# Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
*volkan.cevher@epfl.ch*

*Supplementary Lecture: Kernel Methods*

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

**EE-556** (Fall 2024)

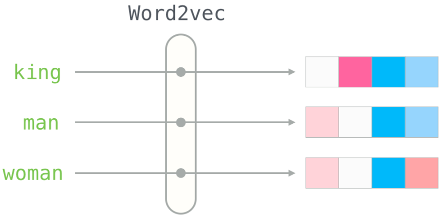
# License Information for Mathematics of Data Slides

## Motivation: Feature embeddings

- Feature embeddings serve to map datasets from a set $\mathcal{A}$ to a more convenient set $\mathcal{H}$.
- For example, if $\mathcal{A}$ is non numerical data, it can be embedded into a subset of $\mathbb{R}^p$.

**Example**

Vector space models of language represent each word with a real-valued vector (Word2vec [7], GloVE [8]).




https://jalammar.github.io/illustrated-word2vec/

# Motivation: Feature embeddings II

∘ If $\mathcal{A}$ is already numerical, it can also be embedded into a more suitable space.

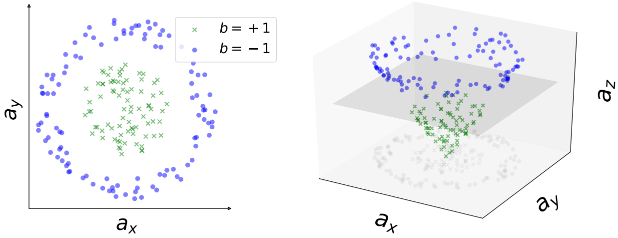



Figure: Non-linearly separable data (left). Linearly separable in $\mathbb{R}^3$ via $\mathbf{a}_z = \sqrt{\mathbf{a}_x^2 + \mathbf{a}_y^2}$ (right).

# Kernels

## Kernels (Informal)

Denote by $\phi : \mathcal{A} \to \mathcal{H}$ the feature embedding that maps data points to a elements of a feature space $\mathcal{H}$.
The kernel $K(\mathbf{a}, \mathbf{b})$ is defined as the inner-product between the feature embeddings:

$$K(\mathbf{a}, \mathbf{b}) := \langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle.$$

**Remarks:**

- Roughly speaking, $K(\mathbf{a}, \mathbf{b})$ represents the similarity between $\mathbf{a}$ and $\mathbf{b}$:

$$K(\mathbf{a}, \mathbf{b}) = \text{“Comparison between } \mathbf{a} \text{ and } \mathbf{b}.\text{”}$$

- The feature embedding $\phi$ helps measure this similarity.
- Examples of $\phi$ are given later in the slides.
- In the sequel, we study feature embeddings in spaces $\mathcal{H}$ with an inner-product.

# Useful definitions

## Inner product for real vector spaces

Let $\mathcal{H}$ be an $\mathbb{R}$-vector space. A binary operation denoted $\langle \cdot, \cdot \rangle_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is said to be an *inner product* if it verifies the following three properties:

- Linearity: For any $f \in \mathcal{H}$, the functions $g \mapsto \langle f, g \rangle_{\mathcal{H}}$ and $g \mapsto \langle g, f \rangle_{\mathcal{H}}$ are linear.
- Symmetry: For any $f, g \in \mathcal{H}$, we have $\langle f, g \rangle_{\mathcal{H}} = \langle g, f \rangle_{\mathcal{H}}$
- Positive definiteness: $\langle f, f \rangle_{\mathcal{H}} = 0 \Leftrightarrow f = 0$.

## Hilbert space

Let $\mathcal{H}$ be an $\mathbb{R}$-vector space that admits an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. The inner-product defines the following norm on $\mathcal{H}$

$$\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}.$$

If $\mathcal{H}$ is complete with respect to this norm, then $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ is called a *Hilbert space*.

**Remarks:**

- ◦ A complete space is a space with "no holes," i.e., all Cauchy sequences converge [14].
- ◦ See Linear Algebra Supplementary Material for more details.

