UNIVERSITÀ DEGLI STUDI DI GENOVA

Scuola di Scienze M.F.N.

Dipartimento di Matematica

Università di Genova

Tesi di Laurea Magistrale in Matematica

A Mathematical Perspective on NLP

Markov Chains and Kernel Methods

Relatore:
Prof. Silvia Villa

Candidato:
Luca Fresu

Correlatore:
Prof. Cesare Molinari

Anno Accademico 2024/2025

A Mathematical Perspective on NLP

Luca Fresu

Contents

Introduction

Introduction

The aim of this thesis is to study the problem of next-word prediction in natural language from a rigorous mathematical perspective. In modern language models, a text is represented as a sequence of discrete linguistic units—words or subword tokens—and the task consists in predicting the most probable subsequent word given the preceding context. This problem lies at the core of many applications in Natural Language Processing (NLP), including text generation, language modeling, and dialogue systems.

From a theoretical standpoint, modeling natural language is highly complex, as it involves syntactic structures, semantic relationships, and long-range contextual dependencies. Capturing all these aspects simultaneously typically requires sophisticated neural architectures with millions of parameters. In this thesis, we deliberately adopt a simplified yet mathematically tractable framework, focusing on the assumption that the next word depends only on the immediately preceding word—a *first-order* or *bigram* model.

Under this assumption, the sequence of words can be modeled as a discrete-time, finite-state, homogeneous Markov chain. Each state represents a word in the vocabulary, and the observed text corresponds to a realization of the stochastic process $(w_t)_{t\in\mathbb{N}}$. However, rather than directly estimating transition probabilities, we adopt a *kernel-based regression approach*: we embed words into a continuous vector space using pre-trained word embeddings, and learn a function f such that

$$f(E(w_t)) \approx \mathbb{E}_\pi[E(w_{t+1}) \mid w_t]$$

where $E : \mathcal{V} \to \mathbb{R}^d$ is the embedding function and π is the invariant distribution of the underlying Markov chain.

This probabilistic formulation raises a fundamental theoretical question: the training data consists of *consecutive* word pairs extracted from text, which are *not independent and identically distributed* (i.i.d.)—a standard assumption in statistical learning theory. How can we justify the application of supervised learning methods to such dependent, non-i.i.d. data?

The key insight is provided by *ergodic theory for Markov chains*. Under mild assumptions (irreducibility, aperiodicity, and positive recurrence), the ergodic theorem guarantees that the empirical risk converges almost surely to expectations under the invariant distribution π. This result rigorously justifies the use of empirical risk minimization on sequential data and establishes the theoretical foundation for our approach.

We apply *Kernel Ridge Regression* (KRR) to learn the mapping from current word embeddings to expected next-word embeddings. To handle the computational challenges of large-scale data, we implement the solution via *gradient descent* rather than direct matrix inversion, enabling continuous monitoring of training and validation loss, early stopping, and scalability to larger datasets.

The thesis is structured as follows:

Chapter 1 is devoted to discrete-time Markov chains, where we introduce their fundamental properties, including state classification, recurrence and transience, invariant measures, and asymptotic behavior.

Chapter 2 presents the foundations of supervised learning within the framework of statistical learning theory. We introduce empirical risk minimization and reproducing kernel Hilbert spaces (RKHS). We derive the closed-form solution of Kernel Ridge Regression.

Chapter 3 formulates text generation as a Markov chain problem, where tokens from a vocabulary are mapped to states and the sequence of words in a corpus is modeled as a discrete-time Markov chain. The transition matrix is estimated via maximum likelihood from empirical transition counts, and its statistical validity is supported by the ergodic theorem, which ensures consistency of the estimator. Higher-order language models are implemented by lifting the state space so that a k-order chain becomes a first-order chain on tuples of tokens.

Chapter 4 combines these two perspectives by applying Kernel Ridge Regression to next-word prediction in natural language. We model language as a first-order Markov chain and show how the ergodic theorem resolves the violation of the i.i.d. assumption, enabling rigorous generalization guarantees. We present a complete Python implementation using the TinyStories dataset, including:

- Word embedding via pre-trained Word2Vec models
- Kernel matrix computation with Linear and Gaussian kernels
- Training via gradient descent with early stopping
- Quantitative evaluation and text generation experiments

Experimental results demonstrate the effectiveness and limitations of the first-order Markov assumption. While the model successfully captures local sequential patterns and achieves meaningful prediction accuracy on frequent word transitions, it inherently cannot model long-range dependencies or complex syntactic structures. These limitations highlight the necessity of higher-order models or modern neural architectures (such as transformers) for production-quality language modeling, while our simplified framework provides a mathematically rigorous foundation for understanding the statistical principles underlying next-word prediction.

1 Markov Chains

Introduction

[Fagnola and Sasso, 2017]

We start this chapter with a brief introduction to stochastic processes, which provide a framework for modeling systems that evolve over time with inherent randomness. They are particularly useful for studying phenomena where future states depend on probabilistic rules, making them crucial for understanding dynamical systems subject to random influences.

Definition 1.1 (Stochastic process)**.** *A* stochastic process *is a collection of random variables $(X_t)_{t\in T}$, where T is an index set representing time. Each random variable X_t maps the sample space Ω of a probability space $(\Omega, , \mathcal{F}, \mathbb{P})$ to a measurable space $(I, \mathcal{B})$ usually called the* state space*, which can be either discrete or continuous. The index set T can be:*

- Discrete*, such as $T = \mathbb{N}$, representing discrete-time processes where the evolution occurs at integer time steps, or*
- Continuous*, such as $T = \mathbb{R}^+$, representing continuous-time processes where the evolution is tracked over a continuous interval.*

In both cases the process represents the evolution of the random variable X_t over the parameter $t \in T$, typically representing time.

One important class of stochastic processes is the one of *Markov processes*, which satisfy the *Markov property*: the future state of the process depends only on the current state, not on the sequence of events that preceded it. Formally we have the following definition.

1.1 Definition and basic properties

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and let I be at most countable, always considered as a measurable space with the σ–algebra consisting of all its subsets.

Definition 1.2 (Markov Chain)**.** *A sequence of random variables $(X_n)_{n\geq 0}$ on $(\Omega, \mathcal{F}, \mathbb{P})$ taking values in I is a **Markov chain** with state space I if, for every integer n and every $i_0, \dots, i_n, j \in I$, with $\mathbb{P}(X_n = i) > 0$, one has*

$$\mathbb{P}(X_{n+1} = j \mid X_n = i_n, \dots, X_0 = i_0) = \mathbb{P}(X_{n+1} = j \mid X_n = i_n).$$

*The chain is called **homogeneous** if the two sides do not depend on n.*

In the following we always consider homogeneous Markov chains, so we often omit this adjective. The **transition probabilities** are the non–negative real numbers

$$p_{ij} = \begin{cases} \mathbb{P}(X_{n+1} = j \mid X_n = i), & \text{if } \mathbb{P}(X_n = i) > 0, \\ 1, & \text{if } \mathbb{P}(X_n = i) = 0 \text{ and } i = j, \\ 0, & \text{if } \mathbb{P}(X_n = i) = 0 \text{ and } i \neq j. \end{cases}$$

The matrix $P = (p_{ij})_{i,j\in I}$ is called the **transition matrix**. It satisfies $p_{ij} \geq 0$ and $\sum_j p_{ij} = 1$.

Definition 1.3 (Stochastic Matrix)**.** *A matrix P satisfying the above properties is called **stochastic**.*

The law μ of X_0 defined by $\mu(\{i\}) = \mathbb{P}(X_0 = i)$ is called the **initial law**.

Lemma 1.4. *For every $n, m \in \mathbb{N}$ with $m \geq 1$ and every $i, j \in I$,*

$$\mathbb{P}(X_{n+m} = j \mid X_n = i) = P^{(m)}[i, j],$$

where $P^{(m)}$ denotes the m–th power of the matrix P.

Proof. By the definition of conditional probability and the law of total probability we write

$$\mathbb{P}(X_{n+m} = j \mid X_n = i) = \sum_{k \in I} \mathbb{P}(X_{n+m} = j \mid X_{n+m-1} = k)\, \mathbb{P}(X_{n+m-1} = k \mid X_n = i).$$

By the Markov property,

$$\mathbb{P}(X_{n+m} = j \mid X_{n+m-1} = k) = p_{kj}.$$

Therefore,

$$\mathbb{P}(X_{n+m} = j \mid X_n = i) = \sum_{k \in I} p_{kj}\, \mathbb{P}(X_{n+m-1} = k \mid X_n = i).$$

Iterating this argument m times yields the (i, j) entry of the matrix $P^{(m)}$. □

Proposition 1.5. *Let $(j_k)_{0 \leq k \leq n}$ be a sequence of states and $(m_k)_{0 \leq k \leq n}$ a sequence of times with $0 = m_0 < m_1 < \cdots < m_n$. Then*

$$\mathbb{P}\left(\bigcap_{0 \leq k \leq n} \{X_{m_k} = j_k\} \right) = \mu(j_0) \prod_{k=1}^{n} P^{(m_k - m_{k-1})}[j_{k-1}, j_k].$$

Proof. Using repeated conditioning and the Markov property,

$$\mathbb{P}\left(\bigcap_{k=0}^{n} \{X_{m_k} = j_k\} \right) = \mathbb{P}(X_{m_n} = j_n \mid X_{m_{n-1}} = j_{n-1})\, \mathbb{P}\left(\bigcap_{k=0}^{n-1} \{X_{m_k} = j_k\} \right).$$

By the time-homogeneous Markov property,

$$\mathbb{P}(X_{m_n} = j_n \mid X_{m_{n-1}} = j_{n-1}) = P^{(m_n - m_{n-1})}[j_{n-1}, j_n].$$

Iterating backwards gives

$$\mathbb{P}\left(\bigcap_{k=0}^{n} \{X_{m_k} = j_k\} \right) = \mu_{j_0} \prod_{k=1}^{n} P^{(m_k - m_{k-1})}[j_{k-1}, j_k].$$

□

Classification of states

Definition 1.6 (Accessibility and Communication)**.** *A state j is **accessible** from a state i if there exists $n \geq 1$ such that $P^{(n)}[i, j] > 0$. States i and j **communicate** if each is accessible from the other.*

Definition 1.7 (Class of States)**.** *A **class** of states is a subset $J \subseteq I$ such that any two states in J communicate, and no state outside J communicates with one inside.*

Definition 1.8 (Irreducibility)**.** *A chain is **irreducible** if the only nonempty class is I itself.*

Definition 1.9 (Period)**.** *Let $i \in I$. If the set $\{n \geq 1 : P^{(n)}[i, i] > 0\}$ is nonempty, the **period** of i is the greatest common divisor of its elements. If the period is 1, i is **aperiodic**.*

Definition 1.10 (Recurrence and Transience)**.** *A state i is **recurrent** if*

$$\mathbb{P}\left(\bigcup_{n \geq 1} \{X_n = i\} \,\middle|\, X_0 = i \right) = 1,$$

*and **transient** otherwise. If $p_{ii} = 1$, then i is called **absorbing**.*

Definition 1.11 (First Hitting Time)**.** *The **first hitting time** of j is the random variable*

$$T_j(\omega) = \begin{cases} \min\{n \geq 1 : X_n(\omega) = j\}, & \text{if the set is nonempty,} \\ +\infty, & \text{otherwise.} \end{cases}$$

For $n \geq 1$, set

$$f_{ij}^{(n)} := \mathbb{P}(T_j = n \,|\, X_0 = i)\,, \qquad f_{ij}^{*} := \sum_{n \geq 1} f_{ij}^{(n)}.$$

A state $i \in I$ is said to be *recurrent* if and only if

$$f_{ij}^{*} = 1,$$

Theorem 1.12 (Renewal Equation)**.** *For all $i, j \in I$ and $n \geq 1$,*

$$P^{(n)}[i,j] = \sum_{k=1}^{n} f_{ij}^{(k)}\, P^{(n-k)}[j,j].$$

Proof. Assume $X_0 = i$. Decompose the event $\{X_n = j\}$ according to the first hitting time T_j:

$$\{X_n = j\} = \bigcup_{k=1}^{n} \{T_j = k, X_n = j\}.$$

For $k < n$, conditioning on $X_k = j$ gives

$$\mathbb{P}(X_n = j, T_j = k) = \mathbb{P}(X_n = j \mid X_k = j)\mathbb{P}(T_j = k) = P^{(n-k)}[j,j] f_{ij}^{(k)}.$$

For $k = n$ the probability is $f_{ij}^{(n)}$. Summing over $k = 1, \ldots, n$ yields

$$P^{(n)}[i,j] = \sum_{k=1}^{n} f_{ij}^{(k)} P^{(n-k)}[j,j].$$

□

Theorem 1.13 (Recurrence Criterion)**.** *For a state i, the following are equivalent:*

1. *i is recurrent;*
2. *$\sum_{n=0}^{\infty} P^{(n)}[i,i] = \infty$.*

If i is transient, then

$$\sum_{n=0}^{\infty} P^{(n)}[i,i] = \frac{1}{1 - f_{ii}^{*}}.$$

Proof. Define the generating functions

$$P_{ii}(s) = \sum_{n=0}^{\infty} P^{(n)}[i,i] s^n, \qquad F_{ii}(s) = \sum_{n=1}^{\infty} f_{ii}^{(n)} s^n,$$

which converge at least for $|s| < 1$ because the terms are bounded by 1. From the renewal equation (with $j = i$) we have for every $n \geq 1$,

$$P^{(n)}[i,i] = \sum_{k=1}^{n} f_{ii}^{(k)}\, P^{(n-k)}[i,i].$$

Multiplying by s^n and summing over $n \geq 1$ yields

$$P_{ii}(s) - 1 = F_{ii}(s)\, P_{ii}(s),$$

hence for $|s| < 1$,

$$P_{ii}(s) = \frac{1}{1 - F_{ii}(s)}.$$

If $f_{ii}^* = F_{ii}(1) = 1$, then $F_{ii}(s) \to 1$ as $s \uparrow 1$ and therefore $P_{ii}(s) \to \infty$, which implies $\sum_{n\geq 0} P^{(n)}[i,i] = \infty$ (by Abel's theorem on power series). Thus i is recurrent.

