

# The Deep Mathematics Behind A.I. and Deep Learning

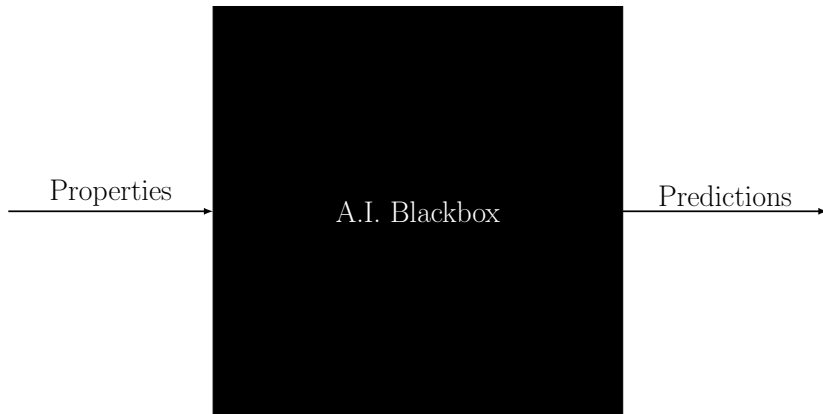
A mathematical generalisation and important theorems in contemporary A.I. and deep learning problems.

Sepehr Saryazdi

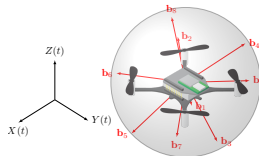
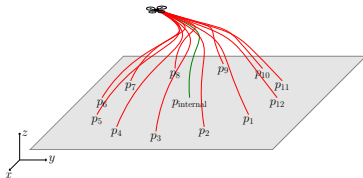
University of Sydney

March 8, 2023

# Why learn the math behind AI?



# Who am I?



# Overview

- 1 Motivation
- 2 The Face-Space Problem
  - Logical Boundaries
  - Fuzzy Logic
  - ROC/Confusion Matrix Adjustment
- 3 A General Mathematical Framework
- 4 Important Theorems
  - Kolmogorov–Arnold Representation Theorem
  - Universal Approximation Theorem
  - Spin Hamiltonian-Loss Correspondence
  - Representer Theorem
- 5 Conclusions

