1 Shannon's negentropy and Ω_1-RPM

For $\alpha = 1$, $\Omega_1(p)$ is the Shannon's negentropy and the corresponding Ω_1-RPM is given by

$$\hat{p}_{\Omega_1}[f](t) = \frac{\exp f(t)}{\int_S \exp(f(t'))d\nu(t')} = \exp(f(t) - A(f)), \tag{3.10}$$

where $A(f) = \log \int_S \exp f(t)$ is the log-partition function (see Martins et al. [27, App. A] for a proof). If S is finite and ν is the *counting measure*, we can write f as a vector in $\mathbb{R}^{|S|}$ and the Ω_1-RPM recovers the softmax transformation,

$$\hat{p}_{\Omega_1}[f] = \text{softmax}(f) = \frac{\exp(f)}{\sum_{k=1}^{|S|} \exp(f_k)} \in \triangle^{|S|}. \tag{3.11}$$

For continuous domains with $S = \mathbb{R}^N$, ν the *Lebesgue measure*, $\mu \in \mathbb{R}^N$, $\Sigma \in \mathbb{R}^{N\times N} \succ 0$ and choosing $f(t) = -\frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu)$, the Ω_1-RPM transformation is a multivariate Gaussian,

$$\hat{p}_{\Omega_1}[f] = \mathcal{N}(t;\mu,\Sigma) = \frac{1}{2\pi\,|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu)\right). \tag{3.12}$$

In particular, this becomes a univariate or a bivariate Gaussian if $N \in \{1, 2\}$, respectively.

Exponential families. Let $\phi(t) \in \mathbb{R}^M$ be a vector of statistics and $\theta \in \Theta \subseteq \mathbb{R}^M$ a vector of canonical parameters and consider a function $f_\theta(t) = \theta^\top \phi(t)$. A family of the form (3.10) parametrized by $\theta \in \Theta \subseteq \mathbb{R}^M$ is called an exponential family [43]. Exponential families include many of the most common distributions (*e.g.*, the Gaussian distribution) and have a large number of properties that make them extremely useful, including having fixed support (dictated by the base measure) within the same family, *i.e.*, its support remains the same across all parameter settings in the family.

3.2.2 Gini-Simpson index and Ω_2-RPM

For $\alpha = 2$, $\Omega_2(p)$ is the Gini-Simpson index and the corresponding Ω_2-RPM can be obtained from f by subtracting a constant λ (that can be both positive or negative) and truncating such that $\int_S \hat{p}_{\Omega_2}[f](t) = 1$:

$$\hat{p}_{\Omega_2}[f](t) = [f(t) - \lambda]_+. \tag{3.13}$$

For finite S the Ω_2-RPM is the sparsemax transformation. On the other hand, for continuous domains with $S = \mathbb{R}^N$, $\Sigma \in \mathbb{R}^{N\times N}$ positive definite and choosing $f(t) = -\frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu)$, the Ω_2-RPM transformation is a multivariate truncated paraboloid,

$$\hat{p}_{\Omega_2}[f](t) = \left[-\frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu) - \lambda\right]_+, \tag{3.14}$$

with

$$\lambda = -\left(\frac{\Gamma(N/2+2)}{\sqrt{\det(2\pi\Sigma)}}\right)^{\frac{2}{2+N}}, \tag{3.15}$$

where $\Gamma(t)$ is the Gamma function. See Martins et al. [27, Section 2.4] for details.

Deformed exponential families and sparse families. For the same parametrization $f_\theta(t) = \theta^\top \phi(t)$ and $\alpha > 0$, deformed exponential families [44, 45] are distributions with the form of (3.6). For $\alpha > 1$, Martins et al. [27] called these α-sparse families. Unlike the case where $\alpha = 0$, these distributions may not have full support, being able to return zero probability values.

Figure 3.1 shows the distributions generated by the Ω_α-RPM for $\alpha \in \{1, 2\}$. The former has full support in $\mathbb{R}^2$ while the latter is able to assign zero probability values.

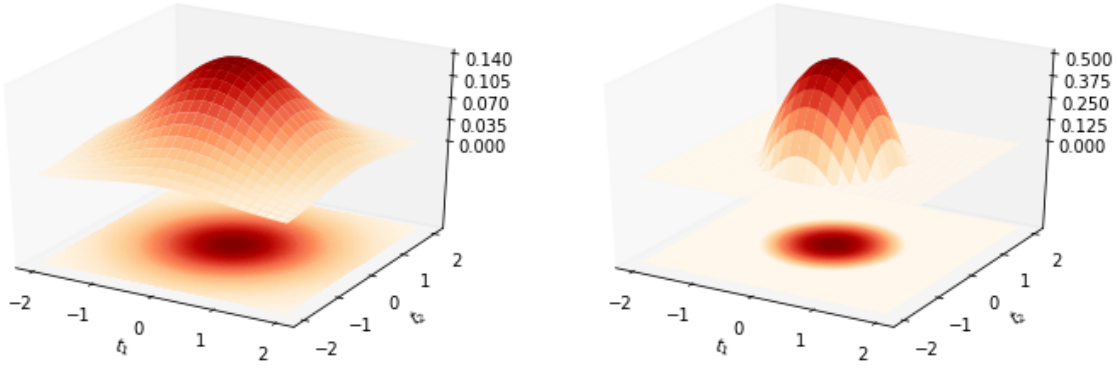

Figure 3.1: 2D distributions generated by the Ω_α-RPM for $\alpha \in \{1, 2\}$. Left: For $\alpha = 1$, bivariate Gaussian $\mathcal{N}(t; 0, I)$. Right: For $\alpha = 1$, truncated paraboloid $\mathcal{TP}(t; 0, I)$. Note that we did not use the same vertical axis' scale and colour scheme for both plots: the *peak* of the density for $\alpha = 1$ ($\mathcal{N}$) is much smaller than for $\alpha = 2$ ($\mathcal{TP}$). Also, the support of the Gaussian density is the whole $\mathbb{R}^2$ plane, whereas for the truncated paraboloid it corresponds to an ellipse (or a circle if it is parametrized by a diagonal covariance matrix such as the 2×2 identity matrix, I).

3.3 Building continuous attention mechanisms

3.3.1 Relation with discrete attention

Usual discrete attention mechanisms work as follows. Assume and input object split in L pieces (*e.g.*, an image with L pixels or regions) with a D-dimensional representation each. It is possible to obtain a matrix representation $V \in \mathbb{R}^{D \times L}$, coming from an encoder in an encoder-decoder architecture (or directly from a CNN). A discrete attention mechanism computes a *score vector* $f = (f_1, ..., f_L)$, in which high scores should correspond to more relevant parts of the input. Then, a discrete transformation $\rho : \mathbb{R}^L \to \triangle^L$ is used to map scores into probabilities, *i.e.*, ρ is applied to the score vector f to produce a probability vector $p = \rho(f)$ – the *attention weights*; softmax is the usual choice for this transformation. Finally, p is used to compute a weighted average of the input (known as the *context vector*), $c = Vp \in \mathbb{R}^D$, that is used to produce the network's decision.

3.3.2 Score and value functions

The transformation ρ can be seen as an Ω-RPM and used to construct continuous attention mechanisms. We explain this procedure assuming that we have an input image. Instead of assuming a finite set, we assume a continuous measure space such as the $\mathbb{R}^2$ plane and represent the image as a continuous *value function* $V : S \subseteq \mathbb{R}^2 \to \mathbb{R}^D$ that maps points in the $\mathbb{R}^2$ plane onto a D-dimensional vector representation. The score vector f is replaced by a *score function* $f : S \to \mathbb{R}$ that can be mapped to a probability density $p \in \mathcal{M}_+^1(S)$. Instead of using a discrete transformation for that mapping, we can use the Ω_α-RPM. The output weighted average (context vector) becomes an expectation of the value function with respect to the probability density, $c = \mathbb{E}_p[V(t)] = \int_S p(t)V(t) \in \mathbb{R}^D$. Note that we could have taken an audio or text signal as input and considered $S \subseteq \mathbb{R}$.

In this construction $\mathcal{M}_+^1(S)$ can be of an arbitrary dimension – including infinite dimensional – so, the score function f and the value function V need to be parametrized: the simplest solution is to consider linear parametrizations defined in terms of a vector of basis functions (*e.g.*, power, sine, cosine or Gaussian basis functions) and a vector of parameters. We can then define $f_\theta(t) = \theta^\top \phi(t)$ and $V_B(t) = B\psi(t)$, where $\phi : S \to \mathbb{R}^M$ and $\psi : S \to \mathbb{R}^N$ are basis functions and $\theta \in \mathbb{R}^M$ and $B \in \mathbb{R}^{D\times N}$ are parameters. Given this, the attention mechanism can be extended to continuous domains and formally defined.

Defining continuous attention mechanism. Consider that $\Omega : \mathcal{M}_+^1(S) \to \mathbb{R}$ is a regularization functional. An attention mechanism is a mapping $\rho : \mathbb{R}^M \to \mathbb{R}^N$ from an input parameter vector $\theta \in \mathbb{R}^M$ to a vector $r \in \mathbb{R}^N$,

$$\rho(\theta) = r = \mathbb{E}_p[\psi(t)], \tag{3.16}$$

with $p = \hat{p}_\Omega[f_\theta]$ and $f_\theta(t) = \theta^\top \phi(t)$. If $\Omega = \Omega_\alpha$, this is called α-entmax attention, denoted as ρ_α. The values $\alpha = 1$ and $\alpha = 2$ lead to softmax and sparsemax attention, respectively. The context vector can then be computed as $c = Br$, which is equivalent to the expression presented above, $c = \mathbb{E}_p[V_B(t)]$.

Defining the value function. An input image is usually represented as a discrete matrix $H \in \mathbb{R}^{D\times L}$ (*e.g.*, a matrix with D channels and L image locations so that each location is represented by a D-dimensional vector that encodes the representation of the image in that region, obtained from a CNN). We can use **multivariate ridge regression** to approximate H and obtain a continuous signal, a value mapping $V_B : S \to \mathbb{R}^D$. Given that $V_B(t) = B\psi(t)$, this consists in optimizing over B to minimize the squared loss plus a ridge penalty. Assuming that $t_l = (\frac{l_1}{\sqrt{L}}, \frac{l_2}{\sqrt{L}})$ for $l_1, l_2 \in [0, \sqrt{L}]$ and choosing the columns of the matrix $F \in \mathbb{R}^{N\times L}$ to be the basis vectors $\psi(t_l)$, we obtain

$$\begin{aligned} B^\star &= \underset{B}{\operatorname{argmin}} \|BF - H\|_F^2 + \lambda \|B\|_F^2 \\ &= H \underbrace{F^\top (FF^\top + \lambda I_N)^{-1}}_{G} = HG, \end{aligned} \tag{3.17}$$

where $\|B\|_F$ is the Frobenius norm of B which corresponds to the Euclidean norm of the vector obtained by listing the coefficients of the matrix,

$$\|B\|_F = (tr(B^\top B))^{1/2} = (\sum_{i=1}^{m}\sum_{j=1}^{n} B_{ij}^2)^{1/2}, \tag{3.18}$$

and $G = F^\top (FF^\top + \lambda I_N)^{-1}$ is a $L \times N$ matrix (see Alg. 1 for pseudo-code). For a specific value of L and N, both F and G depend only on the value of $\psi(t_l)$ and can be obtained offline. Also, the number of regions L in an image is usually constant. Typically, we choose $N << L$, so that the resulting expression for V_B has ND coefficients, much cheaper than the LD coefficients of H.

Algorithm 1: Defining the value function V_B using multivariate ridge regression.

Parameters:

- Representation for discrete parts $H := [h_1, \ldots, h_L] \in \mathbb{R}^{D \times L}$
- Corresponding locations $[t_1, \ldots, t_L] \in S^L$, with $t_l = (\frac{l_1}{\sqrt{L}}, \frac{l_2}{\sqrt{L}})$ for $l_1, l_2 \in [0, \sqrt{L}]$
- Basis functions $\psi : S \to \mathbb{R}^N$
- Ridge penalty $\lambda \geq 0$

Output: Continuous representation $V_B(t) = B\psi(t)$ with $B \in \mathbb{R}^{D \times N}$

Function `Regression`$(H, t_{1:L}, \lambda)$**:**

$F \leftarrow [\psi(t_1), \ldots, \psi(t_L)]$ `// Can be computed offline`

$G \leftarrow F^\top (FF^\top + \lambda I_N)^{-1}$ `// Can be computed offline`

$B \leftarrow HG$ `// Eq. 3.17`

return B

3.3.3 Gradient backpropagation

To train models with gradient-based optimization the Jacobian of the α-entmax transformation ρ_α is needed. Martins et al. [27] introduced and proved an expression for evaluating J_{ρ_α} that uses the β-escort distribution (3.9). For $\beta \geq 0$, a generalized β-covariance is defined,

$$\mathrm{cov}_{p,\beta}[\phi(t), \psi(t)] \;=\; \|p\|_\beta^\beta \times \left(\mathbb{E}_{\tilde{p}_\beta}\left[\phi(t)\psi(t)^\top\right] - \mathbb{E}_{\tilde{p}_\beta}[\phi(t)]\,\mathbb{E}_{\tilde{p}_\beta}[\psi(t)]^\top\right), \tag{3.19}$$

that for $\beta = 1$ recovers the usual covariance; for $\beta = 0$ it can be expressed as a covariance taken with respect to a uniform density on the support of p, scaled by $|\mathrm{supp}(p)|$. The Jacobian of the α-entmax transformation is then

$$J_{\rho_\alpha}(\theta) = \frac{\partial r(\theta)}{\partial \theta} = \mathrm{cov}_{p,2-\alpha}(\phi(t), \psi(t)), \tag{3.20}$$

with $p = \hat{p}_\Omega[f_\theta]$ and $f_\theta(t) = \theta^\top \phi(t)$. This expression allows efficient gradient backpropagation with continuous attention and will be used in the next chapter to derive expressions for the gradient computation of 2D α-entmax continuous attention mechanisms for $\alpha \in \{1, 2\}$.

Chapter 4

2D continuous attention with Gaussian RBFs

In this chapter, we derive expressions for the evaluation and gradient computation of 2D continuous attention mechanisms where $\psi(t)$ are Gaussian radial basis functions (RBFs), *i.e.*, each ψ_j is of the form $\psi_j(t) = \mathcal{N}(t; \mu_j, \Sigma_j)$. We begin by showing how to obtain the parameters μ and Σ for the attention densities, in Section 4.1. The theoretical derivation of 2D continuous attention and its gradient for back-propagation is done in Section 4.2: the cases $\alpha = 1$ (softmax) and $\alpha = 2$ (sparsemax) are studied in Subsections 4.2.1 and 4.2.2, respectively. The reader is referred to Appendices A.1 and A.2 for detailed derivations concerning the second case. Finally, in Section 4.3, we perform a study regarding the implementation of the backward pass, for $\alpha = 2$. We suggest the reader to analize Alg. 2, 3 and 4 along with the text for a better understanding on how to implement 2D continuous attention mechanisms. In this chapter, we keep the notation we have used so far.

4.1 How can we write the attention density?

In Section 3.3 we defined continuous attention mechanism as a mapping $\rho : \mathbb{R}^M \to \mathbb{R}^N$ from an input parameter vector $\theta \in \mathbb{R}^M$ to a vector $r = \mathbb{E}_p[\psi(t)] \in \mathbb{R}^N$, where $p = \hat{p}_\Omega[f_\theta]$ and $f_\theta(t) = \theta^\top \phi(t)$. We also saw that, when considering $\Omega = \Omega_\alpha$, the values $\alpha = 1$ and $\alpha = 2$ lead to softmax and sparsemax attention, respectively. Taking $S = \mathbb{R}^2$, the distribution $\hat{p}_{\Omega_1}[f_\theta]$ is a bivariate Gaussian $\mathcal{N}(t; \mu, \Sigma)$ and $\hat{p}_{\Omega_2}[f_\theta]$ becomes a bivariate truncated paraboloid $\mathcal{TP}\ (t; \mu, \Sigma)$. In both cases, the mean μ and the covariance matrix Σ are related to the canonical parameters by $\theta = [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$.