If $f_{ii}^* < 1$, then taking the limit $s \uparrow 1$ in the identity above gives

$$\sum_{n=0}^{\infty} P^{(n)}[i,i] = P_{ii}(1) = \frac{1}{1 - f_{ii}^*} < \infty,$$

so i is transient and the displayed formula holds. □

Corollary 1.14. *If J is a class and one state in it is recurrent (resp. transient), then all are.*

1.2 Invariant measures and limit theorems

Definition 1.15 (Invariant Measure)**.** *A measure μ on I is **invariant** if X_n has law μ for all n.*

Proposition 1.16. *A measure μ is invariant if and only if*

$$\mu_i = \sum_{j\in I} \mu_j p_{ji}.$$

Proof. Let μ be a probability measure on I and let ν be the law of X_1 when X_0 has law μ. Then for each state i,

$$\nu_i = \mathbb{P}(X_1 = i) = \sum_{j\in I} \mathbb{P}(X_0 = j)\, p_{ji} = \sum_{j\in I} \mu_j p_{ji}.$$

If μ is invariant then $\nu = \mu$ and the displayed equality holds. Conversely, if the displayed equality holds for all i, then by induction one checks that the law of X_n is μ for every n, so μ is invariant. □

Theorem 1.17 (Existence of Invariant Measure)**.** *If I is finite, at least one invariant measure exists.*

Proof. Assume, by contradiction, that every state is transient. By Corollary 1.14, for every i, j the series $\sum_{n\geq 0} P^{(n)}[i,j]$ converges, hence $\lim_{n\to\infty} P^{(n)}[i,j] = 0$ for all i, j. But for each fixed n the matrix $P^{(n)}$ is stochastic, so

$$\sum_{j\in I} P^{(n)}[i,j] = 1 \quad \text{for every } i, n.$$

Passing to the limit as $n \to \infty$ and exchanging the finite sum with the limit yields $0 = 1$, a contradiction. Therefore at least one recurrent state exists. □

Theorem 1.18 (Renewal Limit Theorem)**.** *If i is a recurrent aperiodic state, then*

$$\lim_{n\to\infty} P^{(n)}[i,i] = \frac{1}{\mathbb{E}[T_i \mid X_0 = i]}.$$

Proof. By Theorem 1.12 for $i = j$,

$$P^{(n)}[i,i] = \sum_{k=1}^{n} f_{ii}^{(k)}\, P^{(n-k)}[i,i].$$

Set $a_k = f_{ii}^{(k)}$ and $u_n = P^{(n)}[i,i]$ (with $u_0 = 1$). Since i is recurrent we have $\sum_{k\geq 1} a_k = 1$. Aperiodicity of i means the greatest common divisor of $\{k \geq 1 : a_k > 0\}$ equals 1. The discrete renewal theorem for such a probability distribution (a_k) yields

$$\lim_{n\to\infty} u_n = \frac{1}{\sum_{k\geq 1} k a_k}.$$

But $\sum_{k\geq 1} k a_k = \mathbb{E}[T_i \mid X_0 = i]$, hence

$$\lim_{n\to\infty} P^{(n)}[i,i] = \frac{1}{\mathbb{E}[T_i \mid X_0 = i]}.$$

□

Definition 1.19 (Positive and Null Recurrence)**.** *A recurrent state is **positive** if* $\mathbb{E}[T_i \mid X_0 = i] < \infty$*, and **null** otherwise.*

Theorem 1.20 (Convergence to Invariant Distribution)**.** *Let the chain be irreducible and aperiodic. Then the following are equivalent:*

1. *every state is positive recurrent;*
2. *there exists an invariant probability measure* π *such that for all* i, j*,*

$$\lim_{n\to\infty} P^{(n)}[i,j] = \pi_j.$$

In this case the invariant measure is unique and

$$\pi_j = \frac{1}{\mathbb{E}[T_j \mid X_0 = j]}.$$

Proof. (1) ⇒ (2). Assume every state is positive recurrent. For each j Theorem 1.18 gives the existence of the limit

$$\pi_j := \lim_{n\to\infty} P^{(n)}[j,j] = \frac{1}{\mathbb{E}[T_j \mid X_0 = j]},$$

and $\pi_j > 0$. Using the renewal decomposition

$$P^{(n)}[i,j] = \sum_{k=1}^{n} f_{ij}^{(k)}\, P^{(n-k)}[j,j],$$

and dominated convergence (since $0 \le P^{(n-k)}[j,j] \le 1$ and $\sum_k f_{ij}^{(k)} \le 1$), we obtain

$$\lim_{n\to\infty} P^{(n)}[i,j] = \sum_{k=1}^{\infty} f_{ij}^{(k)}\, \pi_j = \pi_j.$$

One checks that the vector $\pi = (\pi_j)_{j\in I}$ satisfies the invariance equations $\pi_j = \sum_i \pi_i p_{ij}$ and, after normalization if necessary, is a probability measure.

(2) ⇒ (1). Suppose there exists an invariant probability measure π and that for all i, j,

$$\lim_{n\to\infty} P^{(n)}[i,j] = \pi_j.$$

Fix j. From the renewal identity and the existence of the limit $\pi_j > 0$ one deduces (using standard renewal arguments) that $\sum_k k f_{jj}^{(k)} < \infty$, hence $\mathbb{E}[T_j \mid X_0 = j] < \infty$ and j is positive recurrent. By irreducibility this holds for every state.

Uniqueness of the invariant law: if μ is any invariant probability measure then $\mu_j = \sum_i \mu_i P^{(n)}[i,j]$ for all n; letting $n \to \infty$ yields $\mu_j = \pi_j$ for every j, so $\mu = \pi$. Finally, the identity $\pi_j = 1/\mathbb{E}[T_j \mid X_0 = j]$ follows from the renewal limit established above. □

The strong Markov property

Define the natural filtration $\mathcal{F}_n = \sigma(X_0, \dots, X_n)$.

A **filtration** $(\mathcal{F}_n)_{n\ge 0}$ is a sequence of σ-algebras, so that:

$$\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \cdots,$$

Definition 1.21 (Stopping Time)**.** *A random variable* $T : \Omega \to \mathbb{N} \cup \{+\infty\}$ *is a **stopping time** if* $\{T \le n\} \in \mathcal{F}_n$ *for all* n*.*

Theorem 1.22 (Strong Markov Property)**.** *If* T *is a stopping time and a finite aperiodic state, then the process* $\widehat{X_n} = X_{T+n}$ *is a Markov chain with the same transition matrix.*

Proof. First note that for each fixed n and state j the event $\{\widehat{X_n} = j\} = \{X_{T+n} = j\}$ is measurable because

$$\{X_{T+n} = j\} = \bigcup_{v \geq 0} (\{T = v\} \cap \{X_{v+n} = j\}),$$

and each $\{T = v\} \in \mathcal{F}_v \subset \mathcal{F}_{v+n}$. Thus $(\widehat{X_n})$ is an adapted process.

Fix $m \geq 0$ and states $i_0, \ldots, i_m$ and suppose $\mathbb{P}(Y_0 = i_0, \ldots, Y_m = i_m) > 0$. For brevity write $A = \bigcap_{k=0}^{m} \{Y_k = i_k\}$. We compute, for any state j,

$$\mathbb{P}(Y_{m+1} = j \mid A) = \frac{\mathbb{P}(X_{T+m+1} = j,\, A)}{\mathbb{P}(A)} = \frac{\sum_{v \geq 0} \mathbb{P}(X_{v+m+1} = j,\, T = v,\, X_{v+k} = i_k \ (0 \leq k \leq m))}{\mathbb{P}(A)}.$$

For each fixed v the Markov property at time v gives

$$\mathbb{P}(X_{v+m+1} = j \mid X_{v+m} = i_m,\, T = v, \ldots) = \mathbb{P}(X_{v+m+1} = j \mid X_{v+m} = i_m) = p_{i_m j}.$$

Hence each summand equals $p_{i_m j}$ times the corresponding probability without the event $\{X_{v+m+1} = j\}$, and summing over v yields

$$\mathbb{P}(X_{T+m+1} = j,\, A) = p_{i_m j}\, \mathbb{P}(A).$$

Dividing gives $\mathbb{P}(Y_{m+1} = j \mid A) = p_{i_m j}$, which depends only on i_m and not on the earlier values $i_0, \ldots, i_{m-1}$. This is exactly the Markov property for $(\widehat{X_n})$ with the same transition probabilities p_{ij}. □

Theorem 1.23 (Ergodic Theorem for Markov Chains)**.** *Let $(X_n)_{n \geq 0}$ be a time-homogeneous Markov chain with countable state space S and transition matrix $P = (p_{ij})$. Assume that the chain is irreducible, aperiodic, and positive recurrent. Let π be its unique invariant probability distribution.*

Then, for any function $f : S \to \mathbb{R}$ such that

$$\sum_{i \in S} |f(i)| \pi_i < \infty,$$

we have

$$\frac{1}{n} \sum_{k=0}^{n-1} f(X_k) \ \xrightarrow[n \to \infty]{a.s.} \ \sum_{i \in S} f(i)\, \pi_i.$$

In particular, for all $i, j \in S$,

$$\frac{N_n(i)}{n} \xrightarrow{a.s.} \pi_i, \qquad \frac{N_n(i,j)}{n} \xrightarrow{a.s.} \pi_i p_{ij}, \qquad \frac{N_n(i,j)}{N_n(i)} \xrightarrow{a.s.} p_{ij},$$

where

$$N_n(i) = \sum_{k=0}^{n-1} \mathbb{1}_{\{X_k = i\}}, \qquad N_n(i,j) = \sum_{k=0}^{n-1} \mathbb{1}_{\{X_k = i,\, X_{k+1} = j\}}.$$

Proof. Since the chain is irreducible, aperiodic and positive recurrent, Theorem 1.20 implies that there exists a unique invariant probability distribution π and that

$$P^{(n)}[i,j] \to \pi_j \qquad \text{for all } i, j.$$

It is therefore sufficient to prove the result assuming that the chain is started in stationarity, i.e. $X_0 \sim \pi$. Indeed, the almost sure limits do not depend on the initial distribution.

Under this assumption the process $(X_n)_{n \geq 0}$ is strictly stationary. Moreover, irreducibility and aperiodicity imply that the stationary chain is ergodic. Hence Birkhoff's ergodic theorem applies and for any integrable function f,

$$\frac{1}{n} \sum_{k=0}^{n-1} f(X_k) \ \xrightarrow{\text{a.s.}} \ \mathbb{E}_\pi[f(X_0)] = \sum_{i \in S} f(i) \pi_i.$$

Let $f(i) = \mathbf{1}_{\{i = r\}}$ for some fixed $r \in S$. Then

$$\frac{1}{n} \sum_{k=0}^{n-1} \mathbf{1}_{\{X_k = r\}} = \frac{N_n(r)}{n} \ \xrightarrow{\text{a.s.}} \ \pi_r.$$

Now consider the function $g : S \times S \to \mathbb{R}$ defined by

$$g(x, y) = \mathbf{1}_{\{x=i,\, y=j\}}.$$

Applying the ergodic theorem to the stationary process $(X_k, X_{k+1})_{k\geq 0}$, we obtain

$$\frac{1}{n}\sum_{k=0}^{n-1} \mathbf{1}_{\{X_k=i,\, X_{k+1}=j\}} = \frac{N_n(i,j)}{n} \xrightarrow{\text{a.s.}} \mathbb{P}_\pi(X_0 = i, X_1 = j).$$

By stationarity,

$$\mathbb{P}_\pi(X_0 = i, X_1 = j) = \pi_i p_{ij}.$$

Finally, since $\pi_i > 0$ for all i by irreducibility and positive recurrence, we have $N_n(i) \to \infty$ almost surely. Therefore,

$$\frac{N_n(i,j)}{N_n(i)} = \frac{N_n(i,j)/n}{N_n(i)/n} \xrightarrow{\text{a.s.}} \frac{\pi_i p_{ij}}{\pi_i} = p_{ij}.$$

□

1.3 State Space Lifting for Higher-Order Markov Chains

[Xu, 2025]

Higher-Order Markov Chains

Let I be a finite state space with cardinality $|I| = m$. Let $\{X_n\}_{n\in\mathbb{N}}$ be a discrete-time stochastic process taking values in I.

Definition 1.24 (Higher-Order Markov Chain)**.** *The process $\{X_n\}$ is called a k-th order Markov chain if for all $n \geq k$ and all $i_0, \ldots, i_{n+1} \in I$,*

$$\mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = \mathbb{P}(X_{n+1} = i_{n+1} \mid X_n = i_n, \ldots, X_{n-k+1} = i_{n-k+1}).$$

Thus, the conditional distribution of the future state depends only on the previous k states.

Construction of the Lifted State Space Define the augmented state space

$$\widehat{I} := I^k,$$

the set of ordered k-tuples of elements of I.

An element $\mathbf{i} \in \widehat{I}$ is written as

$$\mathbf{i} = (i_1, i_2, \ldots, i_k),$$

which represents the configuration

$$(X_n = i_1,\ X_{n-1} = i_2,\ \ldots,\ X_{n-k+1} = i_k).$$

Definition of the Lifted Process Define a new stochastic process $\{\widehat{X_n}\}_{n\geq k-1}$ by

$$\widehat{X_n} := (X_n, X_{n-1}, \ldots, X_{n-k+1}) \in \widehat{I}.$$

Markov Property of the Lifted Process

By construction, the process $\widehat{X}_n$ retains exactly the information required by the k-th order Markov property, suggesting that it should be Markovian on $\widehat{I}$. This is made precise in the following proposition.