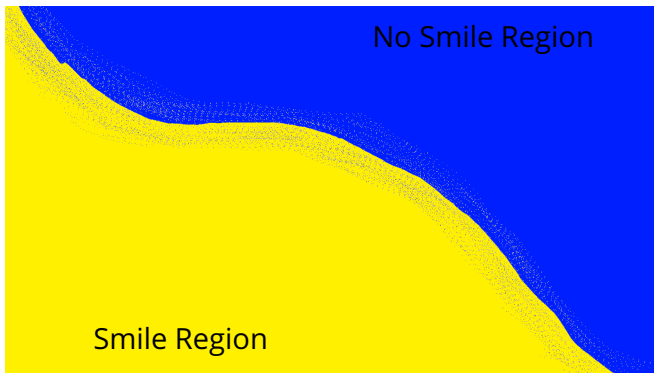
# The Face-Space Problem

Some Subset of the Face-Space

## Logical Boundaries

# Logical Boundaries (Logic Values = $\{0, 1\}$ )

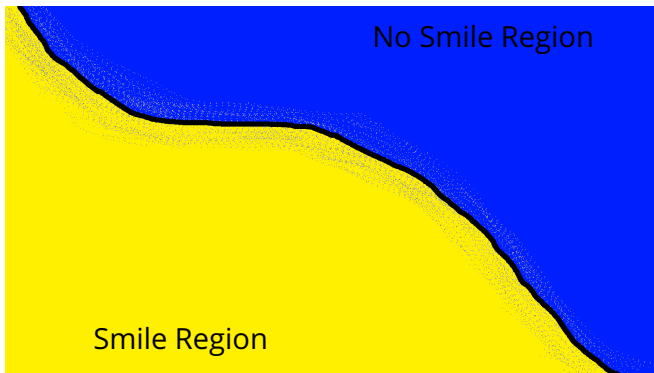
Some Subset of the Face-Space



## Logical Boundaries

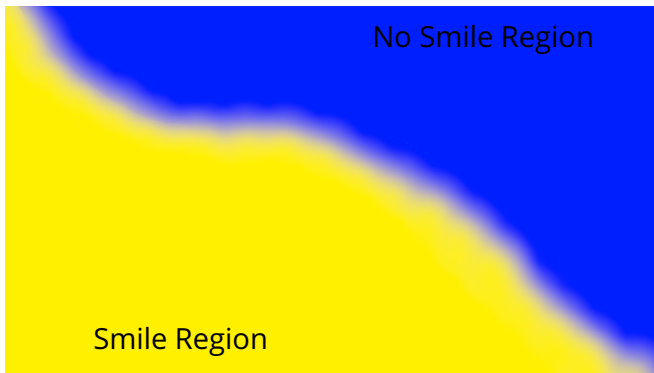
# Logical Boundaries (Logic Values = $\{0, 1\}$ )

Some Subset of the Face-Space



# Fuzzy Logic (Logic Values = $[0, 1] \subseteq \mathbb{R}$ )

Some Subset of the Face-Space





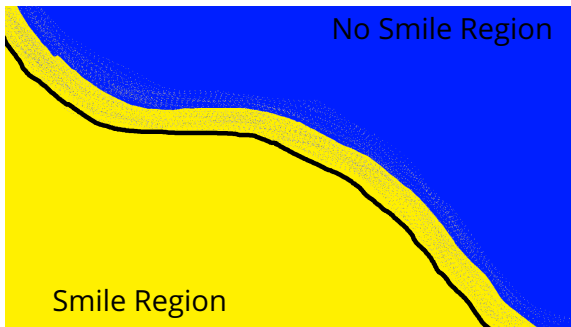
# ROC/Confusion Matrix Adjustment

- ROC (Receiver Operating Characteristic Curve) / Confusion Matrices quantify where things are going wrong for each boundary function choice.



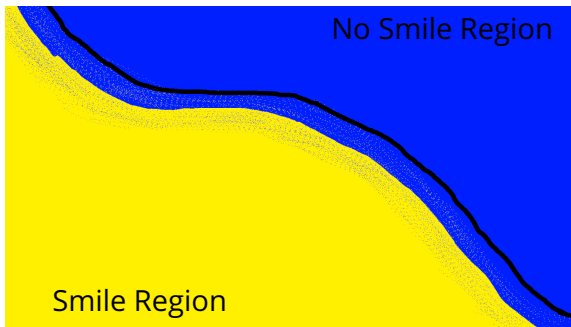
# ROC/Confusion Matrix Adjustment

- ROC (Receiver Operating Characteristic Curve) / Confusion Matrices quantify where things are going right/wrong for each boundary function choice.



# ROC/Confusion Matrix Adjustment

- ROC (Receiver Operating Characteristic Curve) / Confusion Matrices quantify where things are going right/wrong for each boundary function choice.



# A.I. Model Representations

- Let  $P$  be a topological space, representing the model's parameters (i.e.  $P = \mathbb{R}^n$ ).
- Let  $X, Y$  be topological spaces, and let  $C(X, Y)$  be the space of continuous functions from  $X$  to  $Y$ .
- Let a map that takes a parameter to A.I. models for  $X \rightarrow Y$  be denoted by  $\mathcal{M}$ .

## A.I. Model Representations

Every A.I. model may be viewed as a function  $\mathcal{M}$  that maps parameters to continuous functions from  $X \rightarrow Y$ .

$$\mathcal{M} : P \rightarrow C(X, Y)$$

# A.I. Model Representations

## A.I. Model Representations

Every A.I. model may be viewed as a function  $\mathcal{M}$  that maps parameters to continuous functions from  $X \rightarrow Y$ .

$$\mathcal{M} : P \rightarrow C(X, Y)$$

## Example 1: $\mathbb{R} \rightarrow \mathbb{R}$ Linear Model

$$P = \mathbb{R}^2, (a, b) \in P, X = Y = \mathbb{R}$$
$$(\mathcal{M}(a, b))(x) := a + bx$$