# Positive definite kernels

## From feature embeddings to Kernels [1]

Let $\phi : \mathcal{A} \to \mathcal{H}$ be a feature embedding into a feature space $\mathcal{H}$ the kernel $K$ defined as

$$K(\mathbf{a}, \mathbf{b}) := \langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle_{\mathcal{H}}$$

is a *positive definte kernel*.

## Definition

A mapping $K : \mathcal{A} \times \mathcal{A} \mapsto \mathbb{R}$ is called a *positive definite kernel* if

- For all $\mathbf{a}, \mathbf{b} \in \mathcal{A}$, $K(\mathbf{a}, \mathbf{b}) = K(\mathbf{b}, \mathbf{a})$.
- For any set of points $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathcal{A}$, the matrix

$$\mathbf{K}_{ij} = K(\mathbf{a}_i, \mathbf{a}_j) \text{ is positive semi-definite.}$$

**Remark:** ◦ There exists a rich theory of positive definite kernels: see [11].

## From kernels to embeddings in a feature space

◦ The converse is true: A positive definite kernel $K$ implicitly defines a feature mapping.

### Positive definite Kernels to feature embeddings [1]

Let $K : \mathcal{A} \times \mathcal{A} \mapsto \mathbb{R}$ be a positive definite kernel. Then there exists a feature space $\mathcal{H}$ and a feature mapping $\phi : \mathcal{A} \to \mathcal{H}$ such that

$$K(\mathbf{a}, \mathbf{b}) = \langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle_{\mathcal{H}} .$$

**Observation:** ◦ Defining a positive definite similarity measure between the elements of your dataset

$\equiv$ defining a feature embedding.

# Building positive definite kernels from feature embeddings

## Example

Take $\mathcal{A} = \mathbb{R}$. For any $a \in \mathcal{A}$, we can define the feature embedding $\phi$ that computes the powers of $a$:

$$\phi(a) := \begin{bmatrix} 1 & a & a^2 & \dots & a^5 \end{bmatrix} \in \mathbb{R}^6.$$

The similarity measure $K$, as defined as an inner product between elements of $\mathbb{R}^6$, such as

$$K(a, b) := \langle \phi(a), \phi(b) \rangle_{\mathbb{R}^6} = \phi(a)^\top \phi(b)$$

is a positive definite kernel.

# Embeddings in a feature space: continued example

## Example

To prove this, consider a set of points $a_1, \ldots, a_n \in \mathcal{A}$, then for any $x \in \mathbb{R}^n$, we have

$$\begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix} \begin{bmatrix} & \cdots & \\ \vdots & K(a_i, a_j) & \vdots \\ & \cdots & \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j K(a_i, a_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \langle \phi(a_i), \phi(a_j) \rangle_{\mathbb{R}^6}$$

$$= \left\langle \sum_{i=1}^{n} x_i \phi(a_i), \sum_{j=1}^{n} x_j \phi(a_j) \right\rangle_{\mathbb{R}^6}$$

$$= \| \sum_{i=1}^{n} x_i \phi(a_i) \|_{\mathbb{R}^6}^2 \geq 0.$$

# Possibly infinite dimensional spaces

### Example

Take $\mathcal{A} = (-1, 1)$. For any $a \in \mathcal{A}$, we can define a feature embedding $\phi$ that computes the *all* the powers of $a$:

$$\phi(a) := (a^i)_{i=0}^{\infty}.$$

We can then define the kernel

$$K(a, b) := \langle \phi(a), \phi(b) \rangle_{\mathcal{H}} := \sum_{i=0}^{\infty} \phi(a)_i \phi(b)_i = \frac{1}{1 - ab}.$$

**Observation:** ◦ The feature embedding need not be finite dimensional.

# Positive definite kernels II

∘ New kernels can be constructed from existing kernels.

## Kernel operations

Let $K_1, K_2$ be positive definite kernels (PDKs), then it holds that

- $K_1 + K_2$ is a PDK.
- $K_1 K_2$ is a PDK.
- For any $\lambda \geq 0$, $\lambda K_1$ is a PDK.