Transition structure of the lifted chain Let $(X_n)_{n\geq 0}$ be a homogeneous Markov chain with state space I and transition probabilities

$$P(i \mid i_1, \ldots, i_k) = \mathbb{P}(X_{n+1} = i \mid X_n = i_1, \ldots, X_{n-k+1} = i_k).$$

We define the *lifted process* $(\widehat{X}_n)_{n\geq k-1}$ by

$$\widehat{X}_n := (X_n, X_{n-1}, \ldots, X_{n-k+1}) \in \widehat{I} := I^k.$$

Proposition 1.25 (State Space Lifting)**.** *The process $(\widehat{X}_n)_{n\geq k-1}$ is a homogeneous Markov chain with state space $\widehat{I}$.*

Proof. Let

$$\mathbf{i} = (i_1, \ldots, i_k), \qquad \mathbf{j} = (j_1, \ldots, j_k) \in \widehat{I},$$

and assume $\mathbb{P}(\widehat{X}_n = \mathbf{i}) > 0$. Then

$$\widehat{X}_{n+1} = (X_{n+1}, X_n, \ldots, X_{n-k+2}),$$

and therefore

$$\mathbb{P}(\widehat{X}_{n+1} = \mathbf{j} \mid \widehat{X}_n = \mathbf{i}) = \begin{cases} \mathbb{P}(X_{n+1} = j_1 \mid X_n = i_1, \ldots, X_{n-k+1} = i_k), \\ \qquad\qquad \text{if } j_\ell = i_{\ell-1},\ \ell = 2, \ldots, k, \\ 0, \quad \text{otherwise.} \end{cases}$$

By homogeneity of (X_n), the right-hand side does not depend on n, which proves that $(\widehat{X}_n)$ is a homogeneous Markov chain. □

The transition probabilities of the lifted chain are therefore given by

$$\widehat{p}_{\mathbf{ij}} = \mathbb{P}(\widehat{X}_{n+1} = \mathbf{j} \mid \widehat{X}_n = \mathbf{i}) = \begin{cases} P(j_1 \mid i_1, \ldots, i_k), & \text{if } \mathbf{j} = (j_1, i_1, \ldots, i_{k-1}), \\ 0, & \text{otherwise.} \end{cases}$$

The matrix $\widehat{P} = (\widehat{p}_{\mathbf{ij}})_{\mathbf{i},\mathbf{j}\in\widehat{I}}$ is the transition matrix of the lifted chain.

Transition Matrix The matrix $\widehat{P}$ defines a stochastic transition matrix of dimension $m^k \times m^k$. Each row contains at most m nonzero entries, reflecting the shift structure of the lifted chain.

Conclusion A k-th order Markov chain on I can always be represented as a first-order Markov chain on the enlarged state space I^k. This procedure, known as *state space lifting*, allows all classical results for first-order Markov chains to be applied to higher-order models without modification.

2 Supervised Learning

[Rosasco, 2023] The problem of supervised learning is finding an input/output relation $\hat{f}$ on the basis of a training set

$$S = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$$

of input/output pairs, the examples. Given a new input x_{new}, the value $\hat{f}(x_{\text{new}})$ should be a good estimate or prediction of the corresponding output y_{new}. When this happens, we say that $\hat{f}$ *generalizes*. Supervised learning is an instance of the learning-by-examples paradigm.

2.1 Statistical Learning Theory

Next, we formalize these ideas following the framework of Statistical Learning Theory. We first provide a succinct definition of the problem, and then discuss in detail the role of the different elements in the definition: the probability distribution, the loss function, the associated risk, and the space of hypotheses.

- **Learning problem.** Let $(X, \mathcal{A})$ be a measure space with σ-algebra $\mathcal{A}$, and let $Y \subseteq \mathbb{R}$ with the corresponding Borel σ-algebra $\mathcal{B}$. Let (X, Y) be a pair of random variables with values in $(X \times Y)$ and law P. Let $\ell : Y \times \mathbb{R} \to [0, \infty)$ be measurable. Let $M(X, \mathbb{R}) \subset \mathbb{R}^X$ be the space of measurable functions. Define the expected risk

  $$L(f) = \mathbb{E}[\ell(Y, f(X))], \qquad \forall f \in M(X, \mathbb{R}).$$

  The learning problem is

  $$\min_{f \in M(X,\mathbb{R})} L(f),$$

  when P is known only through an i.i.d. sample $(x_i, y_i)_{i=1}^n$ of n copies of (X, Y).

  Remark 2.1. *(Time-series) In many practical situations, however, the inputs $x_1, x_2, x_3, \ldots$ are not merely independent points but consecutive observations in time or space. For in stance, consider a time series xtt=1,2,... describing the state of a system at each discrete time t. Under the classical regression assumption, we still treat (xt,yt) as i.i.d. samples; however, this may be unrealistic in most cases. In the next chapters, we will revisit regression tasks without the i.i.d. requirement on the data, such as Markov samples, where the underlying distribution is not fixed.*

- **Input space.** X is called the input space. Example: $X \subseteq \mathbb{R}^d$ with its Borel σ-algebra. More generally, X may be a metric space with its Borel σ-algebra. Examples include binary strings $\{0, 1\}^d$ with Hamming distance $\sum_{j=1}^d \mathbf{1}_{x_j \neq \bar{x}_j}$, or the simplex

  $$\{x \in \mathbb{R}^d : \sum_{j=1}^d x^j = 1\}$$

  with associated probability metrics.

- **Output space: regression and classification.** Y is the output space. If $Y = \mathbb{R}$, the problem is *regression*, if $Y = \mathbb{R}^d, d \in \mathbb{N}$, the problem is *multivariate regression* If $Y = \{-1, 1\}$, the problem is *binary classification.*

- **Data distribution.** P is a probability measure on $(X \times Y, \mathcal{A} \times \mathcal{B})$.

- **Marginal distribution.** The marginal measure P_X on $(X, \mathcal{A})$ is

$$P_X(A) = \int_{A\times\mathbb{R}} dP(x,y), \qquad \forall A \in \mathcal{A}.$$

- **Conditional distribution.** For all $x \in X$, there exists a Borel probability measure $P(\cdot \mid x)$ on $(\mathbb{R}, \mathcal{B})$ such that for all $B \in \mathcal{B}$ the map $x \mapsto P(B \mid x)$ is measurable, and for all $A \in \mathcal{A}$, $B \in \mathcal{B}$:

$$P(A \times B) = \int_A P(B \mid x)\, dP_X(x).$$

  In binary classification, the conditional distribution reduces to the point mass function:

$$\{P(1 \mid x),\, P(-1 \mid x)\}.$$

- **Training set.** The sample $(x_i, y_i)_{i=1}^n$ is called the training set. The acronym *i.i.d.* stands for identically and independently distributed.

- **Loss functions.** The function ℓ is called a loss function. Often continuous and convex in its second argument.

- **Loss functions for regression.** In regression, losses have the form $\ell(y, y') = V(y - y')$, where $V : \mathbb{R} \to [0, \infty)$. Examples:

$$V(a) = a^2, \qquad V(a) = |a|.$$

- **Loss functions for classification.** In binary classification, losses are of the form $\ell(y, y') = V(yy')$. Examples:

$$V(a) = \mathbf{1}_{\{a<0\}}, \quad V(a) = (1-a)^2, \quad V(a) = \log(1 + e^{-a}), \quad V(a) = e^{-a}, \quad V(a) = |1-a|_+.$$

- **Expected risk.** The functional

$$L(f) = \mathbb{E}[\ell(Y, f(X))]$$

  is the expected risk.

- **Inner risk.** For almost all $x \in X$, define the inner risk

$$L_x(a) = \int \ell(y, a)\, dP(y \mid x), \qquad a \in \mathbb{R}.$$

  Then, for all $f \in M(X, \mathbb{R})$,

$$L(f) = \int L_x(f(x))\, dP_X(x).$$

- **Target functions.** Assume that a function $f_P \in M(X, \mathbb{R})$ exists such that

$$L(f_P) = \min_{f\in M(X,\mathbb{R})} L(f).$$

  Then f_P is called a *target function*.

- **Target functions via inner risk.** If for almost all x there exists a_x minimizing

$$L_x(a_x) = \min_{a\in\mathbb{R}} L_x(a),$$

  and we set $f_P(x) = a_x$, then f_P is a target function.

- **Regression function.** For the square loss, the target function is the regression function:

$$f_P(x) = \int y\, dP(y \mid x).$$

- **Bayes decision rule.** For the misclassification loss, the target function is the Bayes classifier:

$$f_P(x) = \arg\max_{y\in\{-1,1\}} P(y \mid x).$$

2.2 Empirical risk minimization with least squares

In this section* we study perhaps the most common approach to supervised statistical learn- ing namely empirical risk minimization. In particular, we focus on linear least squares. This simple model allows to explore some fundamental ideas to build intuitions and understanding that will naturally extend in later chapters. Some useful notions such as overparametrization, minimal norm solution, regularization and stability are introduced.

- **Empirical risk.** For a given loss function and training set $(x_1, y_1), \ldots, (x_n, y_n)$, consider the functional $\hat{L} : M(X, R) \to R$ defined for all $f \in M(X, R)$ as

$$\hat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)).$$

- **Hypotheses space.** Let $H \subset M(X, R)$. We call H a hypothesis space.

- **Empirical risk and ERM.** For a given loss function, training set $(x_1, y_1), \ldots, (x_n, y_n)$, and hypothesis space H consider the minimization problem

$$\min_{f \in H} \hat{L}(f).$$

  Note that in general the above minimization problem might not have a solution or have multiple ones.

- **Linear functions.** The space of linear function is

$$H_{\text{lin}} = \{f : \mathbb{R}^d \to \mathbb{R} \mid \exists w \text{ s.t. } f(x) = x^\top w, \ \forall x \in \mathbb{R}^d\}.$$

  Note that each function $f \in H_{\text{lin}}$ is uniquely identified by a corresponding vector $w \in \mathbb{R}^d$. Then, we can identify H_{lin} with $\mathbb{R}^d$. In particular we can define an inner product

$$\langle f, f' \rangle_{H_{\text{lin}}} = w^\top w',$$

  and a corresponding norm

$$\|f\|_{H_{\text{lin}}} = \|w\|,$$

  for $f, f' \in H_{\text{lin}}$, where $f(x) = x^\top w$ and $f'(x) = x^\top w'$.

- **Linear least squares.**

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (y_i - x_i^\top w)^2.$$

- **Interpolation.** A system of interpolating equations is given by

$$x_i^\top w = y_i, \quad i = 1, \ldots, n.$$

  An interpolating solution is $w \in \mathbb{R}^d$ satisfying the above equations.

- **Linear systems.** Let $\hat{X} \in \mathbb{R}^{n \times d}$ be the matrix with rows $x_1, \ldots, x_n$ and $\hat{y} \in \mathbb{R}^n$ the vector with entries $y_1, \ldots, y_n$. Consider the system

$$\hat{X} w = \hat{y}.$$

- **Underdetermined systems and overparameterized models.** A system of linear equations is underdetermined if $n < d$. In machine learning this corresponds to overparameterized models.

- **Overdetermined systems and underparameterized models.** A system is overdetermined if $n > d$. This corresponds to underparameterized models.

- **Linear least squares solution.** Assume $n > d$ and columns of $\hat{X}$ are linearly independent. Then the least squares solution is

$$\hat{w}^\dagger = (\hat{X}^\top \hat{X})^{-1} \hat{X}^\top \hat{y}.$$

- **Minimal norm solution.** Assume $n < d$ and rows of $\hat{X}$ are linearly independent. Then the minimal norm solution is the solution of

$$\min_{w \in \mathbb{R}^d} \|w\| \quad \text{s.t.} \quad \hat{X}w = \hat{y}.$$

  Using Lagrange multipliers:

$$\hat{w}^\dagger = \hat{X}^\top (\hat{X}\hat{X}^\top)^{-1} \hat{y}.$$

- **Penalized empirical risk.** For $\lambda > 0$:

$$\min_{f \in H_{\text{lin}}} \hat{L}(f) + \lambda \|f\|^2_{H_{\text{lin}}}.$$

- **Ridge regression aka Tikhonov regularization.**

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (y_i - x_i^\top w)^2 + \lambda \|w\|^2.$$

  The unique solution is

$$\hat{w}_\lambda = (\hat{X}^\top \hat{X} + n\lambda I)^{-1} \hat{X}^\top \hat{y},$$

  and equivalently,

$$\hat{w}_\lambda = \hat{X}^\top (\hat{X}\hat{X}^\top + n\lambda I)^{-1} \hat{y}.$$

2.3 Reproducing kernel Hilbert spaces

The non-linear ridge regression estimator associated with the map Φ defined by only depends on the kernel

$$K_\Phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \quad K_\Phi(x, x') = \Phi(x)^T \Phi(x'). \tag{2.1}$$

In this lecture we show that any kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defines a corresponding hypothesis space $\mathcal{H}$, provided that it satisfies the following condition.

Definition 2.2 (Semi-positive Definite Kernel)**.** *Let $\mathcal{X}$ be a set. A map $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that*

$$K(x, x') = K(x', x) \quad x, x' \in \mathcal{X} \tag{2.2}$$

$$\sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) \geq 0 \quad x_1, \ldots, x_n \in \mathcal{X}, c_1, \ldots, c_n \in \mathbb{R} \tag{2.3}$$

is called a semi-positive definite kernel.

Condition (2.2) and (2.3) are equivalent to the fact that the $n \times n$-matrix

$$\hat{K} = [K(x_i, x_j)]_{i,j=1,\ldots,n} \tag{2.4}$$

is symmetric and semi-positive definite, i.e. $c^T \hat{K} c \geq 0$ for all $c \in \mathbb{R}^n$. It is easy to check that K_Φ defined by (2.1) is semi-positive definite.

The following result allows to define a Hilbert space of functions associated with K.

Theorem 2.3 (Kolmogorov-Moore Theorem)**.** *Take a semi-positive definite kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, then there exists a unique space $\mathcal{H}$ with the following properties*

a) the elements of $\mathcal{H}$ are functions $f : \mathcal{X} \to \mathbb{R}$

b) $\mathcal{H}$ is a vector space with respect to the usual pointwise operations of sum and product by scalar

c) $\mathcal{H}$ is a Hilbert space with scalar product $\langle \cdot, \cdot \rangle_\mathcal{H}$ and norm $\| \cdot \|_\mathcal{H}$

d) for all $x \in \mathcal{X}$ there is $K_x \in \mathcal{H}$ such that

$$f(x) = \langle f, K_x \rangle_\mathcal{H} \quad f \in \mathcal{H} \tag{2.5}$$

such that

$$K(x, x') = \langle K_x, K_{x'} \rangle_\mathcal{H} \quad x, x' \in \mathcal{X}. \tag{2.6}$$

Conversely, take a space $\mathcal{H}$ satisfying a)–d) and define K by means of (2.6), then K is a semi-positive definite kernel.