In this Chapter, we focus on a combined attention setting where we assume that we have access to L discrete attention weights α_i, where $i \in \{1, ..., L\}$. First, we obtain $p \in \triangle^L$ from discrete attention. For 2D continuous softmax we have $\mu = \mathbb{E}_p[t]$ and $\Sigma = \mathbb{E}_p[tt^\top] - \mu\mu^\top$, given that the covariance matrix of a Gaussian coincides with its scale parameter [46]. This convenient property does not hold for multivariate truncated paraboloid distributions thus, for 2D continuous sparsemax, we need to find out how to obtain the parameter Σ from the variance $\mathbb{E}_p[tt^\top] - \mu\mu^\top$.

To estimate Σ of a $\mathcal{TP}(t;\mu,\Sigma)$ from discrete attention weights we perform the following steps:

- express the variance as a function of Σ, $f(\Sigma) = \iint \mathcal{TP}(t;\mu,\Sigma) tt^\top dt$;
- from discrete attention, obtain the variance $\mathrm{Var} = \mathbb{E}_p[tt^\top] - \mu\mu^\top$;
- invert f to obtain $\Sigma = f^{-1}(\mathrm{Var})$.

We now put forward a theorem that results from the direct application of this approach. A detailed proof is in Appendix A.1.

Theorem 1. *Let $\mathcal{TP}\ (t;\mu,\Sigma)$ be a d-dimensional multivariate truncated paraboloid where $t,\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d} \succ 0$, defined as in [27, Section 2.4]. Let $\lambda = -\left(\frac{\Gamma(d/2+2)}{|2\pi\Sigma|^{1/2}}\right)^{\frac{2}{2+d}}$ be the constant that ensures the distribution normalizes to 1, where $\Gamma(t)$ is the Gamma function. Then, the variance of $\mathcal{TP}$ is related to Σ by*

$$\mathrm{Var}(\Sigma) = f(\Sigma) = -\frac{\lambda\,\Sigma}{\frac{d}{2}+2}. \tag{4.1}$$

4.1.1 Examples

The most relevant cases are $d = 1$ (*e.g.*, a 1D text segment or audio input) and $d = 2$ (*e.g.*, an image input). From Theorem 1, we have

- $\boxed{d=1}$

$$\mathrm{Var}(\sigma^2) = -\frac{\lambda\,\sigma^2}{5/2} = \frac{1}{5}\left(\frac{3\sigma^2}{2}\right)^{2/3}. \tag{4.2}$$

We can easily invert (4.2) and obtain

$$\sigma^2 = \frac{2}{3}\left(5\mathrm{Var}(\sigma^2)\right)^{3/2}, \tag{4.3}$$

that can be used to compute σ^2 in a truncated parabola density (1D continuous sparsemax attention), given $\mathrm{Var}(\sigma^2)$ computed from discrete attention weights.

- $\boxed{d=2}$

$$\mathrm{Var}(\Sigma) = -\frac{\lambda\Sigma}{3} = \frac{\Sigma}{3\sqrt{\pi}\,|\Sigma|^{1/4}}. \tag{4.4}$$

Using properties of the determinant,

$$|\mathrm{Var}(\Sigma)| = \left|\frac{\Sigma}{3\sqrt{\pi}\,|\Sigma|^{1/4}}\right| = \frac{|\Sigma|^2}{9\pi}. \tag{4.5}$$

Then, using (4.5) to invert (4.4), we obtain the expression that can be used to compute Σ in a truncated paraboloid density (2D continuous sparsemax attention), given $\mathrm{Var} = \mathbb{E}_p[tt^\top] - \mu\mu^\top$ computed from discrete attention weights:

$$\Sigma = 9\pi|\mathrm{Var}|^{1/2}\mathrm{Var}. \tag{4.6}$$

Algorithm 2: Defining the score function f_θ for 2D α-entmax continuous attention mechanisms with $\alpha \in \{1, 2\}$.

Parameters:

- $\alpha \in \{1, 2\}$
- $p \in \triangle^L$ from discrete attention weights $[\alpha_1, \ldots, \alpha_L]$
- Corresponding locations $[t_1, \ldots, t_L] \in S^L$, with $t_l = (\frac{l_1}{\sqrt{L}}, \frac{l_2}{\sqrt{L}})$ for $l_1, l_2 \in [0, \sqrt{L}]$

Output: Continuous representation $f_\theta(t) = \theta^\top \phi(t)$ with $\theta \in \mathbb{R}^M$

Function `Score`$(\alpha, p, t_{1:L})$**:**

$\mu \leftarrow \mathbb{E}_p[t]$
$\Sigma \leftarrow \mathbb{E}_p[tt^\top] - \mu\mu^\top$
if $\alpha == 2$ **then**
  $\Sigma = 9\pi|\Sigma|^{1/2}\Sigma$ // Eq. 4.6
$\theta \leftarrow [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$
return θ

Alg. 2 illustrates how to use this result in order to obtain the score function f_θ for bidimensional α-entmax continuous attention mechanisms with $\alpha \in \{1, 2\}$.

4.2 Evaluation and gradient computation

We derive expressions for the evaluation and gradient computation of 2D continuous α-entmax attention mechanisms for $\alpha \in \{1, 2\}$. Again, we consider that $\psi(t)$ are Gaussian RBFs.

4.2.1 2D continuous softmax ($\alpha = 1$)

Let us consider an arbitrary D-Dimensional scenario and take $D = 2$ for the 2D continuous softmax case. If $S = \mathbb{R}^D$, for $\phi(t) = [t, tt^\top]$, the distribution $p = \hat{p}_{\Omega_1}[f_\theta]$, with $f_\theta(t) = \theta^\top \phi(t)$, is a multivariate Gaussian where the mean μ and the covariance matrix Σ are related to the canonical parameters as $\theta = [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$. Now, we derive closed-form expressions for the attention mechanism output $\rho_1(\theta) = \mathbb{E}_p[\psi(t)]$ in (3.16) and its Jacobian $J_{\rho_1}(\theta) = \mathrm{cov}_{p,1}(\phi(t), \psi(t))$ in (3.20), when $\psi(t)$ are Gaussian RBFs, i.e., each ψ_j is of the form $\psi_j(t) = \mathcal{N}(t; \mu_j, \Sigma_j)$. Check Alg. 3 for pseudo-code containing a condensed version of our results.

Forward pass. Each coordinate of the attention mechanism output becomes the integral of a product of Gaussians,

$$\mathbb{E}[\psi(t)] = \int_{\mathbb{R}^D} \mathcal{N}(t; \mu, \Sigma)\mathcal{N}(t; \mu_j, \Sigma_j). \tag{4.7}$$

The product of two Gaussians is a scaled Gaussian, *i.e.*, a Gaussian times a scale factor:

$$\mathcal{N}(t; \mu, \Sigma)\mathcal{N}(t; \mu_j, \Sigma_j) = \tilde{s}\mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma}), \tag{4.8}$$

where

$$\tilde{s} = \mathcal{N}(\mu; \mu_j, \Sigma + \Sigma_j), \qquad \tilde{\Sigma} = (\Sigma^{-1} + \Sigma_j^{-1})^{-1}, \qquad \tilde{\mu} = \tilde{\Sigma}(\Sigma^{-1}\mu + \Sigma_j^{-1}\mu_j). \tag{4.9}$$

Algorithm 3: Continuous softmax attention with $S = \mathbb{R}^D$, $\Omega = \Omega_1$ and Gaussian RBFs.

Parameters:

- Gaussian RBFs $\psi(t) = [\mathcal{N}(t; \mu_j, \Sigma_j)]_{j=1}^N$
- Basis functions $\phi(t) = [t, \mathrm{vec}(tt^\top)]$
- Value function $V_B(t) = B\psi(t)$ with $B \in \mathbb{R}^{D \times N}$ `// Alg. 1`
- Score function $f_\theta(t) = \theta^\top \phi(t)$ with $\theta \in \mathbb{R}^M$ `// Alg. 2`

Function `Forward`$(\theta := [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}])$**:**

$r_j \leftarrow \mathbb{E}_{\hat{p}_\Omega[f_\theta]}[\psi_j(t)] = \mathcal{N}(\mu, \mu_j, \Sigma + \Sigma_j), \quad \forall j \in [N]$ `// Eq. 4.10`

return $c \leftarrow Br$ *(context vector)*

Function `Backward`$(\frac{\partial \mathcal{L}}{\partial c}, \theta := [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}])$**:**

for $j \leftarrow 1$ **to** N **do**

$\tilde{s} \leftarrow \mathcal{N}(\mu, \mu_j, \Sigma + \Sigma_j)$

$\tilde{\Sigma} \leftarrow (\Sigma^{-1} + \Sigma_j^{-1})^{-1}$

$\tilde{\mu} \leftarrow \tilde{\Sigma}(\Sigma^{-1}\mu + \Sigma_j^{-1}\mu_j)$

$\frac{\partial r_j}{\partial \theta} \leftarrow \mathrm{cov}_{\hat{p}_\Omega[f_\theta]}(\phi(t), \psi_j(t)) = [\tilde{s}(\tilde{\mu} - \mu); \tilde{s}(\tilde{\Sigma} + \tilde{\mu}\tilde{\mu}^\top - \Sigma - \mu\mu^\top)]$ `// Eqs. 4.11,4.12`

return $\frac{\partial \mathcal{L}}{\partial \theta} \leftarrow \left(\frac{\partial r}{\partial \theta}\right)^\top B^\top \frac{\partial \mathcal{L}}{\partial c}$

Therefore, the forward pass can be computed as:

$$\begin{aligned} \mathbb{E}[\psi(t)] &= \tilde{s} \int_{\mathbb{R}^D} \mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma}) = \tilde{s} \\ &= \mathcal{N}(\mu; \mu_j, \Sigma + \Sigma_j). \end{aligned} \tag{4.10}$$

Backward pass. To compute the backward pass, we have that each row of the Jacobian $J_{\rho_1}(\theta)$ becomes a first or second moment under the resulting Gaussian,

$$\begin{aligned} \mathrm{cov}_{p,1}(t, \psi(t)) &= \mathbb{E}_p[t\psi_j(t)] - \mathbb{E}_p[t]\mathbb{E}_p[\psi_j(t)] \\ &= \int_{\mathbb{R}^D} t\mathcal{N}(t; \mu, \Sigma)\mathcal{N}(t; \mu_j, \Sigma_j) - \tilde{s}\mu \\ &= \tilde{s} \int_{\mathbb{R}^D} t\mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma}) - \tilde{s}\mu \\ &= \tilde{s}(\tilde{\mu} - \mu), \end{aligned} \tag{4.11}$$

and, noting that $\Sigma = \mathbb{E}[(t-\mu)(t-\mu)^\top] = \mathbb{E}[tt^\top] - \mu\mu^\top$,

$$\begin{aligned} \mathrm{cov}_{p,1}(tt^\top, \psi(t)) &= \mathbb{E}_p[tt^\top\psi_j(t)] - \mathbb{E}_p[tt^\top]\mathbb{E}_p[\psi_j(t)] \\ &= \int_{\mathbb{R}^D} tt^\top\mathcal{N}(t; \mu, \Sigma)\mathcal{N}(t; \mu_j, \Sigma_j) - \tilde{s}(\Sigma + \mu\mu^\top) \\ &= \tilde{s} \int_{\mathbb{R}^D} tt^\top\mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma}) - \tilde{s}(\Sigma + \mu\mu^\top) \\ &= \tilde{s}(\tilde{\Sigma} + \tilde{\mu}\tilde{\mu}^\top) - \tilde{s}(\Sigma + \mu\mu^\top) \\ &= \tilde{s}(\tilde{\Sigma} + \tilde{\mu}\tilde{\mu}^\top - \Sigma - \mu\mu^\top). \end{aligned} \tag{4.12}$$

4.2.2 2D continuous sparsemax ($\alpha = 2$)

For $\alpha = 2$, we show in Appendix A.2 how to reduce both the forward and the backward passes to expressions including univariate integrals (with closed form-expression in terms of the erf function) over an interval – easy to integrate numerically – by using the change of variable formula and working with polar coordinates. We prove that for this case both the attention mechanism output,

$$\rho_2(\theta) = \mathbb{E}_p[\psi_j(t)] = \iint_{\mathbb{R}^2} \underbrace{\left[-\lambda - \frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu)\right]_+}_{\mathcal{TP}\ (t;\mu,\Sigma)} \mathcal{N}(t;\mu_j,\Sigma_j)\ dt, \tag{4.13}$$

and its Jacobian,

$$J_{\rho_2}(\theta) = \mathrm{cov}_{p,2}(\phi(t),\psi(t)), \tag{4.14}$$

can be written in terms of functions of the form $-\lambda \int_0^{2\pi} \tilde{s}(\theta) F(\theta)$ and can be easily computed using simple 1D numerical integration methods. Alternatively, we could use 2D methods for numerical integration to solve (4.13) and (4.14) directly; yet, to obtain good results we would have to use complicated bivariate adaptive methods that take a lot of time to reach convergence and that are not GPU friendly – it would become impracticable to plug these mechanisms in neural networks and train them end-to-end. Nevertheless, in Section 4.3, we study different ways to compute 2D integrals over ellipses, including the simplistic approach that we use in this work: in Chapter 6, we show that for Visual Question Answering, in practice, we can approximate these integrals with naive sums, without compromising the overall performance of the model.

4.3 Computation of 2D integrals over ellipses

As discussed before, in order to plug continuous attention mechanisms in neural networks we must have an efficient way of computing both the forward and backward passes within a reasonable amount of time. Both the expectation (4.13) and its Jacobian (4.14) involve solving a bidimensional integral over a region bounded by an ellipse. As we could not find a closed form solution for this problem (that would make this process very fast) we analyzed different approaches to solve it numerically, taking into account its running time and ease of implementation.

First, note that all the integrals in (4.13) and (4.14) are computed over regions bounded by the ellipse that corresponds to the support of a bivariate truncated paraboloid $\mathcal{TP}\ (t;\mu,\Sigma)$ – formally defined by $\{t \in \mathbb{R}^2 \mid \frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu) \leq -\lambda\}$, as in Subsection 3.2.2. We now focus on solving (4.13), given by:

$$\mathbb{E}_p[\psi_j(t)] = \iint_{\mathbb{R}^2} \underbrace{\left[-\lambda - \frac{1}{2}(t-\mu)^\top \Sigma^{-1}(t-\mu)\right]_+}_{\mathcal{TP}\ (t;\mu,\Sigma)} \mathcal{N}(t;\mu_j,\Sigma_j)\ dt,$$

and then make the appropriate changes for the other cases.

A reasonable first approach to solve (4.13) would be to use the fact that in our problem of interest the support of the density p should be in a square (*e.g.*, the unit square $[0,1]^2$ or, to be more inclusive, a

Table 4.1: Comparison between different methods for numerical integration in 2D and 1D. For 1D-Naive, the integrand function was evaluated 100 times with θ linearly spaced between $[0, 2\pi]$. All calculations were done on CPU with NumPy arrays, for a fair comparison.

METHOD	RUNNING TIME	DEPENDENCIES	GPU ALLOWANCE
2D-Num	25.749633	NumPy and SciPy	No
2D-Num-Circle	5.290754	NumPy and SciPy	No
2D-Num-Polar	0.247879	NumPy and SciPy	No
1D-Num	0.023887	NumPy and SciPy	No
1D-Naive	0.014022	NumPy or PyTorch	Yes

10×10 box) if we normalize the input (*e.g.*, an image) to that bound. Given this, we could use numeric integration libraries such as SciPy [47] to directly compute approximations of this integral. Note that this is only possible if we assume that the bound constraints we put on the input are valid; although they are not too restrictive to most of the problems we thought of, it would be good not to rely on this assumption.

To tackle this concern and make our approach as clean as possible, we propose to use the expressions we proved in Appendix A.2 in order to analytically reduce the 2D integral in (4.13) to a 1D integral over an interval, that could be solved using different methods: we could use the same numeric integration library as before or, as a simpler alternative, we could simply evaluate the integrand function on a finite set of points (after reducing the dimension of the 2D integral) and use a weighted sum of these values to approximate the integral over the unit circle.