# A.I. Model Representations: $\mathbb{R} \rightarrow \mathbb{R}$ Linear Model

$$(\mathcal{M}(a, b))(x) := a + bx$$

# A.I. Model Representations: $\mathbb{R} \rightarrow \mathbb{R}$ Quadratic Model

$$(\mathcal{M}(a, b, c))(x) := a + bx + cx^2$$

# A.I. Model Representations: $\mathbb{R} \rightarrow \mathbb{R}^2 \rightarrow \mathbb{R}$ Neural Network Model

## Example 3: $\mathbb{R} \rightarrow \mathbb{R}^2 \rightarrow \mathbb{R}$ Neural Network Model

$$P = \mathbb{R}^7, (a_1, a_1, b_1, b_2, b_3, c_1, c_2) \in P, X = Y = \mathbb{R}$$

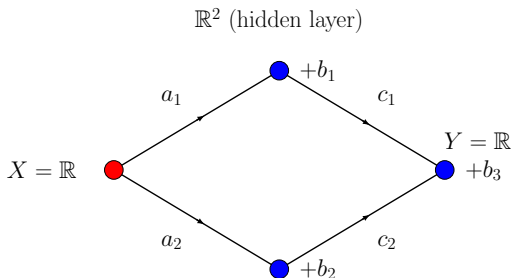
$$(\mathcal{M}(a_1, a_1, b_1, b_2, b_3, c_1, c_2))(x)$$

$$:= (\text{ReLU}((a_1 \ a_2) x + (b_1 \ b_2))) \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + b_3$$

$$\text{ReLU}(x_1, x_2) := (\max(x_1, 0) \ \max(x_2, 0))$$



# A.I. Model Representations: $\mathbb{R} \rightarrow \mathbb{R}^2 \rightarrow \mathbb{R}$ Neural Network Model



# Generalised Loss Functions

- Let  $P, X, Y$  be topological spaces and  $\mathcal{M} : P \rightarrow C(X, Y)$  a function.
- Let  $d : C(X, Y) \times C(X, Y) \rightarrow \mathbb{R}_{\geq 0}$  be a metric.
- Let  $L_f$  for some  $f \in C(X, Y)$  be named a “loss function”.

## Generalised Loss Functions

Define the loss function  $L_f : P \rightarrow \mathbb{R}_{\geq 0}$  to satisfy

$$L_f(p) := d(f, \mathcal{M}(p))$$

# Generalised Loss Functions

Example:  $L^2(X, \mathbb{R}_{\geq 0})$  Loss Function

$$L_f(p) = \int_X \|f(x) - (\mathcal{M}(p))(x)\|^2 dx$$

# Loss Surfaces: $f(x) := \sin x$ , $(M(a, b))(x) := a + bx$ , $L^2$ -Loss Function

# Loss Surfaces: Optimal Parameter Search

- Let  $(p_n)_{n \in \mathbb{N}} \subseteq P$  be a sequence of parameters.

## Optimal Parameter Search

An optimal parameter search is an algorithm where for any  $p_0 \in P$ , it returns a sequence  $(p_n)_{n \in \mathbb{N}} \subseteq P$  such that

$$L_f(p_n) \xrightarrow{n \rightarrow \infty} \inf_{p \in P} L_f(p)$$

- Note: Such a sequence need not have a unique limit, as  $P$  and  $\mathcal{M}$  may over-cover the goal function  $f$ .

## Loss Surfaces: Gradient Descent

# Loss Surfaces: Gradient Descent

- Let  $\gamma \in \mathbb{R}_{>0}$  be named the “learning rate”.
- Let  $\nabla L_f$  denote the gradient function of  $L_f$ .

## Gradient Descent Optimal Parameter Search Algorithm

Assuming  $p_0 \in P$  is given, construct the sequence  $(p_n)_{n \in \mathbb{N}} \subseteq P$  with recursion given by

$$p_{n+1} = p_n - \gamma(\nabla L_f)(p)$$

- In practice, a fixed  $\gamma$  causes either zero movement or large oscillations near local minima of  $L_f$ , so a strictly decreasing sequence  $(\gamma_n)_{n \in \mathbb{N}} \subseteq \mathbb{R}_{>0}$  is often used, or similar ideas like “momentum”.

## Loss Surfaces: Gradient Descent



# Loss Functions: Finite Sampling of $f : X \rightarrow Y$

- In practice, we can only sample  $N \in \mathbb{N}$  finitely many points  $(x_i, y_i) \in X \times Y$  with  $f(x_i) = y_i$ .
- This collapses the  $L^2$  loss function to a finite sum.