Table: Table of commonly used kernels ($K_\nu$ is the Bessel function of order $\nu$, $c, d$ are the bias and degree respectively, $\sigma^2$ is the Gaussian bandwidth, $l, \ell$ are length scales, $T$ is the period)

| **Name** | $\mathcal{A}$ | **Kernel** |
|---|---|---|
| Linear | $\mathbb{R}^p$ | $\mathbf{a}^\top \mathbf{b}$ |
| Polynomial | $\mathbb{R}^p$ | $(\mathbf{a}^\top \mathbf{b} + c)^d$ |
| Gaussian | $\mathbb{R}^p$ | $\exp \frac{-\\|\mathbf{a}-\mathbf{b}\\|^2}{2\sigma^2}$ |
| Matern | $\mathbb{R}^p$ | $\frac{1}{\Gamma(\nu)2^{\nu-1}} \left( \frac{\sqrt{2\nu}}{l} d(\mathbf{a},\mathbf{b}) \right)^\nu K_\nu \left( \frac{\sqrt{2\nu}}{l} d(\mathbf{a},\mathbf{b}) \right)$ |
| Laplace | $\mathbb{R}^p$ | $\exp \frac{-\\|\mathbf{a}-\mathbf{b}\\|}{\sigma}$ |
| Periodic | $\mathbb{R}$ | $\sigma \exp - \frac{2\sin^2(\pi \vert\mathbf{a}-\mathbf{b}\vert/T)}{\ell^2}$ |

# A special feature space of interest: the RKHS

- Given a positive definite kernel $K$, there can be several feature spaces $\mathcal{H}$ such that

$$K(\mathbf{a}, \mathbf{b}) = \langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle_{\mathcal{H}} .$$

- However, there exists *a unique Hilbert space* $\mathcal{H}$ of functions $\mathcal{A} \to \mathbb{R}$ that verify the property below.
- In this case, the embedding $\phi_K$ maps each datapoint $a$ to a function $\phi_K(a) = (b \mapsto K(a, b))$

**Reproducing Kernel Hilbert Space (RKHS) [1]**

Let $K$ be a positive definite kernel. There exists a unique Hilbert space $\mathcal{H} \subset \{\text{functions } \mathcal{A} \to \mathbb{R}\}$ such that the following properties hold:

1. For all $a \in \mathcal{A}$, the embedding $\phi_K(a) = (b \mapsto K(a, b))$ is in $\mathcal{H}$.
2. The reproducing property: $\forall f \in \mathcal{H}, \forall a \in \mathcal{A}, \quad f(a) = \langle \phi_K(a), f \rangle_{\mathcal{H}}$.

The space $\mathcal{H}$ is called a Reproducing Kernel Hilbert Space (RKHS) and $K$ is called the reproducing kernel of $\mathcal{H}$.

# The reproducing property

## Example

We continue the example of the feature map $\phi(a) = \begin{bmatrix} 1 & a & a^2 & \dots & a^5 \end{bmatrix} \in \mathbb{R}^6$.

Now define the function

$$f(a) = x_0 + x_1 a + \cdots + x_5 a^5.$$

This function is a member of a space of functions mapping from $\mathcal{A} = \mathbb{R}^6$ to $\mathbb{R}$. $f$ can be equivalently represented as

$$f(\cdot) = \begin{bmatrix} x_0 & x_1 & \dots & x_5 \end{bmatrix}.$$

With the above notation, we can write

$$f(a) = f(\cdot)^\top \phi(a) := \langle f, \phi(a) \rangle_{\mathbb{R}^6}.$$

# Optimizing over the RKHS

◦ The RKHS is a space of functions that can be used as a hypothesis space for classification, regression, etc.

## ERM over an RKHS

We can consider the problem of minimizing the empirical risk over $\mathcal{H}$:

$$h^\star \in \arg\min_{h \in \mathcal{H}} \left\{ R_n(h) := \frac{1}{n} \sum_{j=1}^{n} L(h(\mathbf{a}_j), b_j) \right\}.$$

**Remarks:**

◦ The space of functions $\mathcal{H}$ is not necessarily finite dimensional!

◦ *A priori*, this optimization problem is not implementable.