In order to better understand the pros and cons of each approach, we consider the following methods based on the the ideas described in the previous paragraphs:

- 2D numerical integration (**2D-Num**) – approximate the integral on a 10×10 box;[1]
- 2D numerical integration in a unit circle (**2D-Num-Circle**) – compute an affine transformation so that the support of the density is a unit circle instead of an ellipse. Then, approximate the integral on a 1×1 box;[2]
- 2D numerical integration in polar coordinates (**2D-Num-Polar**) – compute affine transformation so that we can integrate in a unit circle and then reparametrize to polar coordinates;
- 1D numerical integration (**1D-Num**) – reduce the integral analytically to 1D following the approach discussed in Appendix A.2 and then integrate numerically;
- 1D naive sum (**1D-Naive**) – reduce the integral analytically to 1D following the approach discussed in Appendix A.2 and then do naive sum.

Comparison between methods. Table 4.1 shows the average running time of 100 integrals computed using the 5 methods discussed above, where the parameters μ, Σ, μ_j and Σ_j were chosen randomly. All methods excluding 1D-Naive require SciPy library (or similar) for numerical integration; hence, the implementation of these methods cannot make use of GPU parallelization, making them unfeasible for

[1] The support of the function should be on a 10×10 box.
[2] The support of the function should be on a 1×1 box.

Algorithm 4: Continuous sparsemax attention with $S = \mathbb{R}^2$, $\Omega = \Omega_2$ and Gaussian RBFs.

Parameters:

- Gaussian RBFs $\psi(t) = [\mathcal{N}(t; \mu_j, \Sigma_j)]_{j=1}^N$
- Basis functions $\phi(t) = [t, \mathrm{vec}(tt^\top)]$
- Value function $V_B(t) = B\psi(t)$ with $B \in \mathbb{R}^{D \times N}$ `// Alg. 1`
- Score function $f_\theta(t) = \theta^\top \phi(t)$ with $\theta \in \mathbb{R}^M$ `// Alg. 2`
- Number of integration intervals I

Function `Forward(`$\theta := [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$`):`

$\lambda \leftarrow -(\pi\sqrt{\det(\Sigma)})^{-\frac{1}{2}}$
for $\beta \leftarrow 0$ **to** 2π **do**
$\tilde{s}(\beta) \leftarrow$ (A.17)
$F(\beta) \leftarrow$ (A.22)
$r_j \leftarrow r_j - 2\pi\lambda\tilde{s}(\beta)F(\beta)/I, \quad \forall j \in [N]$ `// Solving integral numerically using 1D-Naive`
$\beta = \beta + 1/I$
return $c \leftarrow Br$ *(context vector)*

Function `Backward(`$\frac{\partial \mathcal{L}}{\partial c}, \theta := [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$`):`

for $j \leftarrow 1$ **to** N **do**
for $\beta \leftarrow 0$ **to** 2π **do**
$\tilde{s}(\beta) \leftarrow$ (A.17)
$G(\beta) \leftarrow$ (A.31), $H(\beta) \leftarrow$ (A.34), $M(\beta) \leftarrow$ (A.40)
`// Solve integrals numerically using 1D-Naive`
$\beta = \beta + 1/I$
$\frac{\partial r_j}{\partial \theta} \leftarrow \mathrm{cov}_{\hat{p}_\Omega[f_\theta]}(\phi(t), \psi_j(t))$ `// Eqs. A.24,A.25`
return $\frac{\partial \mathcal{L}}{\partial \theta} \leftarrow \left(\frac{\partial r}{\partial \theta}\right)^\top B^\top \frac{\partial \mathcal{L}}{\partial c}$

Deep Learning applications. On the contrary, 1D-Naive only requires simple calculations that can be easily done using, for instance, PyTorch [48] tensors. Alg. 4 contains pseudo-code illustrating how to implement 2D continuous sparsemax using 1D-Naive as integration method.

Chapter 5

Multimodal attention densities

In this chapter, we construct multimodal continuous attention mechanisms with mixtures of attention densities. We begin by introducing mixture models, in Section 5.1. Then, we propose to extend the framework presented in Chapters 3 and 4 to multimodal distributions as a natural follow-up on the previous work, pointing out the difference between the novel multimodal attention mechanisms and the standard multi-head attention [8], in Section 5.2. In Section 5.3, we study the case where the attention density can be parametrized as a mixture of Gaussians, adapting the well known Expectation Maximization (EM) algorithm to work with weighted data (*e.g.*, discrete attention weights).

5.1 Mixture models

Although the success of unimodal distributions such as Gaussians is unquestionable, it is also known that they have some limitations when modelling real datasets: by its own nature, they only have a single maximum and so cannot model multimodal distributions properly. Hence, mixture models appear as a very important tool to represent arbitrarily complex probability density functions.

Consider that we are provided with data that has 2 distinct clumps, as in Figure 5.1. A unimodal distribution would fail to capture them, putting much of its probability mass in the central region between them, where the data is scarcer. On the other hand, a better representation would be given by fitting a multimodal distribution (with 2 *peaks*) by *maximum likelihood*, using techniques that we will discuss later. By taking linear combinations of 2 or more distributions (*e.g.*, a unimodal Gaussian), we form the so called mixture distributions. Formally, if x is a d-dimensional vector representing a data point, a mixture model assigns it the probability

$$p(x|\Theta) = \sum_{k=1}^{K} \pi_k p(x|\theta_k), \tag{5.1}$$

where the parameters π_k are called *mixing coefficients* and satisfy $\pi_k \geq 0$ for $k = 1, \dots, K$ and $\sum_{k=1}^{K} \pi_k = 1$; each θ_k is the set of parameters defining the k-th component of the mixture; and $\Theta = \{\theta_1, \dots, \theta_K, \pi_1, \dots, \pi_K\}$ is the complete set of parameters needed to specify the mixture. The

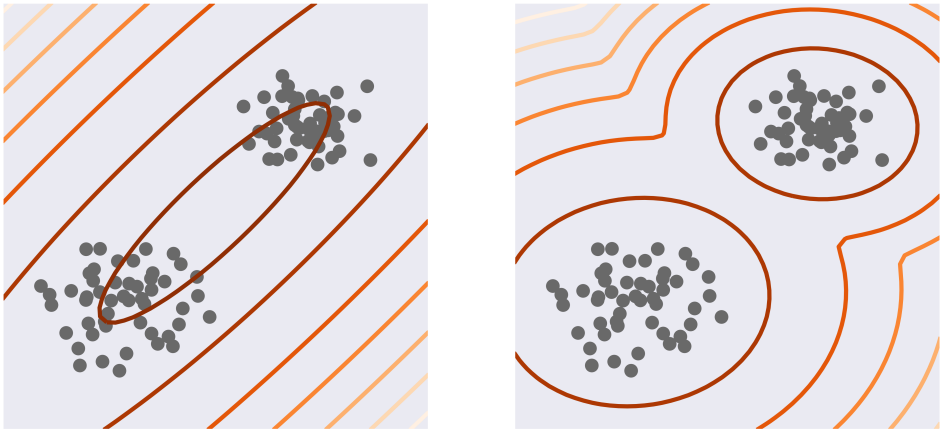

Figure 5.1: Fitting data with 2 clumps using unimodal and multimodal distributions. Contours of constant density have the same colour. Left: Single Gaussian distribution fitted to the data using maximum likelihood. Right: Mixture of two Gaussians fitted to the data using the EM algorithm (discussed below).

definition of mixture is completely general regarding the functional form of each component, meaning that $p(x|\theta_k)$ can take different forms. For instance, a Gaussian Mixture Model (GMM) with K *components* is a superposition of K Gaussian densities (*i.e.*, $p(x|\theta_k) = \mathcal{N}(x|\mu_k, \Sigma_k)$) of the form

$$p(x|\mu, \Sigma, \pi) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k). \tag{5.2}$$

The complete set of parameters of a Gaussian mixture distribution includes $\pi = \{\pi_1, ..., \pi_K\}$, $\mu = \{\mu_1, ..., \mu_K\}$ and $\Sigma = \{\Sigma_1, ..., \Sigma_K\}$. We discuss how to set these parameters in Section 5.3.

5.2 Multimodal continuous attention

We can extend the framework presented in Chapters 3 and 4 to multimodal distributions by considering mixtures of unimodal distributions,

$$p(t) = \sum_{k=1}^{K} \pi_k p_k(t), \tag{5.3}$$

where each $p_k = \hat{p}_\Omega[f_{\theta_k}]$ is a unimodal distribution (*e.g.*, a Gaussian or a truncated paraboloid) and $\pi \in \Delta^K$ are mixing coefficients defining the weight of each component of the mixture. For instance, in the first example, $p(t)$ becomes a mixture of Gaussians; we discuss later possible methods for obtaining the mixing coefficients, π.

Using the definition of continuous attention mechanism presented in Section 3.3 and, from the linearity of expectations, we can compute the output of the multimodal attention mechanism as

$$r = \mathbb{E}_p[\psi(t)] = \sum_{k=1}^{K} \pi_k \underbrace{\mathbb{E}_{p_k}[\psi(t)]}_{r_k} = \sum_{k=1}^{K} \pi_k r_k, \tag{5.4}$$

where r_k is the output of an individual (unimodal) attention mechanism. The context representation is

$$c = \mathbb{E}_p[B\psi(t)] = Br = \sum_{k=1}^{K} \pi_k \underbrace{Br_k}_{c_k}, \tag{5.5}$$

where each c_k is the context representation of each individual attention mechanism; that is, c is a mixture of the context representations for each component. The backpropagation step for the multimodal case is simple, since this decomposes into a linear combination of unimodal attention mechanisms, each of which has a simple/closed-form Jacobian.

Relation to multi-head attention. It is important to note that our construction is not the same as the standard multi-head attention scenario, where the projection matrices learned as model parameters are head-specific (remember (2.14) and (2.15)) [8]. On the contrary, we assume that B is fixed for each k, *i.e.*, B does not depend on k. From a computational point of view, this property seems appealing given that it is possible to compute a single B for each example and still obtain a context vector that contains information from different "heads", through different unimodal attention mechanisms.

How can we obtain π? We may assume that $\pi \in \Delta^K$ is given by a discrete transformation such as softmax or sparsemax on some scores that may depend on $\{p_k\}_{k=1}^K$, defining the weight of each component of the mixture; in the latter case, the number of components in a mixture may vary (K becomes a maximum number of components, but fewer may be selected depending on the input data).

What if we have access to a set of attention weights? Consider that we are provided with a set of points equally spaced in the unit square $[0, 1]^2$ and its correspondent discrete attention weights (the same setting considered in Chapter 4). Intuitively, one could say that higher the attention weight, more important the contribution of that specific point to the network's decision. We have seen that by using moment matching we could parametrize the attention density as a unimodal distribution. For multimodal distributions, we can think of this problem as that of fitting a mixture model to weighted data. In that context, we have to deal with 2 different issues: how to estimate the number of components, which we discuss in Subsection 5.3.3; and how to estimate the parameters defining the mixture model. To the latter question, the usual answer is the EM algorithm [49, 50], which converges to a *maximum likelihood* estimate of the mixture parameters. For instance, for $\alpha = 1$ the corresponding unimodal attention density is a Gaussian; we can easily adapt EM to deal with weighted data, allowing us to obtain the full set of parameters of a mixture of Gaussians – defining a multimodal attention density, $p(t)$.

5.3 The EM algorithm for GMMs

In this section, we explain how to use the EM algorithm to fit a GMM to both non-weighted and weighted data. Bear in mind that we present a brief summary (almost like a recipe) on the EM algorithm, whose detailed exposition can be found in [49, 50].

5.3.1 Non-weighted data

Let $X = \{x_1, ..., x_N\}$ be the observed data. Given a GMM, the goal is to maximize the likelihood function with respect to the parameters (means and covariance matrices of the components and mixing coefficients). The EM algorithm for Gaussian mixtures goes as follows:

1. Initialize the parameters μ_k, Σ_k and π_k and evaluate the initial value of the log likelihood function

$$\ln p(X|\mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\}. \tag{5.6}$$

2. **E step.** Evaluate the responsibilities[1] using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)}. \tag{5.7}$$

3. **M step.** Re-estimate the parameters using the current responsibilities

$$\mu_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n, \tag{5.8}$$

$$\Sigma_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{\text{new}})(x_n - \mu_k^{\text{new}})^\top, \tag{5.9}$$

$$\pi_k^{\text{new}} = \frac{N_k}{N}, \tag{5.10}$$

   where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{5.11}$$

4. Re-evaluate the log likelihood (5.6) using the current parameter values and check for convergence of either the parameters or the log likelihood. Return to step 2 if the convergence criterion is not satisfied.

5.3.2 Weighted data

Let $X = \{x_1, ..., x_N\}$ be the observed data and $W = \{w_1, ..., w_N\}$ the weights associated with X, where $w_n \geq 0$ is the weight indicating the relevance of observation x_n. Note that, if obtained from a softmax transform, we have $w_n > 0$. Gebru et al. [51] proposed to incorporate the weights into the model by "*observing x w times*" and changing the log likelihood function accordingly: they raise $\mathcal{N}(x; \mu, \Sigma)$ to the power w and notice that $\mathcal{N}(x; \mu, \Sigma)^w \propto \mathcal{N}(x; \mu, \Sigma/w)$, deriving a new mixture model where w plays the role of precision. However, they focus on the case where the weights are treated as random variables.

[1] This is also known as the *membership weight* of datapoint x_n in cluster k. The membership weights reflect the uncertainty, given x_n and the parameters Θ, about which of the K components generated x_n. Note that we assume that each x_n was generated by a single component in our generative mixture model.

Our approach is simpler; we can change the algorithm for non-weighted data in Subsection 5.3.1 and include the information provided by the weights by changing the way we re-estimate the parameters each iteration. For the M step, the parameters should now be updated using

$$\mu_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} w_n \gamma(z_{nk}) x_n, \tag{5.12}$$

$$\Sigma_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} w_n \gamma(z_{nk})(x_n - \mu_k^{\text{new}})(x_n - \mu_k^{\text{new}})^\top, \tag{5.13}$$

$$\pi_k^{\text{new}} = N_k, \tag{5.14}$$

where

$$N_k = \sum_{n=1}^{N} w_n \gamma(z_{nk}). \tag{5.15}$$

Instead of using (5.6), we should now evaluate a weighted log likelihood function,

$$\sum_{n=1}^{N} w_n \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \right\}, \tag{5.16}$$

where the log likelihood of each point is multiplied by the correspondent weight. Note that if we consider that the weight associated with each observation is the same, *i.e.*, $w_n = \frac{1}{N}$, we recover the usual expressions for the EM algorithm.

5.3.3 Estimating the number of components

The *maximum likelihood* criterion cannot be used to estimate the number of components K in a mixture density. If $\mathcal{M}_k$ is a class composed by all Gaussian mixtures with K components, it is trivial to show that $\mathcal{M}_K \subseteq \mathcal{M}_{K+1}$ and thus the maximized likelihood is a non decreasing function of K, useless as a criterion to estimate the number of components [52].

Several *model selection* methods were proposed to tackle the concern of estimating the number of components of a mixture [53, Chapter 6]. The likelihood function as defined in (5.16) is of no direct use, since it increases with k. We focus on penalized likelihood methods such as the Bayesian Information Criterion (BIC) [54], the Akaike Information Criterion (AIC) [55] and the Minimum Description Length (MDL) [56], where the EM algorithm is used to obtain different parameter estimates for a range of values of k, $\{\hat{\Theta}_k,\ k = k_{min}, \ldots, k_{max}\}$, and the number of components is chosen according to

$$k^\star = \arg\min_k \{\mathcal{C}(\hat{\Theta}_k, k),\ k = k_{min}, \ldots, k_{max}\}, \tag{5.17}$$

where $\mathcal{C}(\hat{\Theta}_k, k)$ is a model selection criterion that usually has the form

$$\mathcal{C}(\hat{\Theta}_k, k) = -2 \ln (X | \hat{\Theta}_k) + \mathcal{P}(k), \tag{5.18}$$

where $\mathcal{P}(k)$ is an increasing function penalizing higher values of k (*e.g.*, $\mathcal{P}_{\text{BIC}}(k) = k \ln n$, with n being

the number of data points). For the weighted data scenario presented in Section 5.3.2 we cannot use the number of points; thus, we can write

$$\mathcal{P}(k) = \lambda\, k, \tag{5.19}$$

where $\lambda > 0$ is an hyperparameter that can be obtained, for instance, using *cross-validation*. The resulting model selection criterion

$$\mathcal{C}(\hat{\Theta}_k, k) = -2 \ln (X|\hat{\Theta}_k) + \lambda\, k, \tag{5.20}$$

will be used in Section 6.3 to estimate the number of components in a multimodal continuous attention density.