Proof. We assume that K is a semi-positive definite kernel and we prove the existence of $\mathcal{H}$. We split the proof into three steps.

Step 1. Per all $x \in \mathcal{X}$, set

$$K_x : \mathcal{X} \to \mathbb{R} \quad K_x(x') = K(x', x)$$

and

$$W_0 = \left\{ \sum_{i=1}^{n} c_i K_{x_i} \mid n \in \mathbb{N}, x_1, \ldots, x_n \in \mathcal{X}, c_1, \ldots, c_n \in \mathbb{R} \right\}.$$

Clearly W_0 is a vector space of functions from $\mathcal{X}$ to $\mathbb{R}$, with respect to the usual pointwise sum and product by scalar.

Step 2. Given $f, f' \in W_0$ set

$$\langle f, f' \rangle_{W_0} = \sum_{i=1}^{n} \sum_{j=1}^{m} c_i c'_j K(x'_j, x_i)$$

where $f = \sum_{i=1}^{n} c_i K_{x_i}$ and $f' = \sum_{i=1}^{m} c'_i K_{x'_i}$. A simple computation shows that

$$\langle f, f' \rangle_{W_0} = \sum_{j=1}^{m} c'_j f(x'_j) = \sum_{i=1}^{n} c_i f'(x_j),$$

so that the definition of the scalar product is independent of the representation of f and f'. It follows that $\langle f, f' \rangle_{W_0}$ is a bilinear function of f and f'. Furthermore, for all $x \in \mathcal{X}$

$$\langle f, K_x \rangle_{W_0} = f(x).$$

The assumption that K is semi-positive definite implies that

$$\langle f, f' \rangle_{W_0} = \langle f', f \rangle_{W_0} \quad \langle f, f \rangle_{W_0} \geq 0.$$

Finally, we claim that, if $\langle f, f \rangle_{W_0} = 0$, then $f = 0$. Indeed, for all $x \in \mathcal{X}$, by Cauchy-Schwarz inequality

$$|f(x)| = |\langle f, K_x \rangle_{W_0}| \leq \sqrt{\langle f, f \rangle_{W_0} \langle K_x, K_x \rangle_{W_0}} = 0,$$

so that $f(x) = 0$, cioé $f = 0$. Hence W_0 is a vector space of functions with a scalar product satisfying (2.5).

Step 3. Let W be the completion of W_0, i.e. W is a Hilbert space such that W_0 is a dense subspace of W. Set

$$\Phi : \mathcal{X} \to \mathcal{H} \quad \Phi(x) = K_x.$$

Since W_0 is dense W

$$\mathcal{W}_\Phi = \text{span}\{\Phi(x) \mid x \in \mathcal{X}\} = W$$

and, by Theorem 2.9

$$\mathcal{H} = \{f : \mathcal{X} \to \mathbb{R} \mid f(x) = \langle w, K_x \rangle_W, w \in W\}$$

is the unique reproducing kernel Hilbert space with kernel

$$\langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}} = \langle K_x, K_{x'} \rangle_{W_0} = K(x, x').$$

Conversely, take a reproducing kernel Hilbert space with reproducing kernel K, then K is clearly symmetric and

$$\sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) = \sum_{i,j=1}^{n} c_i c_j \langle K_{x_i}, K_{x_j} \rangle_{\mathcal{H}} = \left\| \sum_{i=1}^{n} c_i K_{x_i} \right\|_{\mathcal{H}}^2 \geq 0$$

so that K is semi-positive definite.

□

The above theorem motivates the following definition.

Definition 2.4 (Reproducing Kernel Hilbert Space)**.** *Take a set $\mathcal{X}$. A space $\mathcal{H}$ satisfying*

a) the elements of $\mathcal{H}$ are functions $f : \mathcal{X} \to \mathbb{R}$

b) $\mathcal{H}$ is a vector space with respect to the usual pointwise operations of sum and product by scalar

c) $\mathcal{H}$ is a Hilbert space with scalar product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and norm $\| \cdot \|_{\mathcal{H}}$

d) for all $x \in \mathcal{X}$ there is $K_x \in \mathcal{H}$ such that

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}} \quad f \in \mathcal{H} \tag{2.7}$$

is called a Reproducing Kernel Hilbert spaces (RKHS) on $\mathcal{X}$ with reproducing kernel K

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \quad K(x, x') = \langle K_x, K_{x'} \rangle_{\mathcal{H}}.$$

Eq. (2.7) is called the reproducing property. The above theorem shows that there is a one-to-one correspondence between RKHS and semi-positive definite kernels.

The reproducing property (2.7) implies that the norm of f bounds the value of f in each point $x \in \mathcal{X}$, i.e.

$$|f(x)| \leq C_x \|f\|_{\mathcal{H}}, \tag{2.8}$$

where the best constant is $C_x = \|K_x\|_{\mathcal{H}} = \sqrt{K(x, x)}$. By Riesz-Markov lemma, Eq. (2.8) is equivalent to the reproducing property.

The following proposition states the main properties of a RKHS.

Proposition 2.5 (Properties of RKHS)**.** *Let $\mathcal{H}$ be a reproducing kernel Hilbert space with reproducing kernel K.*

a) For all $x \in \mathcal{X}$, there is a unique $K_x \in \mathcal{H}$ such that the reproducing property holds true,

$$K_x(x') = K(x, x') \quad x' \in \mathcal{X} \implies K_x = K(\cdot, x),$$

the reproducing kernel is unique, and

$$span\{K_x \mid x \in \mathcal{X}\} = \mathcal{H} \tag{2.9}$$

b) if $(f_n)_n$ is a sequence converging to f in $\mathcal{H}$, then

$$\lim_{n \to +\infty} f_n(x) = f(x) \quad \forall x \in \mathcal{X} \tag{2.10}$$

c) let $(\phi_\ell)_{\ell \in \Lambda}$ be a base of $\mathcal{H}$, then

$$K(x, x') = \sum_{\ell \in \Lambda} \phi_\ell(x) \phi_\ell(x') \quad x, x' \in \mathcal{X}$$

where the series convergences absolutely.

d) if $\mathcal{H}'$ is another reproducing kernel Hilbert space with reproducing kernel K, then $\mathcal{H}' = \mathcal{H}$ as Hilbert spaces.

Proof. To prove item a), assume that for some $x \in \mathcal{X}$, there is another $K'_x \in \mathcal{H}$ such that the reproducing property (2.7) holds true, then

$$\langle K'_x - K_x, f \rangle_{\mathcal{H}} = \langle K'_x, f \rangle_{\mathcal{H}} - \langle K_x, f \rangle_{\mathcal{H}} = f(x) - f(x) = 0 \quad \forall f \in \mathcal{H},$$

so that $K'_x - K_x = 0$. Furthermore, the reproducing property applied to $f = K_x$ gives

$$K_x(x') = \langle K_x, K_{x'} \rangle_{\mathcal{H}} = K(x, x'),$$

where the last equality is a consequence of the definition of reproducing kernel. Hence K is uniquely defined, too. Take now $f \in \{K_x\}^\perp$, then by the reproducing property

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}} = 0 \quad \forall x \in \mathcal{X}$$

so that $f = 0$, which provides (2.9).

We prove (2.10). Take $(f_n)_n$ be a sequence converging to f in $\mathcal{H}$ and $x \in \mathcal{X}$. By the reproducing property and the continuity of the scalar product

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}} = \lim_{n\to\infty} \langle f_n, K_x \rangle_{\mathcal{H}} = \lim_{n\to\infty} f_n(x).$$

To show item c), let $(\phi_\ell)_{\ell\in\Lambda}$ be a base of $\mathcal{H}$. Then

$$K(x, x') = \langle K_x, K_{x'} \rangle_{\mathcal{H}} = \sum_{\ell\in\Lambda} \langle K_x, \phi_\ell \rangle_{\mathcal{H}} \langle \phi_\ell, K_{x'} \rangle_{\mathcal{H}} = \sum_{\ell\in\Lambda} \phi_\ell(x)\phi_\ell(x'),$$

where the last inequality is a consequence of the reproducing property. Furthermore, Cauchy-Schwarz inequality gives that

$$\left(\sum_{\ell\in\Lambda} |\phi_\ell(x)\phi_\ell(x')| \right)^2 \leq \sum_{\ell\in\Lambda} \phi_\ell(x)^2 \sum_{\ell\in\Lambda} \phi_\ell(x')^2 = K(x, x)K(x', x') < \infty,$$

so that the series converges absolutely.

About the claim in item d), take another reproducing kernel Hilbert space $\mathcal{H}_1$ with the same reproducing kernel K. Given $x \in \mathcal{X}$, the reproducing property for $\mathcal{H}_1$ ensures there exists $K_x^1 \in \mathcal{H}_1$ such that

$$f_1(x) = \langle f, K_{x,\rangle}^1 \, \mathcal{H}_1.$$

However, since the reproducing kernel is K, by item a) $K_x^1 = K(\cdot, x) = K_x$, so that $K_x \in \mathcal{H}_1$ and, by linearity, the canonical inclusion

$$j : \operatorname{span}\{K_x \mid x \in \mathcal{X}\} \subseteq \mathcal{H} \to \mathcal{H}_1 \quad j(f) = f$$

is well-defined. We claim that j is an isometry. Indeed, take

$$f = \sum_{i=1}^n c_i K_{x_i} \in \operatorname{span}\{K_x \mid x \in \mathcal{X}\} \subseteq \mathcal{H}$$

then

$$\left\| j\left(\sum_{i=1}^n c_i K_{x_i} \right) \right\|_{\mathcal{H}_1}^2 = \sum_{i,j=1}^n c_i c_j K(x_i, x_j) = \left\| \sum_{i=1}^n c_i K_{x_i} \right\|_{\mathcal{H}}^2.$$

Since $\operatorname{span}\{K_x \mid x \in \mathcal{X}\}$ is dense in $\mathcal{H}$ by (2.9), then j extends to an isometry from $\mathcal{H}$ into $\mathcal{H}_1$. Note that, since $j(K_x) = K_x$

$$(jf)(x) = \langle j(f), j(K_x) \rangle_{\mathcal{H}_1} = \langle f, K_x \rangle_{\mathcal{H}} = f(x) \quad f \in \mathcal{H},$$

so that j is the identity and $\mathcal{H}_1 = \mathcal{H}$ as Hilbert spaces. □

Example 2.6 (Linear Kernel). *Let $\mathcal{X} = \mathbb{R}^d$, the linear kernel is*

$$K(x, x') = x^T x', \tag{2.11}$$

which is semi-positive definite since it is symmetric and

$$\sum_{i,j=1}^{n} \alpha_i c_j K(x_i, x_j) = \sum_{i,j=1}^{n} \alpha_i c_j x_i^T x_j = \left(\sum_{i=1}^{n} \alpha_i x_i\right)^T \left(\sum_{j=1}^{n} c_j x_j\right) = \left\|\sum_{i=1}^{n} \alpha_i x_i\right\|_d^2 \geq 0.$$

By (2.9)*, the corresponding Hilbert space is*

$$\begin{aligned}\mathcal{H} &= span\{K_w : \mathcal{X} \to \mathbb{R} \mid K_w(x) = w^T x, w \in \mathbb{R}^d\} \\ &= \{K_w : \mathcal{X} \to \mathbb{R} \mid K_w(x) = w^T x, w \in \mathbb{R}^d\},\end{aligned}$$

which is the space of linear functions with the scalar product

$$\langle K_w, K_{w'} \rangle_{\mathcal{H}} = K(w, w') = w^T w'.$$

Example 2.7 (Gaussian Kernel). *Let $\mathcal{X} = \mathbb{R}^d$. Given $\sigma > 0$, the Gaussian kernel of width σ is*

$$K(x, x') = \exp\left(-\frac{\|x - x'\|_d^2}{\sigma^2}\right) \quad x, x' \in \mathbb{R}^d. \tag{2.12}$$

We prove that it is semi-positive definite kernel. Indeed, it is clearly symmetric and, to prove (2.3)*, recall that*

$$\exp(-\pi \|k\|_d^2) = \int_{\mathbb{R}^d} \exp(-\pi \|\xi\|_d^2) \exp(2\pi i \xi^T k) d\xi, \tag{2.13}$$

then

$$\begin{aligned}\sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) &= \int_{\mathbb{R}^d} \exp(-\pi \|\xi\|_d^2) \sum_{i,j=1}^{n} c_i c_j \exp\left(2\pi i \xi^T \frac{(x_i - x_j)}{\sqrt{\pi}\sigma}\right) d\xi \\ &= \left(\frac{1}{2\pi}\right)^d \int_{\mathbb{R}^d} \exp(-\pi \|\xi\|_d^2) \left|\sum_{i=1}^{n} c_i \exp\left(2\pi i \xi^T \frac{x_i}{\sqrt{2}\sigma}\right)\right|^2 d\xi \geq 0\end{aligned}$$

Note above proof shows that all kernels of the form

$$K(x, x') = C \int_{\mathbb{R}^d} \exp(2\pi i \xi^T (x - x')) d\mu(\xi),$$

where μ is a probability measure, are semi-positive definite.

Next proposition shows that, if K is continuous, the elements of $\mathcal{H}$ are continuous functions and they are square-integrable with respect to a suitable class of probability distributions on $\mathcal{X}$.