# Towards implementation: A first observation

## Observation

Notice that the objective function only depends on the evaluations of $f$ on the points $\mathbf{a}_1, \ldots, \mathbf{a}_n$:

$$R_n(h) := \frac{1}{n}\sum_{j=1}^{n} L(h(\mathbf{a}_i), b_j).$$

**Remarks:**

- Using the reproducing property we can write the objective as a function of inner-products:

$$R_n(h) := \frac{1}{n}\sum_{j=1}^{n} L(\langle h, \phi_K(\mathbf{a}_j)\rangle_{\mathcal{H}}, b_j).$$

- The function $h$ *is only seen through its inner products* with $\{\phi_K(\mathbf{a}_1), \ldots, \phi_K(\mathbf{a}_n)\}$.
- This observation is key to proving the representer theorem (next slide).

# The Representer Theorem (Informal)

## The Representer Theorem [1]

Consider the following regularized version of our optimization problem

$$h^\star \in \arg\min_{h \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{j=1}^{n} L(h(\mathbf{a}_j), b_j) + \lambda \|h\|_{\mathcal{H}}^2 \right\}.$$

where $\lambda > 0$ is some chosen parameter and $L$ *is convex*. This strongly convex problem admits a unique solution $h^\star$. The representer theorem states that this solution lives in the $\mathbf{span}\{\phi_K(\mathbf{a}_1), \ldots, \phi_K(\mathbf{a}_n)\}$. In other words, there exists $\alpha_1, \ldots, \alpha_n$ such that

$$h^\star = \sum_{i=1}^{n} \alpha_i K(\mathbf{a}_i, \cdot),$$

where K is the reproducing kernel of $\mathcal{H}$.

**Remark:**
- The solution $h^\star$ lies in a finite dimensional subspace.
- The resulting is a non-parametric model.

## Implications of the representer theorem

### Tractable formulation

Let $\mathbf{K}$ denote the symmetric, positive semi-definite matrix $\mathbf{K} = (K(\mathbf{a}_i, \mathbf{a}_j))_{i,j}$. Regularized ERM over the RKHS $\mathcal{H}$ reduces to the following finite dimensional problem:

$$\alpha^\star \in \arg\min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{j=1}^{n} L([\mathbf{K}\alpha]_j, b_j) + \lambda \alpha^T \mathbf{K} \alpha \right\}. \quad (1)$$

**Remark:**

- ◦ Any linear model, with feature embedding $\phi$, reduces to this with $(\mathbf{K})_{i,j} = \langle \phi(x_i), \phi(x_j) \rangle$.

## Example: Kernel ridge regression

**Example:** ◦ Take $L$ to be the square loss and the kernel to be the Gaussian kernel $K_\sigma(a,b) = \exp \frac{(a-b)^2}{2\sigma^2}$.

$$\alpha^\star \in \arg\min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \|\mathbf{K}\alpha - \mathbf{b}\|_2^2 + \lambda \alpha^T \mathbf{K}\alpha \right\}.$$

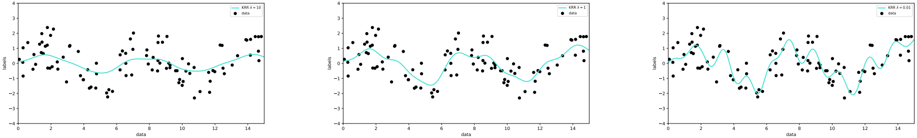



Figure: Kernel ridge regression with decreasing regularization $\lambda$

**Remark:** ◦ The scale parameter $\sigma$ can be chosen through cross validation or clever heuristics, see [4].

## Computational complexity

**Remarks:**

- ∘ The tractable formulation uses the $n \times n$ kernel matrix $\mathbf{K}$.
- ∘ Storage and computational complexity tend to scale with $n^2$ and $n^3$ respectively.
- ∘ This quickly becomes infeasible for large $n$.

### Approximation methods for large $n$

∘ **Nyström method [13]:** substitute $\mathbf{K}$ with a low-rank approximation $\mathbf{K} \approx \mathbf{U}^T\mathbf{U}$ where $\mathbf{U}$ is $r \times n$ for some $r < n$.