5.3.4 Initialization

The EM algorithm requires an initial choice for the complete set of parameters Θ (an *initialization*); when applied to GMMs, Θ includes not only the means $\mu = \{\mu_1, ..., \mu_K\}$ and covariance matrices $\Sigma = \{\Sigma_1, ..., \Sigma_K\}$ of each component but also the set of mixing coefficients $\pi = \{\pi_1, ..., \pi_K\}$. This becomes an issue of the utmost importance because EM is not guaranteed to converge to a global maximizer of the log likelihood function, getting stuck at a local maximizer most of the times [57], meaning that the final estimate depends on the initialization.

A common strategy to alleviate this issue consists of considering several different initializations (*e.g.*, multiple random initializations), run EM that number of times and choose the final estimate that leads to the highest likelihood [50].

Chapter 6

Applications in Visual Question Answering

With both the forward and backward passes solved, we are now able to plug continuous attention mechanisms in neural networks and train them end-to-end via the gradient backpropagation algorithm. In this chapter, we experiment with α-entmax attention mechanisms in visual question answering – a task that combines language and vision – in order to improve focus and possibly provide better explanations via smoother attention maps. We begin by making a brief introduction on this task, in Section 6.1. Then, we describe our experiments with 2D continuous attention and discuss the results according to the standard metrics, in Section 6.2. Moreover, we present some attention plots showing the promising results of our approach. Finally, in Section 6.3 we perform experiments with Multimodal Continuous Attention (MCA), showing the improved attention maps that they are able to generate.

6.1 Understanding the task

Visual Question Answering (VQA) is a task whose main goal is to provide an accurate answer, given an image and a question about it (both the question and the answer are in natural language). Given this, VQA requires a fine-grained simultaneous understanding of images and text-based questions, combining the researching fields of Computer Vision and Natural Language Processing. For example, to answer the question "*Who is wearing glasses?*" in Figure 6.1, first of all, the model must understand the question. Then, it should have some common-sense knowledge to know where to look for the glasses and to distinguish between the man and the woman. This is only possible if the model is capable of focusing on the right region (or regions), understanding which visual information is more relevant to answer that particular question.

In the last few years using *bounding box* features to represent images became the standard choice for many tasks that require both vision and language, including VQA - this is known as *bottom-up* attention [24] and requires the use of pre-trained object detectors [59]. The process of acquiring those features involves two main steps (region selection and region feature computation) that are computa-

Figure 6.1: Examples of open-ended questions from the VQA-v2 dataset requiring common sense knowledge along with a visual understanding of the scene to answer questions. From Goyal et al. [58, Figure 1].

tionally expensive and time consuming, making this process less suitable for practical applications. On the contrary, when using directly VGG [60] or ResNet [61] grid features we can skip some of the expensive steps related to the bottom-up attention approach, resulting in noticeable speed-ups that might be crucial at inference time. In this thesis we focus on the second case.

6.2 Experiments with 2D continuous attention

We now plug our 2D continuous attention mechanisms in a VQA model that uses grid features to represent the images. Throughout this section, all the models we experimented with use the same features and were trained only on the train set without data augmentation.

6.2.1 Dataset and architecture

We used the VQA-v2 dataset [58] with the standard splits (443K, 214K, and 453K question-image pairs for train/dev/test, the latter subdivided into test-dev, test-standard, test-challenge and test-reserve). We adapted the implementation of [62],[1] consisting of a Modular Co-Attention Network (MCAN): our architecture is the same as its encoder-decoder version except that we represent the image input with grid features generated by a ResNet pretrained on ImageNet [61, 63], instead of bounding-box features [24].

Architecture. The images are resized to 448×448 before going through the ResNet that outputs a feature map of size $14 \times 14 \times 2048$. To represent the input question words we use 300-dimensional GloVe word embeddings [64], yielding a question feature matrix representation. The architecture inspired by the Transformer [8] consists of a deep co-attention learning phase that takes the question and the image representations as inputs and passes them through an encoder-decoder architecture (see figure 6.2):

[1] https://github.com/MILVLG/mcan-vqa

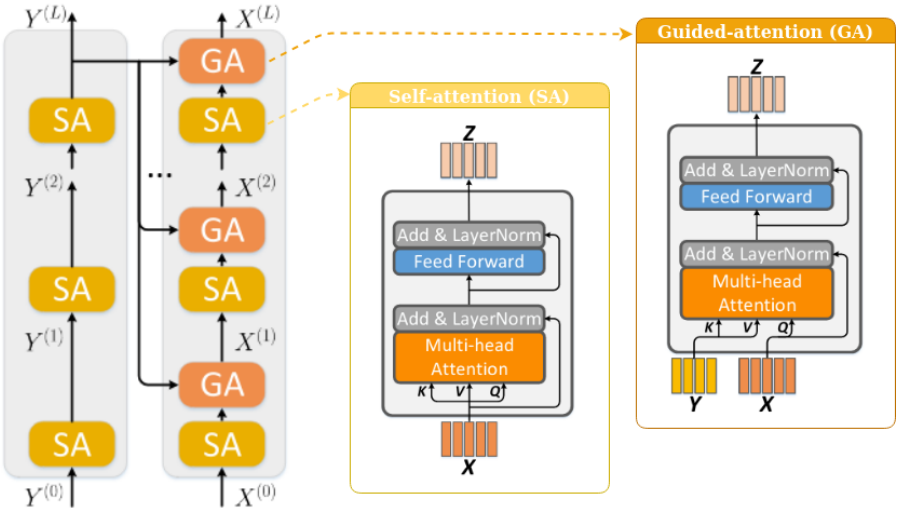

Figure 6.2: Deep co-attention learning phase: encoder-decoder approach including self-attention and guided-attention units. Adapted from Yu et al. [62, Figure 2 and 5].

this can be seen as an encoder to learn the question representation Y and a decoder to learn the image representation X, where Y guides the attention given to the image features. In this case, the encoder is composed by L stacked self-attention layers and the decoder by L self-attention followed by guided attention layers - after this phase, Y and X already contain rich information about the attention weights over the question words and image regions. Y and X are then passed through an output attention model using discrete or 2D continuous attention for X, in order to obtain the final attended features $\tilde{y}$ and $\tilde{x}$, respectively (in the next paragraphs we explore different methods to obtain $\tilde{y}$ and $\tilde{x}$). A fused feature z can be obtained as $z = \text{LayerNorm}(W_y\tilde{y} + W_x\tilde{x})$, where W_y and W_x are linear projection matrices. Finally, z is projected into a vector $s \in \mathbb{R}^N$, where N is the number of possible answers.

6.2.2 Attention model

We consider 3 different attention models: discrete attention, 2D continuous softmax attention and 2D continuous sparsemax attention. The discrete attention model attends over a 14×14 grid and $\tilde{x}$ is obtained as $\tilde{x} = \sum_{i=1}^{L} \alpha_i x_i$, where $\alpha = \text{softmax}(\text{affine}(X))$. Since our focus is to experiment with different 2D attention mechanisms, the final attended question feature $\tilde{x}$ is always obtained with the same procedure, using discrete attention. For continuous attention, we normalize the image size into the unit square $[0, 1]^2$ with each coordinate t_l positioned at $(\frac{l_1}{\sqrt{L}}, \frac{l_2}{\sqrt{L}})$ for $l_1, l_2 \in [0, \sqrt{L}]$ creating a meshgrid. We fit a 2D Gaussian ($\alpha = 1$) or truncated paraboloid ($\alpha = 2$) as the attention density; both correspond to $f(t) = -\frac{1}{2}(t - \mu)^\top \Sigma^{-1}(t - \mu)$, with $\Sigma \succ 0$. We use the mean and variance according to the discrete attention probabilities and obtain μ and Σ with moment matching. We use $N \in \{49, 100\} \ll 14^2$ Gaussian RBFs, with $\tilde{\mu}$ linearly spaced in $[0, 1]^2$ and $\tilde{\Sigma} = 0.001 \cdot \text{I}$. Overall, the number of neural network parameters is the same as in discrete attention.

How can we use 2D continuous attention after all? We want to obtain the final attended feature $\tilde{x}$ from an image matrix representation $X \in \mathbb{R}^{D \times L}$, where $D = 2048$ is the channel dimension and $L = 14 \times 14$ is the spacial dimension - a square grid. Also, we assume that we have access to L discrete attention weights α_i, where $i \in \{1, ..., L\}$.

We opt to use N Gaussian RBFs as basis functions $\psi(t)$ in order to represent the image as a continuous function $V_B(t) = B\psi(t)$. Now, how can we obtain the optimum value $B^\star$, *i.e.*, how can we choose B so that V_B is a good representation of the image? One simple option is to use multivariate ridge regression: we evaluate each basis function on every normalized image location and pack the basis vectors as the columns of a matrix $F \in \mathbb{R}^{N \times L}$; from expression (3.17), we can compute $G \in \mathbb{R}^{L \times N}$ and, finally, obtain $B^\star \in \mathbb{R}^{D \times N}$.[2] The parameters μ and Σ for the attention density – a bivariate Gaussian $\mathcal{N}(t; \mu, \Sigma)$ for 2D continuous softmax and a bivariate truncated paraboloid $\mathrm{TP}(t; \mu, \Sigma)$ for 2D continuous sparsemax – can be obtained with moment matching according to the discrete attention weights α_i, following the procedure discussed in Section 4.1. With this, we have everything that we need to compute $\tilde{x}$: from expression (3.16) we can write $r = \mathbb{E}_p[\psi(t)] \in \mathbb{R}^N$ and the final attended feature is $\tilde{x} = B^\star\, r \in \mathbb{R}^D$. The diagram in Figure 6.3 summarizes our method to obtain the final attended image feature $\tilde{x}$ via a 2D continuous attention mechanism.

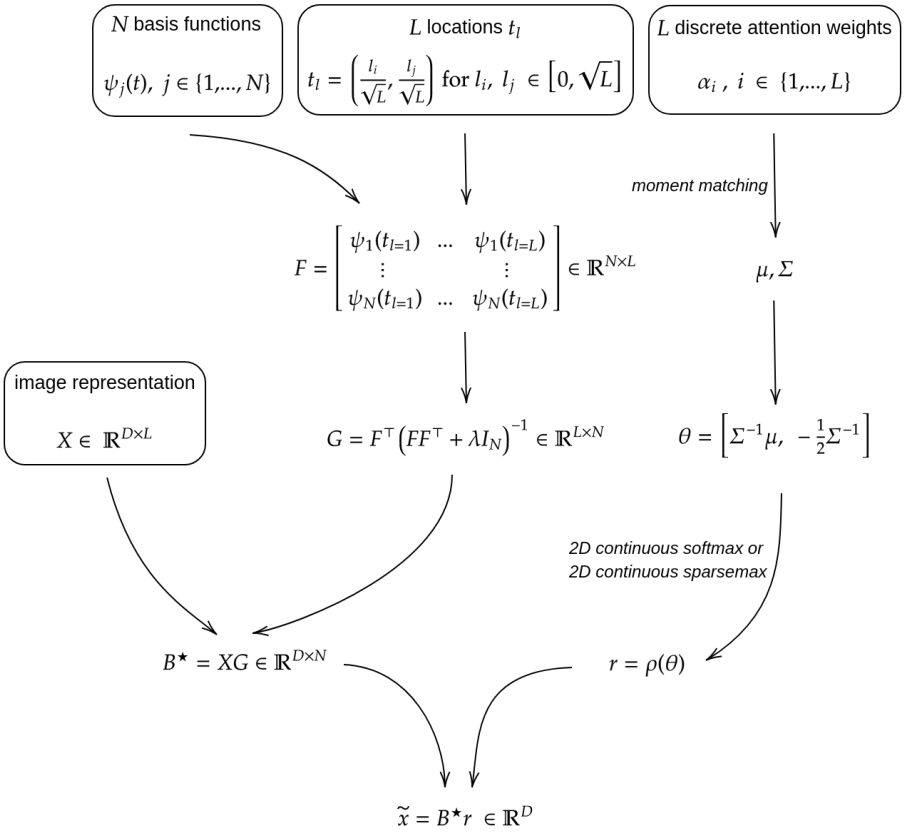

Figure 6.3: Diagram of a 2D continuous attention mechanism as the output attention for VQA. λ represents a Ridge penalty and I_N is the $N \times N$ identity matrix. Note that for given L and N, G depends only on the value of the basis functions and can be obtained offline.

[2] In practice, we can extend the image area using *padding* – considering a square of side 2 instead of 1, containing the unit square in its center – and evaluate each basis function on it. Then, we compute G and ignore the image locations we added. This helps to make sure that near the outer limits of the image V_B approximates zero. Our experiments show that by doing this it is possible to get $+0.5\%$ overall accuracy on the validation set.

Table 6.1: Hyperparmeters for VQA. The second part of the table includes the hyperparameters for continuous attention models. l.s. means *linearly spaced in*.

HYPERPARAMETER	VALUE
Batch size	64
Word embeddings size	300
Input image features size	2048
Input question features size	512
Fused multimodal features size	1024
Multi-head attention hidden size	512
Number of MCA layers	6
Number of attention heads	8
Dropout rate	0.1
MLP size in flatten layers	512
Optimizer	Adam
Base learning rate at epoch t starting from 1	$\min(2.5t \cdot 10^{-5}, 1 \cdot 10^{-4})$
Learning rate decay ratio at epoch $t \in \{10, 12\}$	0.2
Number of epochs	13
N	$\{49,100\}$
$\tilde{\mu}$	l.s. $[0, 1]^2$
$\tilde{\Sigma}$	$0.001 \cdot \mathrm{I}$
Padding	Yes
Ridge penalty	0.01
Number of integration intervals (2D continuous sparsemax)	100

6.2.3 Implementation

Hyperparameters and implementation. Table 6.1 shows the hyperparameters used for all the VQA experiments presented, including the hyperparameters for continuous attention models. Remember that to compute both the forward and the backward passes for 2D continuous sparsemax attention with Gaussian RBFs, we need to integrate univariate integrals over the unit circle numerically (see section A.2 for all the derivations and formulas). Instead of using adaptive methods for numerical integration (*e.g.*, *Gauss-Kronrod quadrature rules* - see Laurie [65] for an overview of these methods) we found out that we could simply evaluate the integrand function on a finite set of points and use a weighted sum of these values to approximate the integral over the unit circle, without comprising the performance of the VQA model. This is an important finding given that it allows an easy implementation of parallel computations, reducing the running time when compared to adaptive numerical integration methods (see Section 4.3).

Our computing infrastructure consists of 4 machines with the specifications shown in appendix B.1 and all our experiments were executed in a single GPU. Moreover, we open-sourced our pytorch implementation based on the original MCAN repository. Our code is available at https://github.com/antonio-farinhas/mcan-vqa-cont, along with additional information on how to reproduce our results.

6.2.4 Results and attention visualization

Results. The results in Table 6.2 show the accuracies for all the attention models: the results are similar, with a slight advantage for 2D continuous softmax with $N = 100$ basis functions. Even though we used less basis functions than image regions ($N \ll L = 14 \times 14$), 2D continuous attention performed as well as (or even better than) discrete attention. Moreover, we can see that we don't need a large

Table 6.2: Accuracies of different models on the *test-dev* and *test-standard* splits of VQA-v2. For the continuous attention models we used $N \in \{49, 100\}$ Gaussian RBFs $\mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma})$, with $\tilde{\mu}$ linearly spaced in $[0, 1]^2$ and $\tilde{\Sigma} = 0.001 \cdot \mathrm{I}$.