## Finite Sampling of $L^2(X, \mathbb{R}_{\geq 0})$ Loss Function (MSE)

$$L_f(p) \approx \frac{1}{N} \sum_{i=1}^N \|f(x_i) - (\mathcal{M}(p))(x_i)\|^2$$

- As  $N \rightarrow \infty$  and assuming the sampling is distributed uniformly over  $X$ , then this will approach the true  $L^2$  loss function.

# Kolmogorov–Arnold Representation Theorem

- This is a solution to the famous 13th Hilbert Problem.
- Colloquially, this theorem says that the “only true continuous multivariable function is the sum”.

## KA Representation Theorem

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  with  $f(\mathbf{x}) := f(x_1, \dots, x_n)$  be a continuous function. Then there exists univariate functions  $\Phi_q : \mathbb{R} \rightarrow \mathbb{R}^m, \phi_{q,p} : \mathbb{R} \rightarrow \mathbb{R}$  such that:

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi_q \left( \sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

# Consequences of KA Representation Theorem

- From this theorem, feature engineering was born.
- This theorem allows us to transform each data column independently before combining them with sums in an A.I. algorithm.

$x_1$	$x_2$	$x_3$	
10	99	85	$0.5x_1 + 2x_2 - x_3$
67	13	8	118
82	48	89	51.5
$\vdots$	$\vdots$	$\vdots$	48
7	76	25	$\vdots$
			130.5

## Universal Approximation Theorem

# Universal Approximation Theorem

- This says that neural networks can be used to approximate any continuous function if  $X$  is closed and bounded.

## Universal Approximation Theorem

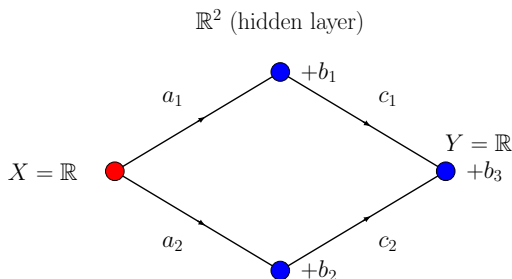
If given  $f : X \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$  and a valid non-polynomial function  $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ , then  $L_f(p) := \|f - \mathcal{M}(p)\|_\infty$  can be made arbitrarily small where

$$(\mathcal{M}(p))(x) := C_p(\sigma \circ (A_p(x) + b))$$

$$p \in P := \mathbb{R}^{k(1+m+n)}, A_p \in \mathbb{R}^{k \times n}, C_p \in \mathbb{R}^{m \times k}, b \in \mathbb{R}^k$$

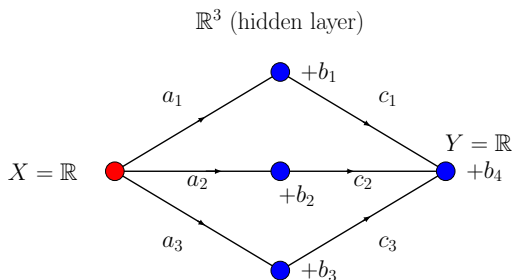
## Universal Approximation Theorem

# Universal Approximation Theorem



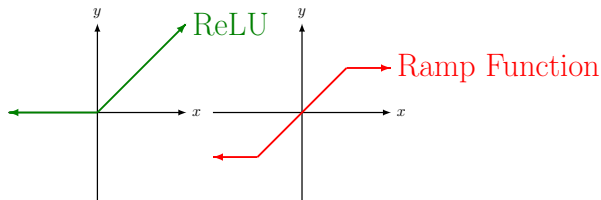
## Universal Approximation Theorem

# Universal Approximation Theorem



# Universal Approximation Theorem: How It Works

- The choice of  $\sigma : \mathbb{R} \rightarrow \mathbb{R}$  was really the crucial part of the entire theorem.
- This is because  $\sigma$  acts as a tool for approximating the 'ramp' function and this can then approximate any continuous function.



# Spin Hamiltonian-Loss Correspondence

## The Fundamental Issue of Machine Learning

How do we know we have found the global minima of  $L_f : P \rightarrow \mathbb{R}_{\geq 0}$ ?