- Storage and computational complexity scale with $nr$ and $nr^2$ respectively.

∘ **Random Fourier features [9]:** For some translation invariant kernels, it holds that $K(\mathbf{a}, \mathbf{b}) = \mathbb{E}[\varphi(\mathbf{a})\varphi(\mathbf{b})]$. A Monte Carlo approximation of this expectation is obtained by taking $D$ random samples.

- Storage and computational complexity scale with $nD$ and $nD^2$ respectively.

**Remark:**

- ∘ Large-scale kernel methods are an active research area.
- ∘ See for instance FALKON [10, 6] which extends Nyström methods to achieve optimal rates.

# Deep Learning's recent interest in kernel theory

∘ For convenience, we deviate from the course's notation of neural networks and write $h(\mathbf{a}, \mathbf{x})$ for $h_{\mathbf{x}}(\mathbf{a})$

### First order Taylor approximation

Consider a neural network $h : \mathbf{a} \mapsto h(\mathbf{a}, \mathbf{x})$ initialized with the weights $\mathbf{x}_0$. A first-order Taylor expansion of around $\mathbf{x}_0$ yields

$$h_0(\mathbf{a}, \mathbf{x}) = h_{\mathbf{x}}(\mathbf{a}, \mathbf{x}_0) + \langle \frac{\partial h}{\partial \mathbf{x}}(\mathbf{a}, \mathbf{x}_0), \mathbf{x} - \mathbf{x}_0 \rangle$$

**Remarks:**

∘ $h_0$ is an affine function of the weights $\mathbf{x}$, so it falls under the well understood family of linear models.

∘ If $h$ is close to $h_0$, then theorems for linear models would transfer to $f$.

*Under which conditions is a neural network $h$ is close to $h_0$, its linearization at initialization ?*

## The Neural Tangent Kernel

**Observations:**

- The approximation $h_0$ is a linear model over features of the data: the training samples are embedded in a feature space by the mapping $\mathbf{a} \mapsto \frac{\partial h}{\partial \mathbf{x}}(\mathbf{a}, \mathbf{x}_0)$.
- This feature space can be studied through the kernel

$$K_{\mathbf{x}_0}(\mathbf{a}, \mathbf{b}) = \left\langle \frac{\partial h}{\partial \mathbf{x}}(\mathbf{a}, \mathbf{x}_0), \frac{\partial h}{\partial \mathbf{x}}(\mathbf{b}, \mathbf{x}_0) \right\rangle.$$

- The initialization $\mathbf{x}_0$ is random, so the kernel $K_{\mathbf{x}_0}$ is also random.

**The Neural Tangent Kernel (See [5] Thm 1 or [2] Thm 3.1 for a precise statement.)**

Under appropriately scaled random initialization $\mathbf{x}_0$, the random kernel $K_{\mathbf{x}_0}$ tends to a deterministic kernel $K_\infty$ as the width of the network goes to infinity:

$$\lim_{\text{width}\to\infty} K_{\mathbf{x}_0} = K_\infty$$

This deterministic limiting kernel is called *the Neural Tangent Kernel* (NTK).

- For sufficiently wide networks, we can study this deterministic kernel and derive properties on $h_0$ .

# The NTK in practice

◦ From a theory perspective, NTK theory is used to prove global convergence of first-order methods in neural network training [3].

**Remarks:**
- ◦ Explicit formulas for the NTK of both fully-connected and convolutional architectures have been computed [2].
- ◦ This makes it possible to solve learning tasks with infinitly wide networks.

### Performance of infinitly wide networks on CIFAR-10 [2]

Least-square classification on the CIFAR-10 dataset with the NTK of an 11-layer convolutional network achieves $77\%$ classification accuracy on CIFAR-10. It is still below the accuracy of finite width networks ($> 98\%$).

**Remarks:**
- ◦ Two learning regimes:
  - ◦ Neural network regime: parametric model for feature learning.
  - ◦ Kernel regime: non-parametric model of size growing with number of samples.