ATTENTION	N	Test-Dev				Test-Standard			
		Yes/No	Number	Other	Overall	Yes/No	Number	Other	Overall
Discrete softmax	-	83.40	43.59	55.91	65.83	83.47	42.99	56.33	66.13
2D continuous softmax	100	83.40	44.80	55.88	**65.96**	83.79	44.33	56.04	**66.27**
2D continuous softmax	49	83.32	44.29	55.78	65.82	83.57	43.70	56.07	66.12
2D continuous sparsemax	100	83.10	44.12	55.95	65.79	83.38	43.91	56.14	66.10
2D continuous sparsemax	49	83.38	44.25	55.87	65.88	83.69	43.00	56.09	66.10

Figure 6.4: Attention maps for an example in the VQA-v2 dataset: original image, discrete attention, 2D continuous softmax ($N = 100$) and 2D continuous sparsemax ($N = 100$). The latter encloses all probability mass within the outer ellipse.

number of basis functions to obtain good results given that for $N = 49$, that is, by choosing the number of basis functions to equal a quarter of the number of image regions, the results are already satisfying: the results for all the attention models are very similar on the *test-standard* split; on the *test-dev* split, 2D continuous sparsemax performed a bit better than the other variants.

Attention visualization. We use the standard way in the literature to visualize where discrete attention is looking when making a decision: splitting the image in a 14×14 grid and superimposing the attention weights. However, as we constructed 2D continuous attention in a different manner, attention visualization should be done in other way. The best way to represent the ellipses that our method is able to identify in images consists in superimposing the contours of the (unnormalized) attention density and the image. Moreover, we use the same colormap for both softmax and sparsemax for a fair comparison; for sparsemax, however, we make the zero-level contour thicker to emphasize sparsity and to show the qualitative difference between the two mechanisms.

Figure 6.4 shows an example where, in the baseline model, discrete attention is too scattered, possibly mistaking the lamp with a TV screen; contrarily, our continuous attention models focus on the right region and answer the question correctly, with 2D continuous sparsemax enclosing all the relevant information in its supporting ellipse. Similarly, in *Yes/No* questions discrete attention tends to be more diffuse than its continuous counterpart, sometimes leading incorrect answers (Figure 6.5). Figure 6.6 illustrates the difficulties that continuous attention models may face when trying to focus on objects that are too far

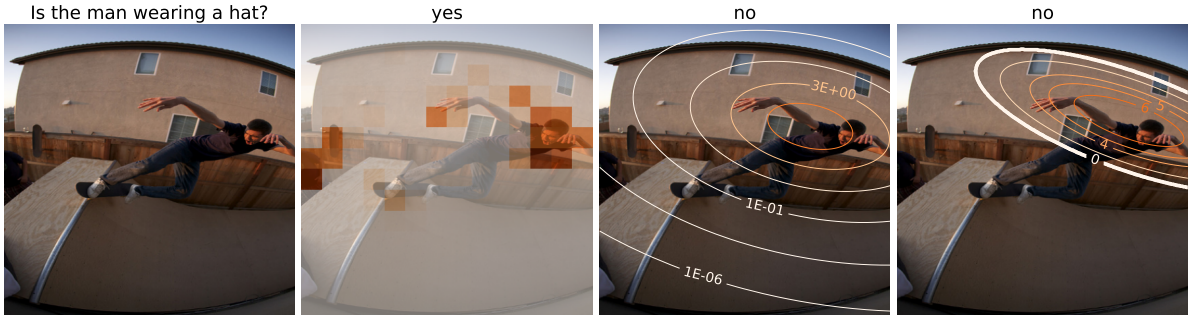

Figure 6.5: Attention maps for an example in the VQA-v2 dataset: original image, discrete attention, 2D continuous softmax ($N = 100$) and 2D continuous sparsemax ($N = 100$). The latter encloses all probability mass within the outer ellipse.

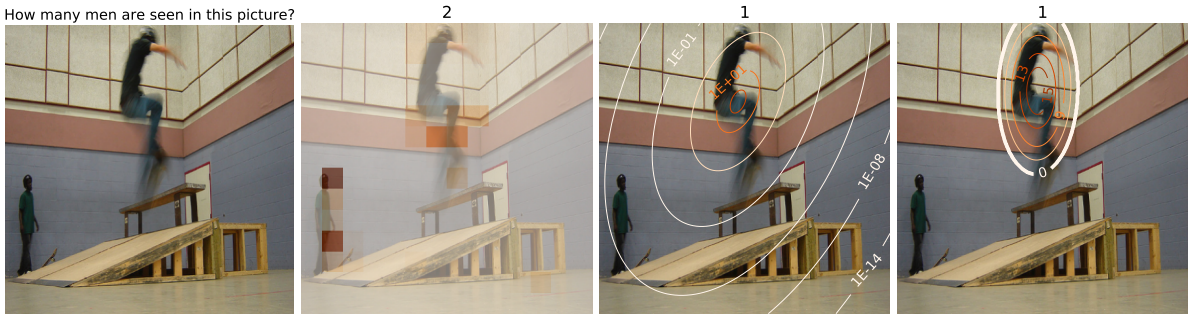

Figure 6.6: Attention maps for an example in the VQA-v2 dataset: original image, discrete attention, 2D continuous softmax ($N = 100$) and 2D continuous sparsemax ($N = 100$). Failure of continuous attention models.

from each other or that seem to have different relative importance to answer the question. Intuitively, in VQA, this becomes a problem when counting objects in those conditions. However, Figure 6.8 shows that continuous attentions ellipses are also capable of becoming wide, including relevant objects that are far from each other. On the other side, in counting questions that require the understanding of a contiguous region of the image only, continuous attention may perform better (Figure 6.7).

Previously, we explained that the parameters of the attention densities for continuous attention models were firstly obtained from discrete attention weights with moment matching. Notwithstanding this, our models were able to learn where they should focus and adapt the attention density parameters so that they do not end up being just the mean and the variance of the discrete attention weights. This is clearly shown in Figures 6.4 and 6.6 where it is possible to distinguish 2 separated regions of high discrete attention probability and the means of the continuous attention densities are located in one of those regions and not in somewhere in the middle of both.

To sum up our findings, we can say that usually discrete attention is more diffuse than its continuous counterpart (which is very intuitive) and attends to multiple regions in the image. For VQA, this might be good for very complex question/image pairs. Note, however, that continuous attention ellipses are also capable of becoming wide, including different regions of interest. On the other hand, when the relevant part of the image is concentrated in a specific region our method conduces to better and more self-explanatory answers. Furthermore, it is important to compare the 2 continuous mechanisms: by fitting a Gaussian as the attention density (continuous softmax) every region in the image is assigned with

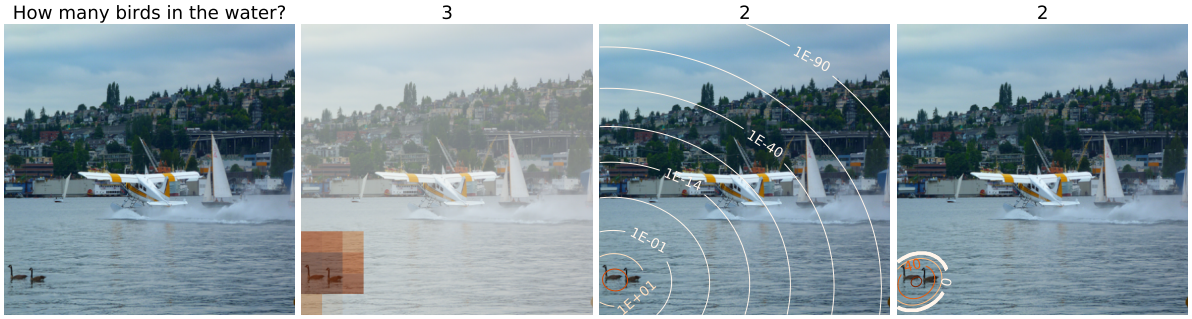

Figure 6.7: Attention maps for an example in the VQA-v2 dataset: original image, discrete attention, 2D continuous softmax ($N = 100$) and 2D continuous sparsemax ($N = 100$). Continuous attention models had a better understanding of a small contiguous region in the image.

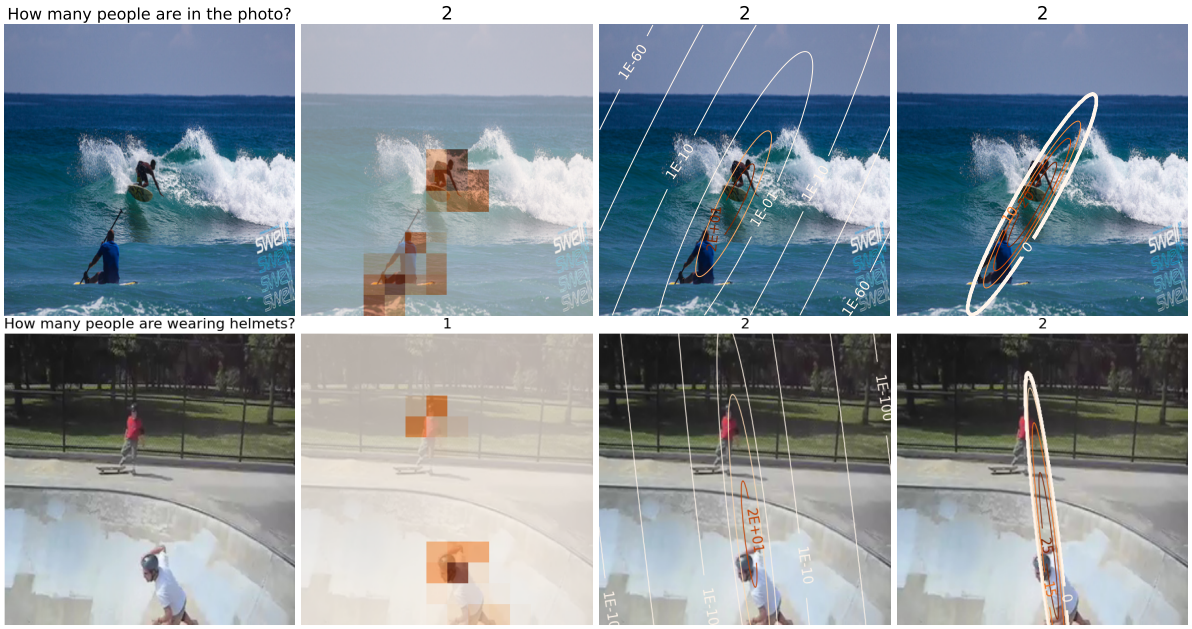

Figure 6.8: Attention maps for an example in the VQA-v2 dataset: original image, discrete attention, 2D continuous softmax ($N = 100$) and 2D continuous sparsemax ($N = 100$). Continuous attention ellipses are capable of becoming wide.

some probability mass; by fitting a truncated paraboloid (continuous sparsemax), the attention density becomes sparse, that is, only the relevant regions of the image are assigned with non-zero probability mass – we found out that this usually results in better explanations.

6.3 Experiments with multimodal continuous attention

Following Section 5.2, we experiment with Multimodal Continuous Attention (MCA) by using mixtures of continuous attention densities. In these experiments, we use the same dataset and architecture as in Subsection 6.2.1. Throughout this section, we omit some of the implementation details regarding unimodal continuous attention models as they can be found in Subsections 6.2.2 and 6.2.3. The hyper-parameters are the same as in Table 6.1.

6.3.1 Attention model

We consider 2 different scenarios. First, we choose a fixed number K and train a model from scratch, assuming that each attention density can be modeled as a K-component multimodal distribution (we refer to this attention model as **K-MCA**, hereinafter). Second, we use a model trained with unimodal continuous attention and, at test time, consider multimodal distributions, using the model selection criterion (5.20) to choose the number of components from a set of possible choices. We refer to the latter as **test-MCA**.

K-MCA. We consider multimodal attention densities with $K \in \{2, 4\}$ components. Instead of initializing the parameters $\Theta = \{\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K\}$ randomly, we split the image in K regions/pieces (*i.e.*, for $K = 2$ we consider the upper and lower halves of the image and for $K = 4$ we resplit each upper and lower half into left and right); then, we obtain $\{\mu_1, \ldots, \mu_K\}$ and $\{\Sigma_1, \ldots, \Sigma_K\}$ with moment matching according to the discrete attention weights in the corresponding region, as in the experiments in Section 6.2. The initial mixing coefficients $\{\pi_1, \ldots, \pi_K\}$ can be obtained according to the probability mass in the corresponding region. Then, we run the EM algorithm for weighted data proposed in Subsection 5.3.2 to obtain the final estimates for Θ. Instead of evaluating the log likelihood function (5.16) each iteration to check for convergence, we re-estimate the parameters of the mixture model for a fixed number of iterations, which can be considered an extra hyperparameter. According to (5.5), we compute K individual 2D continuous softmax attention mechanisms in order to obtain the context representations $\{c_1, ..., c_K\}$. The final context is a mixture of the context representations for each component, $c = \sum_{k=1}^{K} \pi_k \; c_k$.

test-MCA. We use a model trained with unimodal continuous attention (2D continuous softmax with $N = 100$, from Table 6.2) and, at test time, consider multimodal distributions. We consider models with a number of components in the range $K \in \{K_{\min}, \ldots, K_{\max}\}$, with $K_{\min} = 1$ and $K_{\max} = 5$. For $K = 1$ we use the same setup as in the previous section. For $K > 1$ we follow the procedure described in Subsection 5.3.4 and consider multiple random initializations for each K (we opt to use 3 different initializations, $i \in \{1, 2, 3\}$). We use the EM algorithm for weighted data to obtain different parameter estimates $\hat{\Theta}_{Ki}$. All in all, we consider $1+3\times4 = 13$ estimates $\hat{\Theta} \in \{\hat{\Theta}_1, \hat{\Theta}_{21}, \hat{\Theta}_{22}, \hat{\Theta}_{23}, \ldots, \hat{\Theta}_{51}, \hat{\Theta}_{52}, \hat{\Theta}_{53}\}$, and choose the model that minimizes the model selection criterion (5.20),

$$\mathcal{C}(\hat{\Theta}, K) = -2 \ln (X|\hat{\Theta}) + \lambda \; K.$$

If the optimum number of components $K^\star > 1$, the final context is computed as a mixture of the context representations for each component, $c = \sum_{k=1}^{K^\star} \pi_k \; c_k$, where each c_k is obtained through an individual 2D continuous softmax attention mechanism. In fact, this can be thought of as a 2 step procedure: for each example, we first obtain $K^\star$ and then compute $K^\star$-MCA with a random initialization.

Table 6.3: Accuracies of different models on the *test-dev* and *test-standard* splits of VQA-v2. For the continuous attention models we used $N = 100$ Gaussian RBFs $\mathcal{N}(t; \tilde{\mu}, \tilde{\Sigma})$, with $\tilde{\mu}$ linearly spaced in $[0, 1]^2$ and $\tilde{\Sigma} = 0.001 \cdot \mathrm{I}$. For the MCA models, we used the following number of iterations: 2,5,20,20,20.

ATTENTION	λ	Test-Dev				Test-Standard			
		Yes/No	Number	Other	Overall	Yes/No	Number	Other	Overall
Discrete	-	**83.40**	43.59	55.91	65.83	83.47	42.99	**56.33**	66.13
2D continuous softmax	-	**83.40**	**44.80**	55.88	65.96	**83.79**	44.33	56.04	**66.27**
2D continuous sparsemax	-	83.10	44.12	55.95	65.79	83.38	43.91	56.14	66.10
2-MCA	-	83.35	44.28	**56.07**	**65.97**	83.59	43.65	56.24	66.21
4-MCA	-	83.39	43.52	55.96	65.85	83.72	43.47	56.03	66.14
test-MCA	10	83.30	44.60	55.81	65.86	83.76	44.08	56.00	66.21
test-MCA	100	83.35	44.75	55.86	65.92	83.77	44.28	56.02	66.25
test-MCA	500	83.39	44.75	55.87	65.95	**83.79**	**44.36**	56.04	**66.27**

Table 6.4: Optimum number of components when using $K^\star$-MCA for different values of λ (in percentage of examples).