- Fear not! It turns out for large neural networks, any local minima is good enough under reasonable sampling!
- This is thanks to the Spin-Glass Hamiltonian and Neural Network Loss Function Correspondence.



# Spin Hamiltonian-Loss Correspondence: Hamiltonians

- Hamiltonians are a concept from Physics; they quantify how much energy a system is capable of moving around.

## Example: Particle in Gravity Hamiltonian

$$\mathcal{H}(x, v) = \frac{1}{2}mv^2 - \frac{GMm}{|x|}$$

# Spin Hamiltonian-Loss Correspondence: Hamiltonians

## Example: Particle in Gravity Hamiltonian

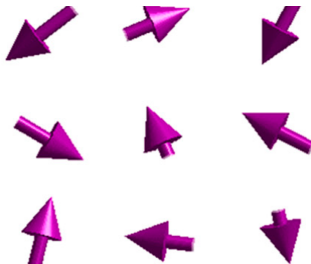
$$\mathcal{H}(x, v) = \frac{1}{2}mv^2 - \frac{GMm}{|x|}$$

- The natural state of the system (when energy moves around the least) occurs whenever  $(x_n, v_n)_{n \in \mathbb{N}} \subseteq \mathbb{R}^2$  is a sequence such that

$$\mathcal{H}(x_n, v_n) \xrightarrow{n \rightarrow \infty} \inf_{(x, v) \in \mathbb{R}^2} \mathcal{H}(x, v)$$

# Spin Hamiltonian-Loss Correspondence: Spin-Glass

- Physicists have been interested in how magnets work for a long time.
- They modelled a block magnet as being made up of locally frozen magnets on the unit sphere  $S^2 \subset \mathbb{R}^3$  that align/repel neighbouring magnets.



# Spin Hamiltonian-Loss Correspondence: Spin-Glass

- Bumping up to  $N$ -dimensional magnets and only considering interactions between  $p \geq 2$  closest neighbouring magnets, we get the  $p$ -Spin Glass Model.

## The Spin-Glass Hamiltonian

$$\mathcal{H}_{N,p}(\sigma) = \frac{1}{N^{(p-1)/2}} \sum_{i_1, \dots, i_p=1}^N J_{i_1, \dots, i_p} \sigma_{i_1} \cdots \sigma_{i_p}$$

$$\sigma = (\sigma_1, \dots, \sigma_N) \in \mathcal{S}^{(N-1)/\sqrt{N}}, \sum_{i=1}^N \sigma_i^2 = N$$

# Spin Hamiltonian-Loss Correspondence: Hamiltonian Local Minima

- With reasonable (but technical) distribution assumptions, one can show that any local minima of  $\mathcal{H}_{N,p}$  is good enough (Auffinger A. et. al., 2011).

## The Spin-Glass Global-Local Minima Proximity

If  $\sigma_{\min} \in \mathcal{S}^{(N-1)/\sqrt{N}}$  is a local minima of  $\mathcal{H}_{N,p}$ , then we expect its Hamiltonian value  $\mathcal{H}_{N,p}(\sigma_{\min})$  to be close to the globally smallest value  $\inf_{\sigma \in \mathcal{S}^{(N-1)/\sqrt{N}}} \mathcal{H}_{N,p}(\sigma)$ .

$$\mathcal{H}_{N,p}(\sigma_{\min}) \approx \inf_{\sigma \in \mathcal{S}^{(N-1)/\sqrt{N}}} \mathcal{H}_{N,p}(\sigma)$$

# Spin Hamiltonian-Loss Correspondence

- It turns out that the loss function  $L_f$  of a neural network (under certain assumptions) can be bijectively mapped to  $\mathcal{H}_{N,p}$  (Choromanska A. et. al, 2015).

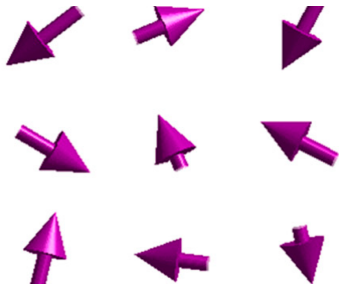
## Spin Hamiltonian-Loss Correspondence

$$\sigma_{\min} \in \mathcal{S}^{(N-1)/\sqrt{N}} \Leftrightarrow p_{\min} \in P$$

$$\mathcal{H}_{N,p}(\sigma_{\min}) \approx \inf_{\sigma \in \mathcal{S}^{(N-1)/\sqrt{N}}} \mathcal{H}_{N,p} \Leftrightarrow L_f(p_{\min}) \approx \inf_{p \in P} L_f(p)$$

# Spin Hamiltonian-Loss Correspondence

- This theorem explains why A.I. works reasonably well even when only approaching one local minima (assuming certain distributional assumptions hold).