# Transformer attention as a Kernel I

∘ Attention mechanism in Transformers can be reformulated as a form of kernel smoothing [12].

∘ Intuition: both kernel learning and Transformers concurrently process all inputs and calculate the similarity between them.

**Attention Mechanism:**

∘ Given a query token $\mathbf{b}_q$ and a set of key tokens $\mathcal{S}_{\mathbf{b}_k}$, the attention output is the kernel smoothing:

$$\text{Attention}(\mathbf{b}_q; M(\mathbf{b}_q, \mathcal{S}_{\mathbf{b}_k})) = \sum_{\mathbf{b}_k \in M(\mathbf{b}_q, \mathcal{S}_{\mathbf{b}_k})} \frac{K(\mathbf{b}_q, \mathbf{b}_k)}{\sum_{\mathbf{b}'_k \in M(\mathbf{b}_q, \mathcal{S}_{\mathbf{b}_k})} K(\mathbf{b}_q, \mathbf{b}'_k)} v(\mathbf{b}_k).$$

# Transformer attention as a Kernel II

**Key Components:**

- **Kernel function** $K(\cdot,\cdot)$: Measures similarity between tokens. Canonical softmax attention is $K(\mathbf{b}_q, \mathbf{b}_k) = \exp\left(\mathbf{b}_q \mathbf{X}_q (\mathbf{b}_k \mathbf{X}_k)^\top\right)$.
- **Set filtering function** $M(\cdot,\cdot)$: Determines relevant tokens. Plays the role of the mask in causal attention.
- **Value function** $v(\cdot)$: Provides the values to be weighted. Usually $v(\mathbf{b}_k) = \mathbf{b}_k \mathbf{X}_v$.

**Remarks:**

- Attention is computed as learnable kernel functions measuring similarity between input tokens.
- Canonical self-attention uses an asymmetric exponential kernel.

# References I

[1] N. Aronszajn.
Theory of reproducing kernels.
*Trans. Amer. Math. Soc.*, 68(3):337–404, 1950.
(Cited on pages 7, 8, 13, and 17.)

[2] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang.
On exact computation with an infinitely wide neural net.
*Advances in Neural Information Processing Systems*, 32, 2019.
(Cited on pages 22 and 23.)

[3] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh.
Gradient descent provably optimizes over-parameterized neural networks.
*arXiv preprint arXiv:1810.02054*, 2018.
(Cited on page 23.)

[4] Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa.
Large sample analysis of the median heuristic.
*arXiv preprint arXiv:1707.07269*, 2017.
(Cited on page 19.)

# References II


[5] Arthur Jacot, Franck Gabriel, and Clément Hongler.
Neural tangent kernel: Convergence and generalization in neural networks.
In *Advances in neural information processing systems*, pages 8571–8580, 2018.
(Cited on page 22.)

[6] Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi.
Kernel methods through the roof: handling billions of points efficiently.
In *Advances in Neural Information Processing Systems*, volume 33, 2020.
(Cited on page 20.)

[7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean.
Distributed representations of words and phrases and their compositionality.
*Advances in neural information processing systems*, 26, 2013.
(Cited on page 3.)

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning.
Glove: Global vectors for word representation.
In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
(Cited on page 3.)

## References III


[9] Ali Rahimi and Benjamin Recht.
Random features for large-scale kernel machines.
In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.
(Cited on page 20.)

[10] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco.
Falkon: An optimal large scale kernel method.
*Advances in neural information processing systems*, 30, 2017.
(Cited on page 20.)

[11] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al.
*Learning with kernels: support vector machines, regularization, optimization, and beyond*.
MIT press, 2002.
(Cited on page 7.)

[12] Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov.
Transformer dissection: An unified understanding for transformer's attention via the lens of kernel.
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 4344–4353, 2019.
(Cited on page 24.)

# References IV


[13] Christopher Williams and Matthias Seeger.
Using the nyström method to speed up kernel machines.
*Advances in neural information processing systems*, 13, 2000.
(Cited on page 20.)

[14] Nicholas Young.
*An introduction to Hilbert space*.
Cambridge university press, 1988.
(Cited on page 6.)