λ	$K^\star = 1$	$K^\star = 2$	$K^\star = 3$	$K^\star = 4$	$K^\star = 5$
10	58.58	20.81	8.02	6.77	5.82
100	84.87	11.22	2.51	1.00	0.41
500	96.17	3.30	0.40	0.10	0.03

6.3.2 Results and attention visualization

Results. The results in Table 6.3 show the accuracies for all the attention models – we include the best results from Table 6.2 for comparison purposes. Again, the results are very similar, suggesting that there is no clear gain in terms of accuracy when using MCA models to answer the questions in the VQA-v2 dataset. Note, however, that these models also use less basis functions than image regions ($N \ll L = 14 \times 14$).

In Section 6.2 we obtained slightly better results when using 2D continuous softmax (unimodal continuous attention) instead of discrete attention. We argued that the former was more focused and, even in counting questions that require to focus on objects that are far from each other, continuous attention ellipses would become as wide as necessary (most of the times). With MCA, the results are almost the same as with unimodal attention for $K = 2$ and slightly worse for $K = 4$. Although we can now identify different regions of interest in images (attention focuses), in terms of accuracy, for VQA, that seems not to be a big advantage.

When using test-MCA, an optimum value of components $K^\star$ is chosen to answer each question. Table 6.4 shows that the percentage of examples in which $K^\star = 1$ increases with λ, as expected from (5.20). From Table 6.3 it is possible to see that the overall accuracy also increases with these, suggesting that a small value of K is better for the model's accuracy.

Figure 6.9: Attention maps generated when answering the question: **How many planes have blue as their main body colour?** Top left: 2D continuous softmax ($N = 100$). Top right: MCA-2. Bottom left: MCA-4. Bottom right: MCA-test ($\lambda = 500$).

Attention visualization. More interesting than the results in terms of accuracy is to look at the attention densities generated by each model. Figure 6.9 shows an example in which the models are asked how many planes have blue as their main body colour. There are 2 blue planes – the leftmost and the rightmost ones. When using a continuous attention model such as 2D continuous softmax, the attention density corresponds to a bivariate Gaussian and the region of interest is correctly identified. However, due to its unimodal nature, it attributes a lot of probability mass to the yellow plane's positions, that corresponds to the central region in between the 2 blue planes. Although a similar situation appears to happen when using the model with MCA-2 attention, the difference in the values of the presented contours shows that the attention density is more spread across the 5 planes instead of being concentrated in the yellow one. Both MCA-4 and MCA-test (in this example, $K^\star = 3$) are not only capable of identifying 2 blue planes but also to "isolate" its positions, suggesting that these models are able to produce more flexible attention densities, while enjoying the advantages of modeling attention as a continuous function.

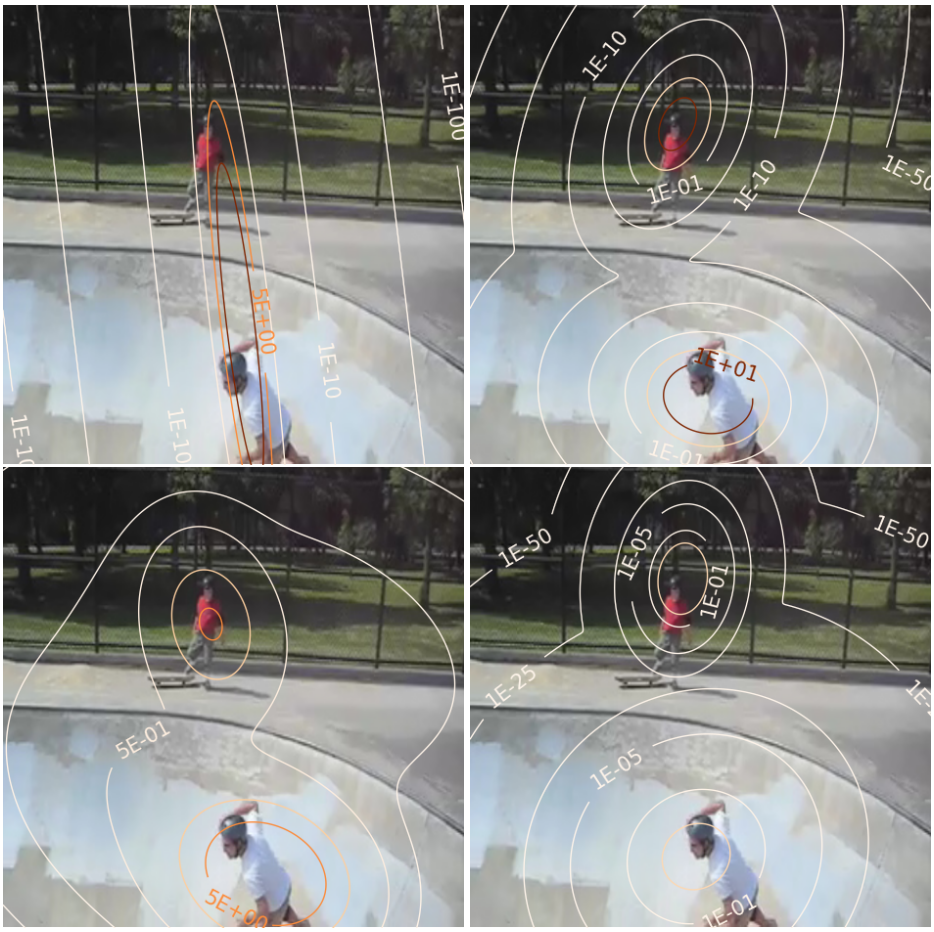

Figure 6.10: Attention maps generated when answering the question: **How many people are wearing helmets?** Top left: 2D continuous softmax ($N = 100$). Top right: MCA-2. Bottom left: MCA-4. Bottom right: MCA-test ($\lambda = 500$).

We now revisit the second example in Figure 6.8, where we stated that the ellipses generated by unimodal attention mechanisms were capable of becoming as wide as necessary. Figure 6.10 shows the attention maps generated by multimodal continuous attention mechanisms when answering the same question (How many people are wearing helmets?). It is clearly seen that these models are looking to the right regions when answering the question. One could anticipate that the model with more components (MCA-4) would become too unfocused due to the higher level of freedom; however, this is not the case: the mixing coefficients of 2 of the components become so small that the attention density has only 2 relevant peaks, identifying 2 people as it should do.

Figure 6.11 shows the same example of Figure 6.4, in which the models are asked what is the woman looking at. Remember that discrete attention was too scattered, possibly mistaking the lamp with a TV screen; using 2D continuous softmax/sparsemax solved the issue by considering continuous and more focused attention densities. When using continuous attention models with more that one mode (*e.g.*, MCA-2 and MCA-4) the attention maps tend to become less focused, putting some probability mass in the region where the lamp is. This can be solved by using test-MCA that selects the optimum number of components for each example – in this case, the best option is to consider an attention distribution with just 1 mode, which corresponds to 2D continuous softmax attention as before.

Figure 6.11: Attention maps generated when answering the question: **What is the woman looking at?** Top left: 2D continuous softmax ($N = 100$). Top right: MCA-2. Bottom left: MCA-4. Bottom right: MCA-test ($\lambda = 500$).

Chapter 7

Conclusions

To wrap up, in this chapter, we start by summarizing our work, in Section 7.1. Afterwards, we present some interesting avenues for future research, in Section 7.2. Finally, we discuss the broader impact of our work, including possible societal consequences, both positive and negative, in Section 7.3.

7.1 Achievements

Over the course of this thesis, we took an in-depth look at current attention mechanisms for vision tasks and addressed some of their known issues. We focused on fully differentiable transformations that can be plugged in deep models and trained end-to-end with backpropagation.

A key component of attention mechanisms is the transformation that maps scores into probabilities – usually a discrete transformation. On the contrary, we presented continuous alternatives based on the extension of regularized prediction maps, originally defined on finite domains, to arbitrary measure spaces. By choosing a regularization function based on the Tsallis α-entropies, we constructed 2D continuous α-entmax attention mechanisms. For $\alpha = 1$ we obtained 2D continuous softmax attention, where the attention density is a Gaussian distribution. For $\alpha = 2$ (2D continuous sparsemax), the attention density becomes a truncated paraboloid distribution with sparse support. We derived their Jacobians, allowing for efficient forward and backward propagation (Chapters 3 and 4).

A unimodal attention distribution may not be enough for certain applications where the relevant features are not located in contiguous regions of an image. As a natural follow-up, we proposed multimodal continuous attention by using mixtures of unimodal attention densities (Chapter 5). These novel attention mechanisms enjoy some of the properties of their unimodal counterparts, while they are able to generate more flexible attention maps and thus can model more complex attention distributions. The backpropagation step for the multimodal case is simple, since this decomposes into a linear combination of unimodal attention mechanisms, each of which has a simple or closed-form Jacobian.

Finally, we performed experiments on Visual Question Answering with promising results (Chapter 6). Continuous attention allowed for obtaining smooth and interpretable attention maps that are difficult to generate with discrete attention based models.

7.2 Future work

The research we have carried out opens up interesting possibilities of future work, both in terms of theoretical developments and more efficient implementations. We discuss both short-term and long-term directions. The former are more accessible while the latter may benefit from our work but require some conceptual modifications.

Multimodal continuous attention for other vision tasks. For VQA, we obtained small improvements in terms of accuracy when using 2D continuous attention (unimodal) over discrete models. Although we were able to obtain better attention maps with multimodal continuous attention, we did not observe relevant improvements in the model's predictive capacity. An interesting direction for future research consists in using multimodal continuous attention for other vision tasks that could benefit more from it. For instance, the state-of-the-art on some visual counting tasks is an attention-based model that uses grid features and therefore could work with multimodal continuous attention [66].

Truncated paraboloid RBFs for 2D continuous sparsemax attention. We obtained an image representation by parametrizing the value function as $V_B(t) = B\psi(t)$, where $\psi(t)$ were Gaussian RBFs. Another option is to consider truncated paraboloid RBFs for 2-entmax continuous attention (sparsemax).

Transformer models with reduced time/memory complexity. Transformer models have demonstrated impressive results in a wide variety of tasks, from NLP and speech to vision. However, its self-attention module processes L-length inputs with a quadratic memory and time complexity $\mathcal{O}(L^2)$, becoming hard to train in practice. In the recent months, a wide spectrum of new transformer models has been proposed to tackle this problem [67, 68]. An interesting direction for future research is to consider continuous transformer models where the self-attention would involve $\mathcal{O}(N^2)$ operations, with N being the number of basis functions that define the value function. As $N << L$, this would be much cheaper than the $\mathcal{O}(L^2)$ requirement of the standard transformer model.

The case with truncated paraboloids and varying number of modes. Mixtures of exponential family distributions (and Gaussians, in particular) are well studied; they have many interesting properties and, one of them, is that of having constant support within the same family. On the other hand, we have seen that α-sparse families (for $\alpha > 1$) abandon that property, having varying support (light tails). A possible way of future research lies in considering mixtures of sparse family distributions, including mixtures of truncated paraboloids; the case where different components have disjoint supports seems particularly interesting to explore. Moreover, if the mixing coefficients $\pi \in \Delta^K$ in (5.3) are obtained from a discrete transformation that produces sparse results (*e.g.*, sparsemax), the number of components in a mixture may vary (K becomes a maximum number of components, but fewer may be selected depending on the input data). This approach eliminates the need for a model selection criterion such as in (5.20), making the training of multimodal attention models with varying number of modes easier.

7.3 Broader impact

We can now discuss the broader impact our work may have, including possible societal consequences, both positive and negative. While we could discuss the impact of visual attention in some specific tasks, we choose to broaden our focus so as to consider the longer term impact of our work. Our methods are not yet tested in broader applications and our contribution is mainly theoretical; thus, this discussion is mainly speculative.

It is well known that interpretability of neural network models can be essential to better discover any ethically harmful biases that exist in both the data or the model itself [69, 70]. However, current state-of-the-art models use discrete softmax attention, whose interpretative ability has been questioned in previous work [71, 72]. On the contrary, our continuous attention models may lead to more interpretable decisions via smoother and less scattered attention maps. Moreover, the variance term of a Gaussian or truncated paraboloid continuous attention density can be possibly thought of as a measure of the model's confidence about where it should attend.

While advances in CV have led to considerable technological improvements, they also raise a great deal of ethical concerns that should not be neglected. Although it might be difficult to control how new technologies are used downstream, it is imperative to study their potential societal implications as they are developed, *e.g.*, by carrying out user studies before deploying such systems. For CV applications, ethical concerns hold over certain intelligent systems, with reports of discrimination [73] and privacy violations [74]. For instance, the VQA-v2 dataset used in the VQA experiments uses COCO images, which have documented biases [75]. 2D continuous attention holds the promise to scale to larger and multi-resolution images, which may, in the longer term, be deployed in such undesirable domains.

Although our models share the same problem of other Deep Learning models in terms of energy consumption [76], there is nothing specific about our research that poses increased environmental concerns. The computational requirements for training systems with 2D continuous attention do not seem considerably higher than the ones which use discrete attention; although, as discussed in the previous section, there are interesting avenues for future research that hold promise to reduce the computational cost by using fewer basis functions.

Bibliography

[1] R. A. Rensink. The Dynamic Representation of Scenes. *Visual Cognition*, 7(1-3):17–42, 2000. doi: 10.1080/135062800394667. URL https://doi.org/10.1080/135062800394667.

[2] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998. doi: 10.1109/34.730558.

[3] H. Larochelle and G. E. Hinton. Learning to combine foveal glimpses with a third-order Boltzmann machine. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1243–1251. Curran Associates, Inc., 2010. URL http://papers.nips.cc/paper/4089-learning-to-combine-foveal-glimpses-with-a-third-order-boltzmann-machine.pdf.

[4] V. Mnih, N. Heess, A. Graves, and koray Kavukcuoglu. Recurrent Models of Visual Attention. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention.pdf.

[5] A. C. Schütz, D. I. Braun, and K. R. Gegenfurtner. Eye movements and perception: A selective review. *Journal of Vision*, 11(5):9–9, 09 2011.

[6] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1462–1471, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/gregor15.html.

[7] D. Bahdanau, K. H. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–15, 2015.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Sys-*

tems 30, pages 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[9] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2048–2057, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/xuc15.html.

[10] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48, 2016.

[11] S. Sharma, R. Kiros, and R. Salakhutdinov. Action recognition using visual attention. In *Neural Information Processing Systems (NIPS) Time Series Workshop*, December 2015. URL http://shikharsharma.com/publications/pdfs/action-recognition-using-visual-attention-nips2015.pdf.

[12] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang. Progressive attention guided recurrent network for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[13] S. Vashishth, S. Upadhyay, G. S. Tomar, and M. Faruqui. Attention interpretability across nlp tasks, 2019.

[14] S. Wiegreffe and Y. Pinter. Attention is not not Explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1002. URL https://www.aclweb.org/anthology/D19-1002.

[15] A. Galassi, M. Lippi, and P. Torroni. Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–18, 2020. doi: 10.1109/TNNLS.2020.3019893.

[16] P. Santana, L. Correia, M. Guedeszy, and J. Baratay. Visual attention and swarm cognition towards fast and robust off-road robots. In *2011 IEEE International Symposium on Industrial Electronics*, pages 2255–2260, 2011. doi: 10.1109/ISIE.2011.5984512.

[17] S. Zhang, L. Zhuo, H. Zhang, and J. Li. Object tracking in unmanned aerial vehicle videos via multifeature discrimination and instance-aware attention network. *Remote Sensing*, 12(16):2646, Aug 2020. ISSN 2072-4292. doi: 10.3390/rs12162646. URL http://dx.doi.org/10.3390/rs12162646.