Proposition 2.8 (Continuity and Integrability in RKHS). *Set $\mathcal{X} = \mathbb{R}^d$ and let $\mathcal{H}$ be a reproducing kernel Hilbert space with a continuous reproducing kernel K.*

a) $\mathcal{H}$ is separable and the elements of $\mathcal{H}$ are continuous functions.

b) Given a probability measure $\rho_{\mathcal{X}}$ on $\mathcal{X}$, assume there is a subset $\mathcal{X}_0 \subseteq \mathcal{X}$ such that

$$\rho_{\mathcal{X}}(\mathcal{X}_0) = 1 \quad \sup_{x \in \mathcal{X}_0} K(x, x) = \kappa^2 < +\infty,$$

then the elements of $\mathcal{H}$ are square-integrable with respect to $\rho_{\mathcal{X}}$ and

$$\|f\|_{L^2} \leq \kappa \|f\|_{\mathcal{H}} \quad f \in \mathcal{H}.$$

Proof. Assume that K is continuous and take $f \in \mathcal{H}$. We prove that f is continuous. Fix $x, x_0 \in \mathcal{X}$, the reproducing property and Cauchy-Schwarz inequality give

$$|f(x) - f(x_0)| \leq \|f\|_{\mathcal{H}} \|K_x - K_{x_0}\|_{\mathcal{H}} = \|f\|_{\mathcal{H}} \sqrt{K(x,x) + K(x_0,x_0) - 2K(x,x_0)}.$$

Since K is continuous, then $\lim_{x \to x_0} f(x) = f(x_0)$. To show that $\mathcal{H}$ is separable it is enough to show the existence of a countable family $\{f_i\}_{i \in \mathbb{N}}$ such that $(\{f_i\}_{i \in \mathbb{N}})^{\perp} = \{0\}$. Since $\mathcal{X}$ is separable, then there exists a dense countable family $\{x_i\}_{i \in \mathbb{N}}$ dense in $\mathcal{X}$. Define $f_i = K_{x_i}$ and let $f \in \mathcal{H}$ such that $\langle f, K_{x_i} \rangle_{\mathcal{H}} = 0$ for all $i \in \mathbb{N}$. The reproducing property implies that $f(x_i) = 0$ per ogni $i \in \mathbb{N}$. Since f is continuous, then $f(x) = 0$ per ogni $x \in \mathcal{X}$, hence $f = 0$. Take $\rho_{\mathcal{X}}$ and $\mathcal{X}_0$ as in item b). Let $f \in \mathcal{H}$, since f is continuous, it is measurable and

$$\int_{\mathcal{X}} |f(x)|^2 d\rho_{\mathcal{X}}(x) = \int_{\mathcal{X}_0} |f(x)|^2 d\rho_{\mathcal{X}}(x) = \int_{\mathcal{X}_0} |\langle f, K_x \rangle_{\mathcal{H}}|^2 d\rho_{\mathcal{X}}(x)$$

$$\leq \int_{\mathcal{X}_0} \|f\|_{\mathcal{H}}^2 \|K_x\|_{\mathcal{H}}^2 d\rho_{\mathcal{X}}(x) \leq \|f\|_{\mathcal{H}}^2 \sup_{x \in \mathcal{X}_0} K(x,x) \rho_{\mathcal{X}}(\mathcal{X}_0) = \kappa^2 \|f\|_{\mathcal{H}}^2,$$

so that $f \in L^2(\mathcal{X}, \rho_{\mathcal{X}})$ and $\|f\|_{L^2} \leq \kappa \|f\|_{\mathcal{H}}$. □

The above result holds true for any separable metric space $\mathcal{X}$.

Next result provides a powerfull technique to define RKHS.

Theorem 2.9 (Feature Map)**.** *Let $\mathcal{W}$ be a Hilbert space with scalar product $\langle \cdot, \cdot \rangle_{\mathcal{W}}$ and a map*

$$\Phi : \mathcal{X} \to \mathcal{W} \quad x \mapsto \Phi(x),$$

called the feature map. For all $w \in \mathcal{W}$ set

$$f_w : \mathcal{X} \to \mathbb{R} \quad f_w(x) = \langle w, \Phi(x) \rangle_{\mathcal{W}}$$

and denote

$$\mathcal{W}_\Phi = span\{\Phi(x) \mid x \in \mathcal{X}\}$$
$$\mathcal{H} = \{f_w \mid w \in \mathcal{W}_\Phi\}.$$

Then

a) $\mathcal{H}$ is a reproducing kernel Hilbert space with respect to the scalar product

$$\langle f_w, f_{w'} \rangle_{\mathcal{H}} = \langle w, w' \rangle_{\mathcal{W}} \tag{2.14}$$

b) $\mathcal{H}$ is the unique reproducing kernel Hilbert space with reproducing kernel

$$K(x,x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{W}} \tag{2.15}$$

and

$$\mathcal{H} = \{f_w \mid w \in \mathcal{W}\} \tag{2.16}$$

c) the map

$$\Phi^* : \mathcal{W} \to \mathcal{H} \quad \Phi^*(w) = f_w \tag{2.17}$$

is a partial isometry with $\ker \Phi^* = \mathcal{W}_\Phi^{\perp}$ and Im $\Phi^* = \mathcal{H}$.

d) Take a base $(e_\ell)_{\ell \in \Lambda}$ of $\mathcal{W}$ and define $\phi_\ell = f_{e_\ell}$ then

$$K(x,x') = \sum_{\ell \in \Lambda} \phi_\ell(x) \phi_\ell(x'), \tag{2.18}$$

where the series converges absolutely. If $\mathcal{W}_\Phi = \mathcal{W}$, then $(\phi_\ell)_{\ell \in \Lambda}$ is a base of $\mathcal{H}$.

The property that $\mathcal{W}_\Phi = \mathcal{W}$ is equivalent to the condition

$$f_w = 0 \iff w = 0. \tag{2.19}$$

If $\mathcal{W}_\Phi \subsetneq \mathcal{W}$, $(\phi_\ell)_{\ell\in\mathbb{N}}$ is not a base of $\mathcal{H}$, nevertheless

$$\|f\|_{\mathcal{H}}^2 = \sum_{\ell=1}^{+\infty} \langle f, \phi_\ell\rangle_{\mathcal{H}}^2 \quad f \in \mathcal{H}.$$

Proof. **Step 1.** We first show that $\mathcal{H}$ is a RKHS. Denote by

$$\mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$$

the vector space of all functions from $\mathcal{X}$ to $\mathbb{R}$ with the usual pointwise operations of sum and of product by scalar. We first show that the map

$$\Phi^* : \mathcal{W} \to \mathbb{R}^{\mathcal{X}} \quad \Phi^* w = f_w$$

is linear. Indeed, for all $w_1, w_2 \in \mathcal{W}$ and $\lambda_1, \lambda_2 \in \mathbb{R}$

$$\begin{aligned} f_{\lambda_1 w_1 + \lambda_2 w_2}(x) &= \langle \lambda_1 w_1 + \lambda_2 w_2, \Phi(x)\rangle_{\mathcal{W}} \\ &= \lambda_1 \langle w_1, \Phi(x)\rangle_{\mathcal{W}} + \lambda_2 \langle w_2, \Phi(x)\rangle_{\mathcal{W}} \\ &= \lambda_1 f_{w_1}(x) + \lambda_2 f_{w_2}(x) \quad x \in \mathcal{X}. \end{aligned}$$

Since Φ^* is linear and $\mathcal{W}_\Phi$ is a vector space, so is $\mathcal{H} = \Phi^*(\mathcal{W}_\Phi)$.

Step 2. We show that the restriction of Φ^* to $\mathcal{W}_\Phi$ is injective. Take $w \in \mathcal{W}$ such that $\Phi^*(w) = 0$, i.e.

$$f_w(x) = \langle w, \Phi(x)\rangle_{\mathcal{W}} = 0 \quad \forall x \in \mathcal{X},$$

then $w \in \{\Phi(x) \mid x \in \mathcal{X}\}^\perp = \mathcal{W}_\Phi^\perp$, so that $\ker \Phi^* \subseteq \mathcal{W}_\Phi^\perp$. Furthermore, by construction $\mathcal{W}_\Phi^\perp \subseteq \ker \Phi^*$, so that

$$\ker \Phi^* = \mathcal{W}_\Phi^\perp. \tag{2.20}$$

As a consequence, Φ^* is a linear isomorphism from $\mathcal{W}_\Phi$ onto $\mathcal{H}$. Since $\mathcal{W}_\Phi$ is a closed subspace of a Hilbert space, $\mathcal{W}_\Phi$ is a Hilbert space, too. Hence, $\mathcal{H}$ becomes a Hilbert space by setting

$$\langle f, f'\rangle_{\mathcal{H}} = \langle \Phi^{*-1}(f), \Phi^{*-1}(f')\rangle_{\mathcal{W}_\Phi},$$

which is equivalent to (2.14) and it implies that Φ^* is a partial isometry with $\ker \Phi^* = \mathcal{W}_\Phi^\perp$.

Step 3. We now prove the reproducing property. Let $x \in \mathcal{X}$ and $f \in \mathcal{H}$, then $f = f_w$ for some $w \in \mathcal{W}_\Phi$ and

$$f(x) = \langle w, \Phi(x)\rangle_{\mathcal{W}} = \langle f_w, f_{\Phi(x)}\rangle_{\mathcal{H}} = \langle f, K_x\rangle_{\mathcal{H}}$$

where $K_x = f_{\Phi(x)}$, i.e.

$$K_x(x') = f_{\Phi(x)}(x') = \langle \Phi(x), \Phi(x')\rangle_{\mathcal{W}} = K(x, x'),$$

which is (2.15). Hence $\mathcal{H}$ is a reproducing kernel Hilbert space with reproducing kernel K and it is unique by item d) of Proposition 2.5.

Step 4. The fact that Φ^* is a partial isometry from $\mathcal{W}$ onto $\mathcal{H}$ is a direct consequence of (2.20) and (2.14).

Step 5. The proof of (2.18) is similar to the proof of item c) of Proposition 2.5. Furthermore, if $\mathcal{W}_\Phi = \mathcal{W}$, then Φ^* is a unitary map, so that the last claim is clear.

□

Example 2.10 (Affine Functions)**.** *Given $d \geq 2$, let $\mathcal{X} = \mathbb{R}^{d-1}$ and $\mathcal{W} = \mathbb{R}^d$ with the Euclidean scalar product. Set $\Phi(x) = (x, 1)$ and write $w = (\tilde{w}, b)$, then*

$$f_{(\tilde{w},b)}(x) = x^T\tilde{w} + b$$

so that $\mathcal{H}$ is the space of affine functions, which is a reproducing kernel Hilbert space kernel

$$K(x, x') = (x^T x' + 1).$$

It is easy to check that (2.19) *holds true, so that the scalar product is*

$$\langle f_{(\tilde{w},b)}, f_{(\tilde{w}',b')}\rangle_{\mathcal{H}} = \tilde{w}^T\tilde{w}' + bb'$$

Example 2.11 (Finite RKHS)**.** *Take a family of p linearly independent functions*

$$\phi_1, \ldots, \phi_p : \mathcal{X} \to \mathbb{R}$$

and define the feature map

$$\Phi : \mathcal{X} \to \mathbb{R}^p \quad \Phi(x) = \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_p(x) \end{bmatrix}. \tag{2.21}$$

Since the functions are linearly independent, (2.19) *holds true so that $\mathbb{R}^p = (\mathbb{R}^p)_\Phi$ and, as a consequence, the reproducing kernel Hilbert space associated with Φ is*

$$\mathcal{H} = \{f_w : \mathcal{X} \to \mathbb{R} \mid f_w(x) = w^T\Phi(x) \text{ where } w \in \mathbb{R}^p\} = span\{\phi_1, \ldots, \phi_p\}$$

with the scalar product

$$\langle f_w, f_{w'}\rangle_{\mathcal{H}} = w^T w' \quad \|f_w\|^2_{\mathcal{H}} = w^T w.$$

Conversely, take a finite dimensional reproducing kernel Hilbert space $\mathcal{H}$ with reproducing kernel K, fix a base $(\phi_\ell)_{\ell=1}^p$ of $\mathcal{H}$ where $p = \dim\mathcal{H}$ and define Φ by (2.21)*. As a consequence of item c) of Proposition 2.5, $K(x, x') = \Phi(x)^T\Phi(x')$, so that the reproducing kernel Hilbert space associated with Φ is precisely $\mathcal{H}$. Hence, we can identify $\mathcal{H}$ and $\mathbb{R}^p$ by means of the unitary map Φ^**

$$\Phi^* : \mathbb{R}^p \to \mathcal{H} \quad \Phi^*(w) = f_w.$$

For example, with the choice of the feature map

$$\Phi : \mathcal{X} \to \mathbb{R}^d \quad \Phi(x) = x \implies K(x, x') = x^T x'$$

the reproducing kernel Hilbert space of linear functions of Example 2.6 is identified with $\mathbb{R}^d$.