## Representer Theorem: Regularisation

- A technical issue that arises, when sampling finitely many points of an infinite space, is the bias-variance tradeoff.
- If you want your model to generalise faster (less variance in predictions), it needs to be more biased!

### $L^2$ Regularisation Bias (Hyper-Parameter $\lambda > 0$ )

$$L_{f,\lambda}(p) := \int_X \|f(x) - (M(p))(x)\|^2 dx + \lambda \|M(p)\|_2^2$$

$$\|M(p)\|_2^2 = \int_X \|(M(p))(x)\|^2 dx$$



# Representer Theorem: Hilbert Spaces

- Due to some nice mathematics,  $L^2$  is what's called a *Hilbert Space*  $\mathcal{H}$ , i.e. it has a notion of 'angles' which allows for a faster way of measuring how similar two functions are.

## Hilbert Space Inner Product

$$\langle f, g \rangle_{\mathcal{H}} = \int_X f(x)g(x)dx = \|f\|_{\mathcal{H}}\|g\|_{\mathcal{H}} \cos \theta$$

# Representer Theorem: Kernels

- Kernels show up everywhere in machine learning due to something called the *Kernel Trick*.
- The idea is to lift our sample points  $(x_i, y_i) \in X \times Y := X \times \mathbb{R}$  into a higher dimensional Hilbert Space, then use the inner product as a 'similarity' measure for any two data points.
- Let  $\kappa : X \times X \rightarrow \mathbb{R}$  be the Kernel which lifts the data via a transformation  $g : X \rightarrow \mathcal{H}$  and evaluates the inner product, defined as

## Kernel Function

$$\kappa(x, y) := \langle g(x), g(y) \rangle_{\mathcal{H}}$$

# Representer Theorem: Reproducing Kernel Hilbert Space

# Representer Theorem: Minimiser A.I. Model

$L^2$  Regularisation Bias (Hyper-Parameter  $\lambda > 0$ )

$$L_{f,\lambda}(p) \approx \frac{1}{N} \sum_{i=1}^N \|f(x_i) - (M(p))(x_i)\|^2 + \lambda \|M(p)\|_2^2$$

- The Representer Theorem beautifully constructs an A.I. model function  $M^*$  which becomes linear in the coefficients and directly keeps track of all the sampled data points, enabling data efficiency.

# Representer Theorem: Minimiser A.I. Model

## Representer Theorem

There exists  $p = (p_1, \dots, p_N) \in P := \mathbb{R}^N$  and  $M^* : \mathbb{R}^N \rightarrow C(X, \mathbb{R})$  which minimises the loss function  $L_{f,\lambda}$  given as

$$(M^*(p))(x) = \sum_{i=1}^N p_i \kappa(x, x_i) \in \text{span}_{\mathbb{R}} \{ \kappa(x, x_i) \mid i \in \{1, \dots, N\} \}$$

# Representer Theorem: Applications

- This shows up in reinforcement learning, where the goal function  $f : \mathcal{S} \rightarrow \mathcal{A}$  is a decision process function with state-space  $\mathcal{S}$  and action-space  $\mathcal{A}$ .
- Since  $\mathcal{S} \subseteq \mathbb{R}^n$  for continuously moving objects, then this is a high dimensional space so making  $f$  linearly represented by a finite function basis generated by the Kernel  $\kappa : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$  is very data-efficient.

# Conclusions

- Overall, A.I. is secretly the study of functions.
- The KA Representation Theorem justifies the univariate transformation of features in feature engineering.
- The Universal Approximation Theorem explains why neural networks are so successful at approximating continuous functions.
- The Spin Hamiltonian-Loss Correspondence explains why local minima of loss in neural networks are close to the global minima.
- The Representer Theorem allows for a linear representation of the A.I. model that utilises the “similarity” measure (the Kernel) about all the data points, so it’s data-efficient.