[18] T. Hoeser and C. Kuenzer. Object detection and image segmentation with deep learning on earth observation data: A review-part i: Evolution and recent trends. *Remote Sensing*, 12(10):1667,

May 2020. ISSN 2072-4292. doi: 10.3390/rs12101667. URL http://dx.doi.org/10.3390/rs12101667.

[19] T. Hoeser, F. Bachofer, and C. Kuenzer. Object detection and image segmentation with deep learning on earth observation data: A review—part ii: Applications. *Remote Sensing*, 12(18): 3053, Sep 2020. ISSN 2072-4292. doi: 10.3390/rs12183053. URL http://dx.doi.org/10.3390/rs12183053.

[20] J. M. Haut, M. E. Paoletti, J. Plaza, A. Plaza, and J. Li. Visual attention-driven hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57:8065–8080, 2019.

[21] L. Xie, S. Wang, A. Markham, and N. Trigoni. Towards monocular vision based obstacle avoidance through deep reinforcement learning. In *RSS 2017 workshop on New Frontiers for Deep Learning in Robotics*, 2017.

[22] R. Polvara, M. Patacchiola, S. Sharma, J. Wan, A. Manning, R. Sutton, and A. Cangelosi. Toward end-to-end control for uav autonomous landing via deep reinforcement learning. In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 115–123, 2018. doi: 10.1109/ICUAS.2018.8453449.

[23] S. L. Brunton, J. N. Kutz, K. Manohar, A. Y. Aravkin, K. Morgansen, J. Klemisch, N. Goebel, J. Buttrick, J. Poskin, A. Blom-Schieber, T. Hogan, and D. McDonald. Data-driven aerospace engineering: Reframing the industry with machine learning, 2020.

[24] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018. ISSN 10636919. doi: 10.1109/CVPR.2018.00636.

[25] P. H. Martins, V. Niculae, Z. Marinho, and A. Martins. Sparse and structured visual attention, 2020.

[26] M. Blondel, A. F. T. Martins, and V. Niculae. Learning with fenchel-young losses. *Journal of Machine Learning Research*, 21:35:1–35:69, 2020.

[27] A. F. T. Martins, A. Farinhas, M. Treviso, V. Niculae, M. A. T. Figueiredo, and P. M. Q. Aguiar. Sparse and continuous attention mechanisms. In *Proc. NeurIPS*, 2020. URL https://arxiv.org/abs/2006.07214.

[28] C. Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of Statistical Physics*, 52: 479–487, 07 1988. doi: 10.1007/BF01016429.

[29] A. F. T. Martins, A. Farinhas, M. Treviso, V. Niculae, M. A. T. Figueiredo, and P. M. Q. Aguiar. Sparse and continuous attention mechanisms *(to be submitted)*. 2020.

[30] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[31] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. 323(6088):533–536, Oct. 1986. doi: 10.1038/323533a0.

[33] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[34] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

[35] T. Luong, H. Pham, and C. D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://www.aclweb.org/anthology/D15-1166.

[36] A. Martins and R. Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1614–1623, New York, New York, USA, 20–22 Jun 2016. PMLR. URL http://proceedings.mlr.press/v48/martins16.html.

[37] B. Peters, V. Niculae, and A. F. T. Martins. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1146. URL https://www.aclweb.org/anthology/P19-1146.

[38] P. Sermanet, A. Frome, and E. Real. Attention for fine-grained categorization. *CoRR*, abs/1412.7054, 2015.

[39] P. R. Halmos. *Measure Theory*, volume 18. Springer, 2013.

[40] M. Blondel, A. F. T. Martins, and V. Niculae. Learning Classifiers with Fenchel-Young Losses: Generalized Entropies, Margins, and Algorithms. 89, 2018. URL http://arxiv.org/abs/1805.09717.

[41] S.-i. Amari and A. Ohara. Geometry of q-exponential family of probability distributions. *Entropy*, 13 (6):1170–1185, 2011.

[42] C. Gini. *Variabilità e mutabilità: contributo allo studio delle distribuzioni e delle relazioni statistiche. [Fasc. I.]*. Studi economico-giuridici pubblicati per cura della facoltà di Giurisprudenza della R. Università di Cagliari. Tipogr. di P. Cuppini, 1912.

[43] O. Barndorff-Nielsen. *Information and Exponential Families in Statistical Theory*. John Wiley & Sons, 2014.

[44] J. Naudts. The q-exponential family in statistical physics. *Central European Journal of Physics*, 7 (3):405–413, 2009.

[45] H. Matsuzoe and A. Ohara. Geometry for q-exponential families. In *Recent Progress in Differential Geometry and its Related Fields*, pages 55–71. World Scientific, 2012.

[46] B. Muzellec and M. Cuturi. Generalizing point embeddings using the wasserstein space of elliptical distributions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 10258–10269, Red Hook, NY, USA, 2018. Curran Associates Inc.

[47] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2.

[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[49] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[50] G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. Wiley New York, 1997. ISBN 0471123587.

[51] I. D. Gebru, X. Alameda-Pineda, F. Forbes, and R. Horaud. Em algorithms for weighted-data clustering with application to audio-visual scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(12):2402–2415, 2016. doi: 10.1109/TPAMI.2016.2522425.

[52] M. A. T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 24:381–396, 2000.

[53] G. J. McLachlan and D. Peel. *Finite mixture models*. Wiley Series in Probability and Statistics, New York, 2000.

[54] G. Schwarz. Estimating the Dimension of a Model. *Annals Statist.*, 6:461–464, 1978.

[55] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974. doi: 10.1109/TAC.1974.1100705.

[56] J. Rissanen. *Stochastic Complexity in Statistical Inquiry Theory*. World Scientific Publishing Co., Inc., USA, 1989. ISBN 9971508591.

[57] L. Xu and M. I. Jordan. On convergence properties of the em algorithm for gaussian mixtures. *Neural Comput.*, 8(1):129–151, Jan. 1996. ISSN 0899-7667. doi: 10.1162/neco.1996.8.1.129. URL https://doi.org/10.1162/neco.1996.8.1.129.

[58] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[59] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf.

[60] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[61] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:770–778, 2016. ISSN 10636919. doi: 10.1109/CVPR.2016.90.

[62] Z. Yu, J. Yu, Y. Cui, D. Tao, and Q. Tian. Deep modular co-attention networks for visual question answering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:6274–6283, 2019. ISSN 10636919. doi: 10.1109/CVPR.2019.00644.

[63] 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. *International Journal of Computer Vision*, 115(3):211–252, 2015. ISSN 15731405. doi: 10.1007/s11263-015-0816-y.

[64] GloVe: Global vectors for word representation. 2014. ISBN 9781937284961. doi: 10.3115/v1/d14-1162.

[65] D. P. Laurie. Calculation of gauss-kronrod quadrature rules. *Math. Comput.*, 66(219):1133–1145, July 1997. ISSN 0025-5718. doi: 10.1090/S0025-5718-97-00861-2. URL https://doi.org/10.1090/S0025-5718-97-00861-2.

[66] D.-K. Nguyen, V. Goswami, and X. Chen. Revisiting modulated convolutions for visual counting and beyond. *arXiv preprint arXiv:2004.11883*, 2020.

[67] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler. Efficient transformers: A survey. *ArXiv*, abs/2009.06732, 2020.

[68] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, and D. Metzler. Long range arena: A benchmark for efficient transformers, 2020.

[69] T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2018.07.007. URL http://www.sciencedirect.com/science/article/pii/S0004370218305988.

[70] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv: Machine Learning*, 2017.

[71] S. Serrano and N. A. Smith. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1282. URL https://www.aclweb.org/anthology/P19-1282.

[72] S. Jain and B. C. Wallace. Attention is not explanation. *CoRR*, abs/1902.10186, 2019. URL http://arxiv.org/abs/1902.10186.

[73] J. Buolamwini and T. Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In S. A. Friedler and C. Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 77–91, New York, NY, USA, 23–24 Feb 2018. PMLR. URL http://proceedings.mlr.press/v81/buolamwini18a.html.

[74] C. Garvie, G. U. C. on Privacy, Technology, and G. U. L. C. C. on Privacy & Technology. *The Perpetual Line-up: Unregulated Police Face Recognition in America*. Georgetown Law, Center on Privacy & Technology, 2016. URL https://books.google.pt/books?id=uAYjngAACAAJ.

[75] T. Wang, J. Zhao, M. Yatskar, K.-W. Chang, and V. Ordonez. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[76] E. Strubell, A. Ganesh, and A. McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1355. URL https://www.aclweb.org/anthology/P19-1355.

[77] G. B. Folland. How to Integrate A Polynomial Over A Sphere. *The American Mathematical Monthly*, 108(5):446–448, 2001. doi: 10.1080/00029890.2001.11919774. URL https://doi.org/10.1080/00029890.2001.11919774.

Appendix A

Continuous attention with Gaussian RBFs

A.1 Obtaining the parameter Σ from the variance for a multivariate truncated paraboloid distribution

Theorem 1. *Let $\mathcal{TP}\ (t; \mu, \Sigma)$ be a d-dimensional multivariate truncated paraboloid where $t, \mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d \times d} \succ 0$, defined as in [27, Section 2.4]. Let $\lambda = -\left(\frac{\Gamma(d/2+2)}{|2\pi\Sigma|^{1/2}}\right)^{\frac{2}{2+d}}$ be the constant that ensures the distribution normalizes to 1, where $\Gamma(t)$ is the Gamma function. Then, the variance of $\mathcal{TP}$ is related to Σ by*

$$\mathrm{Var}(\Sigma) = f(\Sigma) = -\frac{\lambda\, \Sigma}{\frac{d}{2}+2}. \tag{A.1}$$

Proof. Since the variance does not depend on the location parameter μ, we assume $\mu = 0$ without loss of generality. Starting with

$$\mathrm{Var}(\Sigma) = \int_{\frac{1}{2}t^\top \Sigma^{-1} t \leq -\lambda} \left(-\lambda - \frac{1}{2} t^\top \Sigma^{-1} t\right) t t^\top dt, \tag{A.2}$$

we first use the change of variable formula to reparametrize $s = \Sigma^{-1/2}\ t$ so that we can compute the integral over a ball, $\{s \in \mathbb{R}^d \mid\ ||s||^2 \leq -2\lambda\}$. In practice, this means that $t = \Sigma^{1/2} s$ and we can write:

$$\begin{aligned} \mathrm{Var}(\Sigma) &= \int_{||s||^2 \leq -2\lambda} \Sigma^{1/2}\, s \left(-\lambda - \frac{1}{2}||s||^2\right) s^\top\ \Sigma^{1/2}\ |\Sigma|^{1/2}\ ds \\ &= \Sigma^{1/2} \left(\int_{||s||^2 \leq -2\lambda} s \left(-\lambda - \frac{1}{2}||s||^2\right) s^\top\ ds\right) |\Sigma|^{1/2}\ \Sigma^{1/2}. \end{aligned} \tag{A.3}$$

Consider the change of variable $s = \gamma\ r$, where γ lies on the unit hypersphere of dimension $(d-1)$ and

$r \in [0, (-2\lambda)^{1/2}]$. Note that $||s|| = r$. From the substitution rule for integration, we can write:

$$\text{Var}(\Sigma) = \Sigma^{1/2} \left(\int_0^{(-2\lambda)^{1/2}} r^{(d-1)} \left(\int_{||\gamma||=1} \gamma r \, (-\lambda - \frac{r^2}{2}) r\gamma^\top d\gamma \right) dr \right) |\Sigma|^{1/2} \, \Sigma^{1/2}, \tag{A.4}$$

where $r^{(d-1)}$ is the determinant of the Jacobian matrix of the transformation. Applying Fubini's theorem, we have:

$$\text{Var}(\Sigma) = \Sigma^{1/2} \underbrace{\left(\int_{||\gamma||=1} \gamma\gamma^\top \, d\gamma \right)}_{\frac{1}{d} \, S_{d-1} \, I_d} \left(\int_0^{(-2\lambda)^{1/2}} r \, (-\lambda - \frac{r^2}{2}) r r^{d-1} dr \right) |\Sigma|^{1/2} \, \Sigma^{1/2}, \tag{A.5}$$

where S_{d-1} denotes the surface of a $(d-1)$ hypersphere, given by:

$$S_{d-1} = \frac{d\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}. \tag{A.6}$$

Making use of properties of the Gamma function, we can write:

$$\begin{aligned}
\text{Var}(\Sigma) &= \Sigma^{1/2} \left(\frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} I_d \right) \left(\int_0^{(-2\lambda)^{1/2}} \left(-\lambda r^{d+1} - \frac{r^{d+3}}{2} \right) dr \right) |\Sigma|^{1/2} \, \Sigma^{1/2} \\
&= \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \left(\frac{-\lambda(-2\lambda)^{(\frac{d+2}{2})}}{d+2} - \frac{(-2\lambda)^{(\frac{d+4}{2})}}{d+4} \right) |\Sigma|^{1/2} \, \Sigma \\
&= \frac{\pi^{d/2} 2^{(d/2+1)}}{\Gamma(\frac{d}{2}+1)} \left(\frac{(-\lambda)^{(\frac{d+4}{2})}}{d+2} - \frac{(-\lambda)^{(\frac{d+4}{2})}}{d+4} \right) |\Sigma|^{1/2} \, \Sigma \\
&= \underbrace{\left(\frac{2}{(d+2)\Gamma(d/2+1)} \right)}_{\frac{1}{(d/2+1)\Gamma(d/2+1)} = \frac{1}{\Gamma(d/2+2)}} \frac{2(2\pi)^{d/2}(-\lambda)^{\frac{d+4}{2}}}{d+4} |\Sigma|^{1/2} \, \Sigma. \\
&= \underbrace{\frac{(2\pi)^{d/2}(-\lambda)^{\frac{d+4}{2}} |\Sigma|^{1/2}}{(d/2+2)\,\Gamma(d/2+2)}}_{\Gamma(d/2+3)} \Sigma = \frac{(2\pi)^{d/2}(-\lambda)^{(d/2+2)} |\Sigma|^{1/2}}{\Gamma(d/2+3)} \Sigma.
\end{aligned} \tag{A.8}$$

Finally, noting that:

$$(-\lambda)^{(d/2+1)} = \frac{\Gamma(d/2+2)}{|2\pi\Sigma|^{1/2}} = \frac{\Gamma(d/2+2)}{(2\pi)^{d/2}|\Sigma|^{1/2}}, \tag{A.9}$$

yields:

$$\text{Var}(\Sigma) = -\frac{\lambda\Sigma}{\frac{d}{2}+2}. \tag{A.10}$$

□

[1] First, note that this result must be a diagonal matrix: for $i \neq j$, $\int_{S_d} \gamma_i \gamma_j$ equals zero by symmetry – the integral over half of the sphere cancels out the integral over the opposite half. This means that we only need to be concerned with terms with even exponents like γ_i^2. Furthermore, $\int_{S_d} \gamma_i^2$ does not depend on i by the rotational symmetry of the region of integration and thus the result must be proportional to the identity matrix. Finally, using [77, Theorem] we have

$$\int_{S_d} \gamma_i^2 = \frac{2\Gamma(3/2)\Gamma(1/2)^{d-1}}{\Gamma(d/2+1)} = \frac{1}{d} \, S_{d-1}. \tag{A.7}$$

A.2 Continuous sparsemax in 2D

Consider the case where $D = 2$. For $\phi(t) = [t, tt^\top]$, the distribution $p = \hat{p}_{\Omega_2}[f_\theta]$, with $f_\theta(t) = \theta^\top \phi(t)$, becomes a bivariate truncated paraboloid where μ and Σ are related to the canonical parameters as before, $\theta = [\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}]$. We obtain expressions for the attention mechanism output $\rho_2(\theta) = \mathbb{E}_p[\psi(t)]$ and its Jacobian $J_{\rho_2}(\theta) = \mathrm{cov}_{p,2}(\phi(t), \psi(t))$ that include 1D integrals (simple to integrate numerically), when $\psi(t)$ are Gaussian RBFs, *i.e.*, when each ψ_j is of the form $\psi_j(t) = \mathcal{N}(t; \mu_j, \Sigma_j)$.