Example 2.12 (Gaussian Kernel RKHS)**.** *Let $\mathcal{W} = \ell^2$ be the Hilbert space of square-summable sequences with the scalar product*

$$\langle (a_\ell)_\ell, (b_n)_\ell\rangle_{\ell^2} = \sum_{\ell=0}^{+\infty} a_\ell b_\ell.$$

Define

$$\Phi : \mathbb{R} \to \ell^2 \quad \Phi(x)_\ell = e^{-\frac{x^2}{2}}\frac{x^\ell}{\sqrt{\ell!}} = \phi_\ell(x) \quad \ell \in \mathbb{N},$$

which is well-defined since

$$\sum_{\ell=0}^{+\infty}\phi_\ell(x)^2 = e^{-x^2}\sum_{\ell=0}^{+\infty}\frac{x^{2\ell}}{\ell!} = e^{-x^2}e^{x^2} = 1.$$

Hence

$$\mathcal{H} = \{f : \mathbb{R} \to \mathbb{R} \mid f(x) = \sum_{\ell=1}^{+\infty} a_\ell\phi_\ell(x), \sum_{\ell=1}^{+\infty} a_\ell^2 < +\infty\}$$

is a reproducing kernel Hilbert space with kernel

$$K(x, x') = e^{-\frac{x^2}{2}-\frac{x'^2}{2}}\sum_{\ell=0}^{+\infty}\frac{(xx')^\ell}{\ell!} = e^{-\frac{x^2}{2}-\frac{x'^2}{2}+xx'} = \exp\left(-\frac{(x-x')^2}{2}\right),$$

which is the Gaussian kernel with $\sigma = \sqrt{2}$*. By the uniqueness of the Taylor series,* (2.18) *holds true and, since* $\phi_\ell = f_{e_\ell}$ *where* $(e_\ell)_\ell$ *is the canonical base of* ℓ^2*, we get that* $(\phi_\ell)_{\ell \in \mathbb{N}}$ *is a base of* $\mathcal{H}$*. Furthermore, given* $f \in \mathcal{H}$

$$e^{\frac{x^2}{2}} f(x) = \sum_{\ell=0}^{+\infty} a_\ell \frac{x^\ell}{\sqrt{\ell!}} \implies a_\ell = \frac{1}{\sqrt{\ell!}} D^\ell (e^{\frac{x^2}{2}} f(x))|_{x=0}$$

where $f^{(\ell)}$ *denotes the* ℓ*-th derivative of a function* f*.*

The above example can be generalized to any sequence $(\phi_\ell)_{\ell \in \mathbb{N}}$ of functions from a set $\mathcal{X}$ to $\mathbb{R}$ such that

$$\sum_{\ell=0}^{+\infty} \phi_\ell(x)^2 < +\infty \quad x \in \mathcal{X}.$$

It is easy to prove that

$$\mathcal{H} = \{f : \mathbb{R} \to \mathbb{R} \mid f(x) = \sum_{\ell=1}^{+\infty} a_\ell \phi_\ell(x), \sum_{\ell=1}^{+\infty} a_\ell^2 < +\infty\}$$

is a reproducing kernel Hilbert space with kernel

$$K(x, x') = \sum_{\ell=1}^{+\infty} \phi_\ell(x) \phi_\ell(x').$$

The Representer Theorem

The Representer Theorem is a fundamental result in regularization theory that characterizes the form of optimal solutions. Consider the Tikhonov regularization problem:

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_w(x_i)) + \lambda \|w\|^2 \tag{2.22}$$

where ℓ is an arbitrary loss function, $f_w(x) = w^T x$ is the linear model, and $\lambda > 0$ is the regularization parameter.

Theorem 2.13 (Representer Theorem)**.** *The optimal solution* w^* *of problem* (2.22) *can always be written as a linear combination of the training points:*

$$w^* = \sum_{i=1}^{n} \alpha_i x_i = X_n^T \alpha \tag{2.23}$$

where $X_n \in \mathbb{R}^{n \times D}$ *is the data matrix and* $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$ *is the coefficient vector.*

The immediate consequence is that the predictive function can be written as:

$$f(x) = w^{*T} x = \sum_{i=1}^{n} \alpha_i (x_i^T x) \tag{2.24}$$

Note that the prediction depends on the training data only through inner products $x_i^T x$.

Representer Theorem with Kernels

[Rosasco, 2023]

The remarkable property of using kernels is that it allows us to reuse the Representer Theorem in a more general form, by replacing the standard inner products with a kernel function $K(x, x')$.

Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric and positive definite kernel. Then the predictive function takes the form:

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) \tag{2.25}$$

and the optimization problem becomes:

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} \ell \left(y_i, \sum_{j=1}^{n} K(x_j, x_i)\alpha_j \right) + \lambda \alpha^T K \alpha \tag{2.26}$$

where $K \in \mathbb{R}^{n \times n}$ is the kernel matrix with entries $(K)_{ij} = K(x_i, x_j)$.
This result has two fundamental implications:

- The optimization problem is reduced from $\mathbb{R}^D$ to $\mathbb{R}^n$: instead of searching for optimal weights in dimension D (potentially very high or infinite), we search for coefficients c in dimension n.
- It is not necessary to explicitly compute the feature map $\Phi(x)$: all computations depend only on kernel values $K(x, x')$, allowing us to work implicitly in spaces of arbitrary dimension.

Kernel Ridge Regression Formulation (Scalar-Valued)

We consider N training examples $\{(x_i, y_i)\}_{i=1}^N$ with inputs $x_i \in \mathcal{X}$ and scalar-valued outputs $y_i \in \mathbb{R}$. Let $\mathcal{H}$ be a scalar-valued Reproducing Kernel Hilbert Space (RKHS) associated with a scalar reproducing kernel

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \tag{2.27}$$

so that for each $x, x' \in \mathcal{X}$, the value $K(x, x')$ is a real number.

The Kernel Ridge Regression objective is:

$$\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^2 \tag{2.28}$$

where $\lambda > 0$ is the regularization parameter.

Solution via Representer Theorem

[Schölkopf and Smola, 2002] By the Representer Theorem for scalar-valued RKHSs, the optimal solution has the form:

$$f(x) = \sum_{i=1}^{N} K(x_i, x)\alpha_i, \qquad \alpha_i \in \mathbb{R} \tag{2.29}$$

where $\alpha_i \in \mathbb{R}$ are scalar coefficients.

Matrix Formulation

We introduce the coefficient vector and output vector:

$$\alpha := \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} \in \mathbb{R}^N, \qquad y := \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N \tag{2.30}$$

Define the kernel matrix $K(X, X) \in \mathbb{R}^{N \times N}$ whose (i, j)-th entry is the scalar $K(x_i, x_j)$:

$$K(X, X) := \begin{bmatrix} K(x_1, x_1) & \cdots & K(x_1, x_N) \\ \vdots & \ddots & \vdots \\ K(x_N, x_1) & \cdots & K(x_N, x_N) \end{bmatrix} \tag{2.31}$$

With this notation, the predictions on all training inputs (stacked) are:

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{bmatrix} = K(X, X)\alpha \tag{2.32}$$

Derivation of the Closed-Form Solution

Using the coefficient vector and kernel matrix, the objective becomes:

$$\min_{\alpha \in \mathbb{R}^N} \frac{1}{N} \|y - K(X, X)\alpha\|_2^2 + \lambda \alpha^T K(X, X)\alpha \tag{2.33}$$

For notational simplicity, let $K := K(X, X)$.

Compact Approach: Matrix Formulation

The objective function can be written as:

$$L(\alpha) = \frac{1}{N}(y - K\alpha)^T(y - K\alpha) + \lambda \alpha^T K \alpha \tag{2.34}$$

Expanding:

$$L(\alpha) = \frac{1}{N}(y^T y - 2y^T K\alpha + \alpha^T K^T K\alpha) + \lambda\alpha^T K\alpha \tag{2.35}$$

$$= \frac{1}{N}y^T y - \frac{2}{N}y^T K\alpha + \frac{1}{N}\alpha^T K^2\alpha + \lambda\alpha^T K\alpha \tag{2.36}$$

where we used the symmetry of the kernel matrix: $K^T = K$.

Gradient Computation

Taking the gradient with respect to α:

$$\frac{\partial}{\partial\alpha}L(\alpha) = -\frac{2}{N}Ky + \frac{2}{N}K^2\alpha + 2\lambda K\alpha \tag{2.37}$$

Setting the gradient to zero:

$$-\frac{2}{N}Ky + \frac{2}{N}K^2\alpha + 2\lambda K\alpha = 0 \tag{2.38}$$

Final Solution

Simplifying:

$$Ky = K^2\alpha + N\lambda K\alpha \tag{2.39}$$

$$Ky = K(K + N\lambda I_N)\alpha \tag{2.40}$$

Assuming K is invertible on its range (or equivalently, working in the subspace spanned by the kernel), we can left-multiply by K^{-1}:

$$y = (K + N\lambda I_N)\alpha \tag{2.41}$$

Therefore, the optimal coefficient vector is:

$$\boxed{\alpha = (K(X,X) + N\lambda I_N)^{-1}y} \tag{2.42}$$

In practice, we often absorb the factor N into λ and write:

$$\boxed{\alpha = (K(X,X) + \lambda I_N)^{-1}y} \tag{2.43}$$

Prediction on New Data

For a test input x_*, we define the kernel vector:

$$k_{x_*} := \begin{bmatrix} K(x_*, x_1) \\ \vdots \\ K(x_*, x_N) \end{bmatrix} \in \mathbb{R}^N \tag{2.44}$$

where each entry $K(x_*, x_i)$ is a scalar. The predicted value $f(x_*) \in \mathbb{R}$ is:

$$f(x_*) = \sum_{i=1}^{N} K(x_i, x_*)\alpha_i = k_{x_*}^T\alpha \tag{2.45}$$

Relationship to Least Squares

Note that the KRR solution can be viewed as a regularized least squares problem in the feature space induced by the kernel K. If we define the implicit feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that $K(x, x') = \langle\phi(x), \phi(x')\rangle_{\mathcal{H}}$, then the kernel matrix can be written as:

$$K = \Phi\Phi^T \tag{2.46}$$

where $\Phi = [\phi(x_1), \ldots, \phi(x_N)]^T$ is the feature matrix.

The KRR solution:

$$\alpha = (K + \lambda I_N)^{-1}y = (\Phi\Phi^T + \lambda I_N)^{-1}y \tag{2.47}$$

is equivalent to solving ridge regression in the feature space, demonstrating the "kernel trick" that allows us to work in high (or infinite) dimensional spaces through inner products alone.

Kernel Ridge Regression Formulation (Vector-Valued)

We consider N training examples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ with inputs $\mathbf{x}_i \in \mathcal{X}$ and vector-valued outputs $\mathbf{y}_i \in \mathbb{R}^D$. Let $\mathcal{H}$ be a vector-valued Reproducing Kernel Hilbert Space (RKHS) associated with a matrix-valued reproducing kernel

$$\mathbf{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{D \times D}, \tag{2.48}$$

so that for each $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, the value $\mathbf{K}(\mathbf{x}, \mathbf{x}')$ is a $D \times D$ matrix.

The Kernel Ridge Regression objective is:

$$\min_{\mathbf{f} \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{f}(\mathbf{x}_i)\|_2^2 + \lambda \|\mathbf{f}\|_{\mathcal{H}}^2 \tag{2.49}$$

where $\lambda > 0$ is the regularization parameter.

Solution via Representer Theorem

[Alvarez et al., 2012] By the Representer Theorem for vector-valued RKHSs , the optimal solution has the form:

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{K}(\mathbf{x}_i, \mathbf{x}) \mathbf{c}_i, \qquad \alpha_i \in \mathbb{R}^D \tag{2.50}$$

where $\alpha_i \in \mathbb{R}^D$ are coefficient vectors.

Matrix Formulation

We introduce the concatenated coefficient vector and concatenated output vector:

$$\overline{\alpha} := \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} \in \mathbb{R}^{ND}, \qquad \overline{\mathbf{y}} := \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{bmatrix} \in \mathbb{R}^{ND} \tag{2.51}$$

Define the block kernel matrix $\mathbf{K}(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{ND \times ND}$ whose (i, j)-th block is the $D \times D$ matrix $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$:

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) := \begin{bmatrix} \mathbf{K}(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \mathbf{K}(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ \mathbf{K}(\mathbf{x}_N, \mathbf{x}_1) & \cdots & \mathbf{K}(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \tag{2.52}$$

With this notation, the predictions on all training inputs (stacked) are:

$$\begin{bmatrix} \mathbf{f}(\mathbf{x}_1) \\ \vdots \\ \mathbf{f}(\mathbf{x}_N) \end{bmatrix} = \mathbf{K}(\mathbf{X}, \mathbf{X}) \overline{\alpha} \tag{2.53}$$

Derivation of the Closed-Form Solution

[Schölkopf and Smola, 2002]

Using the concatenated vectors and block kernel matrix, the objective becomes:

$$\min_{\overline{\alpha} \in \mathbb{R}^{ND}} \frac{1}{N} \|\overline{\mathbf{y}} - \mathbf{K}(\mathbf{X}, \mathbf{X}) \overline{\alpha}\|_2^2 + \lambda \overline{\alpha}^T \mathbf{K}(\mathbf{X}, \mathbf{X}) \overline{\alpha} \tag{2.54}$$

For notational simplicity, let $\mathbf{K} := \mathbf{K}(\mathbf{X}, \mathbf{X})$.

Compact Approach: Matrix Formulation

The objective function can be written as:

$$L(\overline{\alpha}) = \frac{1}{N}(\overline{\mathbf{y}} - \mathbf{K}\overline{\alpha})^T(\overline{\mathbf{y}} - \mathbf{K}\overline{\alpha}) + \lambda\overline{\alpha}^T\mathbf{K}\overline{\alpha} \tag{2.55}$$

Expanding:

$$L(\overline{\alpha}) = \frac{1}{N}(\overline{\mathbf{y}}^T\overline{\mathbf{y}} - 2\overline{\mathbf{y}}^T\mathbf{K}\overline{\alpha} + \overline{\alpha}^T\mathbf{K}^T\mathbf{K}\overline{\alpha}) + \lambda\overline{\alpha}^T\mathbf{K}\overline{\alpha} \tag{2.56}$$

$$= \frac{1}{N}\overline{\mathbf{y}}^T\overline{\mathbf{y}} - \frac{2}{N}\overline{\mathbf{y}}^T\mathbf{K}\overline{\alpha} + \frac{1}{N}\overline{\alpha}^T\mathbf{K}^2\overline{\alpha} + \lambda\overline{\alpha}^T\mathbf{K}\overline{\alpha} \tag{2.57}$$

where we used the symmetry of the block kernel matrix: $\mathbf{K}^T = \mathbf{K}$.