Lemma 1. *Let $\mathcal{N}(t, \mu, \Sigma)$ be a D-dimensional multivariate Gaussian, Let $A \in \mathbb{R}^{D\times R}$ be a full column rank matrix (with $R \leq D$), and $b \in \mathrm{R}^D$. Then we have $\mathcal{N}(Au + b; \mu, \Sigma) = \tilde{s}\mathcal{N}(u; \tilde{\mu}, \tilde{\Sigma})$ with:*

$$\begin{aligned}
\tilde{\Sigma} &= (A^\top \Sigma^{-1} A)^{-1} \\
\tilde{\mu} &= \tilde{\Sigma} A^\top \Sigma^{-1}(\mu - b) \\
\tilde{s} &= (2\pi)^{\frac{R-D}{2}} \frac{|\tilde{\Sigma}|^{1/2}}{|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mu - b)^\top P(\mu - b)\right), \quad P = \Sigma^{-1} - \Sigma^{-1} A \tilde{\Sigma} A^\top \Sigma^{-1}.
\end{aligned}$$

If $R = D$, then A is invertible and the expressions above can be simplified to:

$$\begin{aligned}
\tilde{\Sigma} &= A^{-1}\Sigma A^{-\top} \\
\tilde{\mu} &= A^{-1}(\mu - b) \\
\tilde{s} &= |A|^{-1}.
\end{aligned}$$

Proof. The result can be derived by writing:

$$\mathcal{N}(Au + b; \mu, \Sigma) = (2\pi)^{-R/2}|\Sigma|^{-1/2} \exp(-\tfrac{1}{2}(Au + b - \mu)^\top \Sigma^{-1}(Au + b - \mu)), \tag{A.11}$$

and splitting the exponential of the sum as a product of exponentials.

Forward pass. For the forward pass, we need to compute

$$\mathbb{E}_p[\psi_j(t)] = \iint_{\mathbb{R}^2} \left[-\lambda - \frac{1}{2}(t - \mu)^\top \Sigma^{-1}(t - \mu)\right]_+ \mathcal{N}(t; \mu_j, \Sigma_j) dt, \tag{A.12}$$

with

$$\mathcal{N}(t; \mu_j, \Sigma_j) = \frac{1}{2\pi\,|\Sigma_j|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(t - \mu_j)^\top \Sigma_j^{-1}(t - \mu_j)\right), \tag{A.13}$$

and from (3.15):

$$\lambda = -\left(\frac{1}{\pi\sqrt{\det(\Sigma)}}\right)^{\frac{1}{2}}. \tag{A.14}$$

Using Lemma 1 and the change of variable formula (which makes the determinants cancel), we can reparametrize $u = (-2\lambda)^{-\frac{1}{2}}\Sigma^{-\frac{1}{2}}(t - \mu)$ and write this as an integral over the unit circle:

$$\mathbb{E}_p[\psi_j(t)] = \iint_{\|u\|\leq 1} -\lambda(1 - \|u\|^2)\mathcal{N}(u; \tilde{\mu}, \tilde{\Sigma}) du, \tag{A.15}$$

with $\tilde{\mu} = (-2\lambda)^{-\frac{1}{2}}\Sigma^{-\frac{1}{2}}(\mu_j - \mu)$, $\tilde{\Sigma} = (-2\lambda)^{-1}\Sigma^{-\frac{1}{2}}\Sigma_j\Sigma^{-\frac{1}{2}}$. We now do a change to polar coordinates, $u = (r\cos\theta, r\sin\theta) = ar$, where $a = [\cos\theta, \sin\theta]^\top \in \mathbb{R}^{2\times 1}$. The integral becomes:

$$\begin{aligned}\mathbb{E}_p[\psi_j(t)] &= \int_0^{2\pi}\int_0^1 -\lambda(1-r^2)\mathcal{N}(ar;\tilde{\mu},\tilde{\Sigma})r\,dr\,d\theta \\ &= \int_0^{2\pi}\int_0^1 -\lambda r(1-r^2)\tilde{s}\mathcal{N}(r;r_0,\sigma^2)\,dr\,d\theta, \end{aligned} \tag{A.16}$$

where in the second line we applied again Lemma 1, resulting in

$$\begin{aligned}\sigma^2(\theta)\equiv\sigma^2 &= (a^\top\tilde{\Sigma}^{-1}a)^{-1} \\ r_0(\theta)\equiv r_0 &= \sigma^2 a^\top\tilde{\Sigma}^{-1}\tilde{\mu} \\ \tilde{s}(\theta)\equiv\tilde{s} &= \frac{1}{\sqrt{2\pi}}\frac{\sigma}{|\tilde{\Sigma}|^{1/2}}\exp\left(-\frac{1}{2}\tilde{\mu}^\top P\tilde{\mu}\right), \quad P = \tilde{\Sigma}^{-1} - \sigma^2\tilde{\Sigma}^{-1}aa^\top\tilde{\Sigma}^{-1}. \end{aligned} \tag{A.17}$$

Applying Fubini's theorem, we fix θ and integrate with respect to r. We use the fact that, for any $u, v \in \mathbb{R}$ such that $u \le v$:

$$\int_u^v \mathcal{N}(t;0,1) = \frac{1}{2}\left(\operatorname{erf}\left(\frac{v}{\sqrt{2}}\right) - \operatorname{erf}\left(\frac{u}{\sqrt{2}}\right)\right), \tag{A.18}$$

$$\int_u^v t\mathcal{N}(t;0,1) = -\mathcal{N}(v;0,1) + \mathcal{N}(u;0,1), \tag{A.19}$$

$$\int_u^v t^2\mathcal{N}(t;0,1) = \frac{1}{2}\left(\operatorname{erf}\left(\frac{v}{\sqrt{2}}\right) - \operatorname{erf}\left(\frac{u}{\sqrt{2}}\right)\right) - v\mathcal{N}(v;0,1) + u\mathcal{N}(u;0,1), \tag{A.20}$$

and

$$\int_u^v t^3\mathcal{N}(t;0,1) = -\mathcal{N}(v;0,1)(2+v^2) + \mathcal{N}(u;0,1)(2+u^2). \tag{A.21}$$

We obtain a closed from expression for the inner integral:

$$\begin{aligned}F(\theta) &= \int_0^1 r(1-r^2)\mathcal{N}(r;r_0,\sigma^2)\,dr \\ &= (2\sigma^3 + r_0^2\sigma + r_0\sigma)\mathcal{N}\left(\frac{1-r_0}{\sigma};0,1\right) - (2\sigma^3 + r_0^2\sigma - \sigma)\mathcal{N}\left(-\frac{r_0}{\sigma};0,1\right) \\ &\quad -\frac{r_0^3 + (3\sigma^2-1)r_0}{2}\left[\operatorname{erf}\left(\frac{1-r_0}{\sqrt{2}\sigma}\right) - \operatorname{erf}\left(-\frac{r_0}{\sqrt{2}\sigma}\right)\right]. \end{aligned} \tag{A.22}$$

The desired integral can then be expressed in a single dimension as

$$\mathbb{E}_p[\psi_j(t)] = -\lambda\int_0^{2\pi}\tilde{s}(\theta)F(\theta), \tag{A.23}$$

which may be integrated numerically.

Backward pass. For the backward pass we need to solve

$$\operatorname{cov}_{p,2}(t,\psi_j(t)) = \iint_E t\mathcal{N}(t;\mu_j,\Sigma_j) - \frac{1}{|E|}\left(\iint_E t\right)\left(\iint_E \mathcal{N}(t;\mu_j,\Sigma_j)\right) \tag{A.24}$$

and

$$\mathrm{cov}_{p,2}(tt^\top, \psi_j(t)) = \iint_E tt^\top \mathcal{N}(t;\mu_j,\Sigma_j) - \frac{1}{|E|}\left(\iint_E tt^\top\right)\left(\iint_E \mathcal{N}(t;\mu_j,\Sigma_j)\right) \tag{A.25}$$

where $E = \mathrm{supp}(p) = \{t \in \mathbb{R}^2 \mid \frac{1}{2}(t-\mu)^\top\Sigma^{-1}(t-\mu) \leq -\lambda\}$ denotes the support of the density p, a region bounded by an ellipse. Note that these expressions include integrals of vector-valued functions and that (A.24) and (A.25) correspond to the first to second and the third to sixth row of the Jacobian, respectively. The integrals that do not include Gaussians have closed form expressions and can be computed as

$$\frac{1}{|E|}\left(\iint_E t\right) = \mu, \qquad \frac{1}{|E|}\left(\iint_E tt^\top\right) = \mu\mu^\top + \frac{\Sigma}{|E|}, \tag{A.26}$$

where $|E|$ is the area of the region E given by

$$|E| = \frac{\pi}{\sqrt{\det\left(\frac{1}{-2\lambda}\Sigma^{-1}\right)}}. \tag{A.27}$$

All the other integrals are solved using the same affine transformation and change to polar coordinates as in the forward pass. Given this, $\tilde{\mu}$, $\tilde{\Sigma}$, a, σ^2, r_0 and $\tilde{s}$ are the same as before. To solve (A.24) we write

$$\iint_E t\mathcal{N}(t;\mu_j,\Sigma_j) = \iint_{\|u\|\leq 1}\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}u + \mu\right)\mathcal{N}(u;\tilde{\mu},\tilde{\Sigma})du \tag{A.28}$$

in polar coordinates,

$$\int_0^{2\pi}\int_0^1 r\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}ar + \mu\right)\tilde{s}\,\mathcal{N}(r;r_0,\sigma^2)dr\,d\theta, \tag{A.29}$$

which can be then expressed in a single dimension as

$$\iint_E t\mathcal{N}(t;\mu_j,\Sigma_j) \quad = \quad \int_0^{2\pi}\tilde{s}(\theta)G(\theta)d\theta, \tag{A.30}$$

with

$$\begin{aligned}
G(\theta) &= \int_0^1 r\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}ar + \mu\right)\mathcal{N}(r;r_0,\sigma^2)\,dr \\
&= \int_{-\frac{r_0}{\sigma}}^{\frac{1-r_0}{\sigma}}(s\sigma + r_0)\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}a(s\sigma+r_0) + \mu\right)\mathcal{N}(r;r_0,\sigma^2)\,ds \\
&= \left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}a\sigma(r_0) + \mu\sigma\right)\mathcal{N}\left(-\frac{r_0}{\sigma};0,1\right) \\
&\quad - \left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}a\sigma(1+r_0) + \mu\sigma\right)\mathcal{N}\left(\frac{1-r_0}{\sigma};0,1\right) \\
&\quad + \frac{1}{2}\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}a(\sigma^2+r_0^2) + \mu r_0\right)\left[\mathrm{erf}\left(\frac{1-r_0}{\sqrt{2}\sigma}\right) - \mathrm{erf}\left(-\frac{r_0}{\sqrt{2}\sigma}\right)\right].
\end{aligned} \tag{A.31}$$

We do the same for

$$\iint_E \mathcal{N}(t;\mu_j,\Sigma_j) = \iint_{\|u\|\leq 1}\mathcal{N}(u;\tilde{\mu},\tilde{\Sigma})du = \int_0^{2\pi}\int_0^1 r\tilde{s}\,\mathcal{N}(r;r_0,\sigma^2)dr\,d\theta, \tag{A.32}$$

which can then be expressed in a single dimension as

$$\iint_E \mathcal{N}(t;\mu_j,\Sigma_j) \;=\; \int_0^{2\pi} \tilde{s}(\theta)H(\theta)d\theta, \tag{A.33}$$

with

$$\begin{aligned} H(\theta) &= \int_0^1 r\mathcal{N}(r;r_0,\sigma^2)\,dr = \int_{-\frac{r_0}{\sigma}}^{\frac{1-r_0}{\sigma}} (s\sigma+r_0)\mathcal{N}(r;r_0,\sigma^2)\,ds \\ &= \sigma\left[\mathcal{N}\left(-\frac{r_0}{\sigma};0,1\right)-\mathcal{N}\left(\frac{1-r_0}{\sigma};0,1\right)\right]+\frac{r_0}{2}\left[\operatorname{erf}\left(\frac{1-r_0}{\sqrt{2}\sigma}\right)-\operatorname{erf}\left(-\frac{r_0}{\sqrt{2}\sigma}\right)\right]. \end{aligned} \tag{A.34}$$

Finally, to solve (A.25) we simplify the integral

$$\begin{aligned} \iint_E tt^\top\mathcal{N}(t;\mu_j,\Sigma_j) &= \iint_{\|u\|\le 1}\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}u+\mu\right)\left((-2\lambda)^{\frac{1}{2}}\Sigma^{\frac{1}{2}}u+\mu\right)^\top\mathcal{N}(u;\tilde{\mu},\tilde{\Sigma})du \\ &= \int_0^{2\pi}\int_0^1 r(r^2A+rB+C)\tilde{s}\,\mathcal{N}(r;r_0,\sigma^2)dr\,d\theta \end{aligned} \tag{A.35}$$

with

$$A=(-2\lambda)\Sigma^{\frac{1}{2}}aa^\top(\Sigma^{\frac{1}{2}})^\top \tag{A.36}$$

$$B=(-2\lambda)^{\frac{1}{2}}\left(\Sigma^{\frac{1}{2}}a\mu^\top+\mu a^\top(\Sigma^{\frac{1}{2}})^\top\right) \tag{A.37}$$

$$C=\mu\mu^\top. \tag{A.38}$$

The integral can then be expressed in a single dimension as

$$\iint_E tt^\top\mathcal{N}(t;\mu_j,\Sigma_j) \;=\; \int_0^{2\pi}\tilde{s}(\theta)M(\theta)d\theta, \tag{A.39}$$

with

$$\begin{aligned} M(\theta) &= \int_0^1 (r^3A+r^2B+rC)\mathcal{N}(r;r_0,\sigma^2)dr \\ &= \int_{-\frac{r_0}{\sigma}}^{\frac{1-r_0}{\sigma}} (s^3\tilde{A}+s^2\tilde{B}+s\,\tilde{C}+\tilde{D})\mathcal{N}(s;0,1)\,ds \\ &= \left[\left(2+\left(-\frac{r_0}{\sigma}\right)^2\right)\tilde{A}-\frac{r_0}{\sigma}\tilde{B}+\tilde{C}\right]\mathcal{N}\left(-\frac{r_0}{\sigma};0,1\right) \\ &\quad -\left[\left(2+\left(\frac{1-r_0}{\sigma}\right)^2\right)\tilde{A}+\frac{1-r_0}{\sigma}\tilde{B}+\tilde{C}\right]\mathcal{N}\left(\frac{1-r_0}{\sigma};0,1\right) \\ &\quad +\frac{1}{2}\left(\tilde{B}+\tilde{D}\right)\left[\operatorname{erf}\left(\frac{1-r_0}{\sqrt{2}\sigma}\right)-\operatorname{erf}\left(-\frac{r_0}{\sqrt{2}\sigma}\right)\right] \end{aligned} \tag{A.40}$$

where

$$\tilde{A}=\sigma^3A,\quad \tilde{B}=\sigma^2(3r_0\,A+B),\quad \tilde{C}=\sigma(3r_0^2\,A+2r_0\,B+C),\quad \tilde{D}=r_0^3\,A+r_0^2\,B+r_0\,C. \tag{A.41}$$

Appendix B

Experimental details

B.1 Computing infrastructure

Our infrastructure consists of 4 machines with the specifications shown in Table B.1. The machines were used interchangeably, and all experiments were executed in a single GPU. Despite having machines with different specifications, we did not observe large differences in the execution time of our models across different machines.

#	GPU	CPU
1.	4 × Titan Xp - 12GB	16 × AMD Ryzen 1950X @ 3.40GHz - 128GB
2.	4 × GTX 1080 Ti - 12GB	8 × Intel i7-9800X @ 3.80GHz - 128GB
3.	3 × RTX 2080 Ti - 12GB	12 × AMD Ryzen 2920X @ 3.50GHz - 128GB
4.	3 × RTX 2080 Ti - 12GB	12 × AMD Ryzen 2920X @ 3.50GHz - 128GB

Table B.1: Computing infrastructure.