Gradient Computation

Taking the gradient with respect to $\overline{\alpha}$:

$$\frac{\partial}{\partial\overline{\alpha}}L(\overline{\alpha}) = -\frac{2}{N}\mathbf{K}\overline{\mathbf{y}} + \frac{2}{N}\mathbf{K}^2\overline{\alpha} + 2\lambda\mathbf{K}\overline{\alpha} \tag{2.58}$$

Setting the gradient to zero:

$$-\frac{2}{N}\mathbf{K}\overline{\mathbf{y}} + \frac{2}{N}\mathbf{K}^2\overline{\alpha} + 2\lambda\mathbf{K}\overline{\alpha} = \mathbf{0} \tag{2.59}$$

Final Solution

Simplifying:

$$\mathbf{K}\overline{\mathbf{y}} = \mathbf{K}^2\overline{\alpha} + N\lambda\mathbf{K}\overline{\alpha} \tag{2.60}$$

$$\mathbf{K}\overline{\mathbf{y}} = \mathbf{K}(\mathbf{K} + N\lambda\mathbf{I}_{ND})\overline{\alpha} \tag{2.61}$$

Assuming $\mathbf{K}$ is invertible on its range (or equivalently, working in the subspace spanned by the kernel blocks), we can left-multiply by $\mathbf{K}^{-1}$:

$$\overline{\mathbf{y}} = (\mathbf{K} + N\lambda\mathbf{I}_{ND})\overline{\alpha} \tag{2.62}$$

Therefore, the optimal concatenated coefficient vector is:

$$\boxed{\overline{\alpha} = (\mathbf{K}(\mathbf{X},\mathbf{X}) + N\lambda\mathbf{I}_{ND})^{-1}\overline{\mathbf{y}}} \tag{2.63}$$

In practice, we often absorb the factor N into λ and write:

$$\boxed{\overline{\alpha} = (\mathbf{K}(\mathbf{X},\mathbf{X}) + \lambda\mathbf{I}_{ND})^{-1}\overline{\mathbf{y}}} \tag{2.64}$$

Prediction on New Data

For a test input $\mathbf{x}_*$, we define the block column vector:

$$\mathbf{K}_{\mathbf{x}_*} := \begin{bmatrix} \mathbf{K}(\mathbf{x}_*, \mathbf{x}_1) \\ \vdots \\ \mathbf{K}(\mathbf{x}_*, \mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{ND\times D} \tag{2.65}$$

Note that each block $\mathbf{K}(\mathbf{x}_*, \mathbf{x}_i)$ is a $D \times D$ matrix. The predicted vector $\mathbf{f}(\mathbf{x}_*) \in \mathbb{R}^D$ is:

$$\mathbf{f}(\mathbf{x}_*) = \sum_{i=1}^{N}\mathbf{K}(\mathbf{x}_i, \mathbf{x}_*)\alpha_i = \mathbf{K}_{\mathbf{x}_*}^T\,\overline{\alpha} \tag{2.66}$$

3 Modeling Natural Language via Markov Chains

In this chapter we present the mathematical formulation of the text generation problem using discrete-time Markov chains, and describe the computational implementation of both first-order and higher-order models.

3.1 Mathematical Formulation of the Problem

The Text Generation Problem

Let $\mathcal{V} = \{v_1, v_2, \ldots, v_V\}$ be a finite vocabulary of size V, where each v_i represents a unique token (word or symbol). A text corpus can be represented as a sequence of tokens:

$$w_1, w_2, \ldots, w_N \in \mathcal{V} \tag{3.1}$$

where N is the total length of the corpus.

The **text generation problem** consists in predicting the next token w_{t+1} given the history of previous tokens $w_1, \ldots, w_t$. In full generality, we seek to estimate the conditional distribution:

$$P(w_{t+1} \mid w_1, w_2, \ldots, w_t) \tag{3.2}$$

However, modeling the complete dependency structure is computationally intractable and requires estimating an exponentially large number of parameters.

The First-Order Markov Assumption

To make the problem tractable, we adopt the **first-order Markov assumption**: the next token depends exclusively on the current token. Formally:

$$P(w_{t+1} = v_j \mid w_1, \ldots, w_t = v_i) = P(w_{t+1} = v_j \mid w_t = v_i) \tag{3.3}$$

This assumption allows us to model the text generation process as a discrete-time, finite-state, homogeneous Markov chain $(X_t)_{t\in\mathbb{N}}$ on the state space $I = \{1, 2, \ldots, V\}$ (see Definition 1.2), where:

- Each state $i \in I$ corresponds to a token $v_i \in \mathcal{V}$;
- $X_t = i$ means that the t-th token in the sequence is v_i;
- The sequence $X_1, X_2, \ldots, X_N$ represents the observed text corpus.

The model is completely characterized by the transition matrix $P = (p_{ij})_{i,j\in I}$ and the initial distribution μ on I, as established in Chapter 1.

State Space Construction

Vocabulary Construction

Given a text corpus $\mathcal{C}$, we extract the vocabulary $\mathcal{V}$ by collecting all unique tokens that appear in $\mathcal{C}$. In practice, we apply the following preprocessing steps:

1. **Tokenization**: splitting the text into individual tokens;
2. **Normalization**: conversion to lowercase (optional);
3. **Vocabulary extraction**: identification of unique tokens.

Let $\mathcal{V} = \{v_1, v_2, \ldots, v_V\}$ denote the resulting vocabulary, ordered lexicographically.

Mapping Between Tokens and States

We define two bijective mappings to establish the correspondence between tokens and states:

Definition 3.1 (Word-to-index mapping)**.**

$$\phi : \mathcal{V} \to I, \quad v_i \mapsto i \tag{3.4}$$

Definition 3.2 (Index-to-word mapping)**.**

$$\psi : I \to \mathcal{V}, \quad i \mapsto v_i \tag{3.5}$$

where $\psi = \phi^{-1}$.
These mappings allow us to:

- convert a sequence of tokens $w_1, \ldots, w_N \in \mathcal{V}$ into a sequence of states $\phi(w_1), \ldots, \phi(w_N) \in I$;
- interpret the state $i \in I$ of the Markov chain as a representation of the token $\psi(i) = v_i$.

Representation as a Markov Chain

Given a text sequence $w_1, w_2, \ldots, w_N$, we construct the corresponding Markov chain realization:

$$X_1 = \phi(w_1), \quad X_2 = \phi(w_2), \quad \ldots, \quad X_N = \phi(w_N) \tag{3.6}$$

The transition from word w_t to word w_{t+1} corresponds to the transition from state $X_t = \phi(w_t)$ to state $X_{t+1} = \phi(w_{t+1})$ in the Markov chain. This construction allows us to apply the complete theory of Chapter 1 to analyze the structure and dynamics of natural language.

Empirical Estimation of the Transition Matrix

The Transition Matrix

Recall from Definition 1.3 that the transition matrix $P = (p_{ij})_{i,j \in I}$ is stochastic: $p_{ij} \geq 0$ and $\sum_{j \in I} p_{ij} = 1$ for all i. For the text generation model,

$$p_{ij} = \begin{cases} P(X_{n+1} = j \mid X_n = i), & \text{if } P(X_n = i) > 0, \\ 1, & \text{if } P(X_n = i) = 0 \text{ and } i = j, \\ 0, & \text{if } P(X_n = i) = 0 \text{ and } i \neq j. \end{cases} \tag{3.7}$$

Maximum Likelihood Estimator

Given an observed sequence of states $X_1, X_2, \ldots, X_N$, we estimate the transition probabilities using the **maximum likelihood estimator** (MLE).

Definition 3.3 (Transition count)**.** *Let $N(i, j)$ denote the number of times state j follows state i in the sequence:*

$$N(i, j) = \sum_{t=1}^{N-1} \mathbb{1}_{\{X_t = i,\, X_{t+1} = j\}} \tag{3.8}$$

where $\mathbb{1}_A$ is the indicator function of event A.

Definition 3.4 (Maximum likelihood estimator)**.** *The MLE for the transition probability is:*

$$\hat{p}_{ij} = \frac{N(i, j)}{\sum_{k \in I} N(i, k)} = \frac{N(i, j)}{N(i)} \tag{3.9}$$

where $N(i) = \sum_{k \in I} N(i, k)$ is the total number of times state i appears in positions $1, \ldots, N - 1$.

Justification of the Estimator

Proposition 3.5. *The estimator $\hat{p}_{ij}$ is the maximum likelihood estimator for p_{ij}.*

Proof. Consider the likelihood of observing the sequence $X_1, \ldots, X_N$ given the transition matrix P:

$$L(P) = P(X_1) \prod_{t=1}^{N-1} P(X_{t+1} \mid X_t) = P(X_1) \prod_{t=1}^{N-1} p_{X_t, X_{t+1}} \tag{3.10}$$

Taking the logarithm:

$$\log L(P) = \log P(X_1) + \sum_{t=1}^{N-1} \log p_{X_t, X_{t+1}} = \log P(X_1) + \sum_{i,j \in I} N(i,j) \log p_{ij} \tag{3.11}$$

Maximizing with respect to P subject to the stochastic constraints $\sum_j p_{ij} = 1$ yields:

$$\hat{p}_{ij} = \frac{N(i,j)}{N(i)} \tag{3.12}$$

□

Ergodic Theorem and Consistency of the Estimator

The fundamental theoretical justification for using the empirical estimator $\hat{p}_{ij}$ is provided by Theorem 1.23. This result is crucial because the observed pairs (X_t, X_{t+1}) are neither independent nor identically distributed, violating the standard assumption of most statistical methods.

Strong Consistency of the Estimator

Proposition 3.6 (Consistency)**.** *For all $i, j \in I$ such that $\pi_i > 0$, the estimator $\hat{p}_{ij} = N(i,j)/N(i)$ is strongly consistent:*

$$\hat{p}_{ij} \xrightarrow[N \to \infty]{a.s.} p_{ij} \tag{3.13}$$

Proof. By Theorem 1.23, for all $i, j \in I$ we have almost surely:

$$\frac{N(i,j)}{N-1} \xrightarrow[N \to \infty]{} \pi_i p_{ij}, \qquad \frac{N(i)}{N-1} \xrightarrow[N \to \infty]{} \pi_i. \tag{3.14}$$

Since $\pi_i > 0$, it follows that:

$$\hat{p}_{ij} = \frac{N(i,j)/(N-1)}{N(i)/(N-1)} \xrightarrow[N \to \infty]{\text{a.s.}} \frac{\pi_i p_{ij}}{\pi_i} = p_{ij}. \tag{3.15}$$

□

Unbiasedness of the Estimator

Proposition 3.7 (Unbiasedness)**.** *For all $i, j \in I$ with $N(i) > 0$:*

$$\mathbb{E}[\hat{p}_{ij} \mid N(i)] = p_{ij} \tag{3.16}$$

Proof. Let $\tau_1 < \tau_2 < \cdots < \tau_{N(i)}$ denote the successive times at which the chain visits state i in positions $1, \ldots, N-1$, i.e.

$$\tau_r = \inf\{t > \tau_{r-1} : X_t = i\}, \qquad r = 1, \ldots, N(i),$$

with $\tau_0 = 0$.

For each visit define the indicator

$$Y_r = \mathbb{1}_{\{X_{\tau_r + 1} = j\}}, \qquad r = 1, \ldots, N(i).$$

Then

$$N(i,j) = \sum_{r=1}^{N(i)} Y_r.$$

By the (strong) Markov property applied at the stopping times τ_r, the conditional distribution of the next state after each visit to i is

$$\mathbb{P}(X_{\tau_r+1} = j \mid X_{\tau_r} = i) = p_{ij}.$$

Therefore

$$\mathbb{E}[Y_r \mid N(i)] = p_{ij} \qquad \text{for all } r = 1, \dots, N(i).$$

Using linearity of expectation,

$$\mathbb{E}[N(i,j) \mid N(i) = n] = \sum_{r=1}^{n} \mathbb{E}[Y_r \mid N(i) = n] = n\, p_{ij}.$$

Hence

$$\mathbb{E}[\hat{p}_{ij} \mid N(i) = n] = \frac{1}{n}\, \mathbb{E}[N(i,j) \mid N(i) = n] = p_{ij}.$$

This proves

$$\mathbb{E}[\hat{p}_{ij} \mid N(i)] = p_{ij}.$$

□

Handling Unobserved States

For states i with $N(i) = 0$ (never observed as a non-final token), we define:

$$\hat{p}_{ij} = \begin{cases} 1, & \text{if } j = i, \\ 0, & \text{if } j \neq i. \end{cases} \tag{3.17}$$

This makes state i absorbing and ensures that $\hat{P}$ is stochastic.

Remark 3.8. *An alternative approach consists in assigning uniform transition probabilities $p_{ij} = 1/V$ for all $j \in I$. However, the absorbing-state convention is more conservative and consistent with the formal definition adopted in Chapter 1.*

Decoding: From the Predicted Distribution to a Token

Once the transition distribution $\hat{p}_{i\,\cdot}$ has been estimated, a *decoding step* is required to map the predicted distribution back to a discrete token. Since the encoding $\phi : \mathcal{V} \to I$ is bijective, the inverse map ϕ^{-1} is well defined, and decoding reduces to selecting an index $j \in I$ and returning the corresponding vocabulary word $\phi^{-1}(j)$. Two natural strategies are available.

Greedy decoding. The simplest choice is to always select the most probable next token:

$$\hat{w}_{t+1} = \phi^{-1}\left(\arg\max_{j \in I} \hat{p}_{ij} \right).$$

Greedy decoding is deterministic and computationally trivial, requiring only a single pass over the distribution. Its main drawback is a tendency to produce repetitive or degenerate sequences: once the chain reaches a high-probability cycle — a sequence of states where each token deterministically leads to the next — it cannot escape, and the generated text degenerates into an infinite loop.

Sampling. A more flexible alternative is to draw the next token directly from the estimated distribution:

$$j \sim \hat{p}_{i\,\cdot}, \qquad \hat{w}_{t+1} = \phi^{-1}(j).$$

This stochastic approach produces more varied and natural-sounding sequences, since low-probability transitions can still be selected occasionally. To control the trade-off between diversity and coherence, one can introduce a temperature parameter $\tau > 0$ and sample from the tempered distribution:

$$\hat{p}_{ij}^{(\tau)} = \frac{\hat{p}_{ij}^{1/\tau}}{\sum_{j' \in I} \hat{p}_{ij'}^{1/\tau}}.$$

As $\tau \to 0$ the tempered distribution concentrates on the argmax, recovering greedy decoding. For $\tau = 1$ one recovers the original estimated distribution, and for $\tau > 1$ the distribution becomes flatter, increasing diversity at the cost of lower average coherence.

3.2 Implementation of Higher-Order Models

From First-Order to Higher-Order via State Space Lifting

As established in the previous chapter (Proposition 1.25), a k-th order Markov chain on I can be represented as a first-order chain on the lifted state space $\widehat{I} = I^k$. The implementation exploits this reduction directly: rather than introducing separate estimation algorithms for each order, the code constructs the lifted chain $(\widehat{X}_n)$ and applies the same first-order machinery.

The lifted state at time n is the tuple

$$\widehat{X}_n = (X_n, X_{n-1}, \ldots, X_{n-k+1}) \in I^k,$$

and the transition probabilities of the lifted chain are

$$\widehat{p}_{\mathbf{ij}} = \begin{cases} \hat{p}(j_1 \mid i_1, \ldots, i_k), & \text{if } \mathbf{j} = (j_1, i_1, \ldots, i_{k-1}), \\ 0, & \text{otherwise,} \end{cases}$$

where $\hat{p}(j_1 \mid i_1, \ldots, i_k)$ is the empirical conditional frequency of j_1 following the context $(i_1, \ldots, i_k)$ in the training corpus (the MLE of the k-th order transition probability).

Data Structures and Complexity

The lifted state space has cardinality $|\widehat{I}| = V^k$, which grows exponentially in the order k. For moderate values ($k \leq 2$ with V not too large), the transition matrix $\widehat{P}$ can be stored as a sparse matrix of dimension $V^k \times V^k$: each row contains at most V nonzero entries, reflecting the shift structure of the lifted chain. For larger orders or vocabularies, storing even the nonzero entries of a $V^k \times V^k$ sparse matrix becomes impractical; in this regime the implementation falls back to a dictionary representation, storing only the observed contexts and their empirical transition distributions.

Concretely, the code selects the sparse-matrix representation when $V^k < 10^7$, and the dictionary representation otherwise. In both cases the n-gram extraction procedure is identical: from the token index sequence $x_1, x_2, \ldots, x_N$ we extract every $(k+1)$-tuple

$$\big((x_t, x_{t+1}, \ldots, x_{t+k-1}),\ x_{t+k}\big), \qquad t = 1, \ldots, N-k,$$

count co-occurrences, and normalize each row to obtain the empirical distribution.

Transition Kernel on the Lifted Space

Beyond the prediction task, the implementation provides a measure of similarity between two length-k contexts. Given two lifted states $\mathbf{i}, \mathbf{j} \in \widehat{I}$, the **transition kernel** is defined as the inner product of their empirical transition distributions:

$$K(\mathbf{i}, \mathbf{j}) = \sum_{s \in I} \widehat{P}[\mathbf{i}, s]\ \widehat{P}[\mathbf{j}, s]. \tag{3.18}$$

This quantity lies in $[0, 1]$: it equals 1 when the two contexts induce identical distributions over next tokens, and 0 when the distributions have disjoint supports. It can be interpreted as a positive semi-definite kernel on the space of contexts, measuring how interchangeable two contexts are from the point of view of the estimated language model.

Empirical Evaluation

[Wikipedia contributors, 2026]

Information-theoretic background. The evaluation metrics used in this section are grounded in classical information theory. These quantities provide a natural way to measure the quality of probabilistic models, since they quantify how well a model approximates the true distribution that generates the data.

Let X be a discrete random variable with distribution P. The *entropy* of X is defined as

$$H(X) = -\sum_{x} P(x) \log P(x), \tag{3.19}$$

and measures the average amount of information (or uncertainty) associated with observing the outcome of X. Entropy is minimal when the distribution is concentrated on a single outcome and maximal when the distribution is uniform. In the context of language, entropy quantifies the intrinsic unpredictability of tokens drawn from a vocabulary.

For sequential data such as text, the relevant quantity is the *conditional entropy*

$$H(X_t \mid X_{t-k:t-1}) = -\sum_{\mathbf{i},j} P(\mathbf{i}, j) \log P(j \mid \mathbf{i}), \tag{3.20}$$

which measures the average uncertainty of the next token X_t given a context $\mathbf{i} = (X_{t-k}, \ldots, X_{t-1})$ of length k. For a Markov language model, this quantity characterizes the intrinsic predictability of the text-generating process: lower conditional entropy means that the next word is largely determined by its context.

In practice the true distribution P is unknown and must be approximated from a finite training corpus. A Markov model of order k provides an estimate $\hat{P}(j \mid \mathbf{i})$ of the conditional distribution. To evaluate the quality of this approximation on unseen data we use the *cross-entropy*

$$H(P, \hat{P}) = -\sum_{\mathbf{i},j} P(\mathbf{i}, j) \log \hat{P}(j \mid \mathbf{i}), \tag{3.21}$$

which measures the expected number of nats required to encode a token drawn from the true distribution when using a code optimized for the model $\hat{P}$. The normalized test log-likelihood reported in this work is an empirical estimate of $-H(P, \hat{P})$ computed on the held-out corpus.

The difference between cross-entropy and entropy is given by the *Kullback–Leibler divergence*

$$D_{\mathrm{KL}}(P \,\|\, \hat{P}) = \sum_{x} P(x) \log \frac{P(x)}{\hat{P}(x)}, \tag{3.22}$$

which measures the discrepancy between two probability distributions. The KL divergence is always non-negative and equals zero if and only if the two distributions coincide.

In the context of language modeling, P represents the true transition distribution of the underlying text-generating process, while $\hat{P}$ is the distribution estimated from the training corpus. The KL divergence therefore quantifies how much information is lost when the model $\hat{P}$ is used in place of the true distribution P.

For Markov chains the divergence can be decomposed over contexts,

$$D_{\mathrm{KL}} = \sum_{\mathbf{i}} \pi(\mathbf{i}) \sum_{j} P(j \mid \mathbf{i}) \log \frac{P(j \mid \mathbf{i})}{\hat{P}(j \mid \mathbf{i})}, \tag{3.23}$$

where $\pi(i)$ denotes the stationary probability of context i, that is, the long–run probability that the system is in context i. Intuitively, $\pi(i)$ represents how frequently context i occurs in the sequence generated by the source, and therefore acts as a weight when averaging quantities over all contexts. This decomposition shows that the divergence measures the average mismatch between the true and estimated conditional transition distributions across all contexts.

These information-theoretic quantities are particularly suitable for evaluating Markov language models because they directly assess the quality of the learned probability distribution. A model that closely matches the true transition structure of the language achieves lower cross-entropy and smaller KL divergence, meaning that it assigns high probability to the sequences that actually occur in the data.

Let $X_1^{\mathrm{test}}, X_2^{\mathrm{test}}, \ldots, X_M^{\mathrm{test}}$ be a test sequence drawn from the same chain but disjoint from the training corpus, with empirical transition counts

$$N_{\mathrm{test}}(i, j) = \sum_{t=1}^{M-1} \mathbb{1}_{\{X_t^{\mathrm{test}}=i,\, X_{t+1}^{\mathrm{test}}=j\}}, \qquad N_{\mathrm{test}}(i) = \sum_{j \in I} N_{\mathrm{test}}(i, j),$$

and empirical transition frequencies $\tilde{p}_{ij} = N_{\mathrm{test}}(i, j)/N_{\mathrm{test}}(i)$ for $N_{\mathrm{test}}(i) > 0$. We use $\hat{P}$ to denote the transition matrix estimated on the training corpus.

First-Order Case

Test log-likelihood. Consider a test sequence $X_1^{\text{test}}, X_2^{\text{test}}, \ldots, X_M^{\text{test}}$ generated by the same Markov chain but disjoint from the training corpus. Let

$$N_{\text{test}}(i,j) = \sum_{t=1}^{M-1} \mathbb{1}_{\{X_t^{\text{test}}=i,\ X_{t+1}^{\text{test}}=j\}}$$

denote the number of observed transitions from state i to state j in the test sequence, and

$$N_{\text{test}}(i) = \sum_{j \in I} N_{\text{test}}(i,j)$$

the number of times state i appears as a predecessor.

Given the transition matrix $\hat{P}$ estimated from the training corpus, the log-likelihood of the test sequence under the model is

$$\ell_{\text{test}}(\hat{P}) = \sum_{i,j \in I} N_{\text{test}}(i,j)\, \log \hat{p}_{ij}. \tag{3.24}$$

This quantity measures how well the estimated model explains unseen data. It is the natural out-of-sample analogue of the training log-likelihood maximized by the maximum likelihood estimator.

By the ergodic theorem (Theorem 1.23), as $M \to \infty$, the normalized log-likelihood converges almost surely to

$$\frac{1}{M}\, \ell_{\text{test}}(\hat{P}) \xrightarrow{\text{a.s.}} \sum_{i,j \in I} \pi_i p_{ij} \log \hat{p}_{ij}, \tag{3.25}$$

where π is the stationary distribution of the chain.

This limit can be decomposed into two fundamental information-theoretic quantities:

$$\sum_{i,j} \pi_i p_{ij} \log \hat{p}_{ij} = -H(X_1 \mid X_0) - D_{\text{KL}}(P \,\|\, \hat{P})_\pi, \tag{3.26}$$

where

$$H(X_1 \mid X_0) = -\sum_{i,j} \pi_i p_{ij} \log p_{ij}$$

is the conditional entropy of the chain, and

$$D_{\text{KL}}(P \,\|\, \hat{P})_\pi = \sum_{i \in I} \pi_i\, D_{\text{KL}}(p_{i\cdot} \,\|\, \hat{p}_{i\cdot}) \geq 0 \tag{3.27}$$

is the stationary-weighted KL divergence between the true and estimated transition matrices.

This identity shows that the normalized test log-likelihood is maximized precisely when the estimated transition matrix coincides with the true one. Any deviation from the optimum is measured by the weighted KL divergence (3.27), which quantifies the average mismatch between the true and estimated transition distributions across states.

Stationary-weighted KL divergence. Since the true transition probabilities p_{ij} are unknown, the quantity (3.27) cannot be computed directly. Instead we estimate it using empirical transition frequencies obtained from the test sequence.

Let

$$\tilde{p}_{ij} = \frac{N_{\text{test}}(i,j)}{N_{\text{test}}(i)}$$

denote the empirical transition probability from state i to state j for all states with $N_{\text{test}}(i) > 0$. An empirical estimator of the weighted KL divergence is then

$$\widehat{D}_{\text{KL}} = \sum_{i \in I} \hat{\pi}_i\, D_{\text{KL}}(\tilde{p}_{i\cdot} \,\|\, \hat{p}_{i\cdot}), \tag{3.28}$$

where $\hat{\pi}$ is the empirical stationary distribution estimated from the training sequence via Theorem 1.23.

The per-state divergence is given by

$$D_{\mathrm{KL}}(\tilde{p}_{i\cdot} \,\|\, \hat{p}_{i\cdot}) = \sum_{j \in I} \tilde{p}_{ij} \log \frac{\tilde{p}_{ij}}{\hat{p}_{ij}}, \tag{3.29}$$

where the sum is taken only over indices j such that $\tilde{p}_{ij} > 0$.

If $\hat{p}_{ij} = 0$ while $\tilde{p}_{ij} > 0$, the divergence becomes $+\infty$, indicating that the model assigns zero probability to an event that occurs in the data. This situation corresponds exactly to the unobserved-transition problem discussed in the previous section.

3.3 Experimental Results

The TinyStories Dataset

The experiments in this chapter are conducted on the TinyStories V2-GPT4 dataset Eldan and Li [2023], a corpus of short stories in English generated by GPT-4. The dataset was originally designed to study the language acquisition capabilities of small language models, with the explicit goal of producing grammatically correct and semantically coherent narratives using only the vocabulary and syntactic structures accessible to a 3–4 year old child.

Each story consists of approximately 100–300 tokens and is delimited by the special token `<|endoftext|>`. The restricted and repetitive vocabulary makes the corpus particularly well-suited to our Markov chain framework: the limited range of linguistic dependencies reduces the effective memory of the process, making the first-order Markov assumption more plausible than it would be on a general-purpose corpus.

Experimental Setup

We use 500 stories for training (approximately 81,600 tokens) and 200 held-out stories for evaluation (approximately 33,400 tokens, of which 32,363 fall within the training vocabulary).

Text preprocessing consists of lowercase normalization and whitespace tokenization. The resulting vocabulary has $|\mathcal{V}| = 2{,}940$ unique tokens. Models of order $k \in \{1, 2, 4\}$ are evaluated on the same training/test split; in all cases the context window starts from the beginning of the token sequence and no padding is introduced.

Text Generation

We evaluate the qualitative behavior of each model by generating 80-token sequences starting from the phrase `once upon a time`. Each model uses only the last k words of this phrase as its actual initial context: `time` for $k = 1$, `a time` for $k = 2$, and the full `once upon a time` for $k = 4$. At each step the next token is sampled according to the estimated conditional distribution $\hat{p}(\cdot \mid \mathbf{c}_t)$ (greedy decoding with ties broken randomly).

Table 3.1: Sample generated sequences (80 tokens) for orders $k = 1, 2, 4$. The starting phrase is `once upon a time` in all cases, but each model receives only its last k words as the initial context.

Order	Initial context	Generated text
$k = 1$	`[once]`	once upon a small mouse and even and they played together all was too fast and forgot he realized it is rex does not know what to fall but suddenly a new doll and wanted to teach sue came to listen and played all the dog stopped crying his messy and we should share with me try dancing are also found a dog wagged his friend sam and the big yard there was a tent tim told the big fireplace
$k = 2$	`[once, upon]`	once upon a time there was a boy named timmy got sick he did not have a card for my family and i feel cold and sweet treats amy asked her dad said jump sue i ll wait for anything one day a little bird who was sad because it was a wise old turtle the turtle was also a busy little girl named sue she had ever seen tommy and said yes he puts the tower wobbling she runs
$k = 4$	`[once, upon, a, time]`	once upon a time there was a lovely cat named lucy lucy loved to play with all the animals once upon a time there was a little boy named tim went to the park with his mom to play with his friends he was having so much fun but then something unexpected happened the leash came off the dog the dog stopped and ran away sara and tom hugged their mom they were sorry we are sorry mom we did

The qualitative difference across orders is immediately apparent. The order-1 model has no memory beyond the single preceding token: the generated text is locally plausible at the bigram level but globally incoherent, jumping erratically between unrelated topics and characters. The order-2 model, which conditions on the last two tokens, already recovers short syntactic phrases and maintains local topic coherence for a few words at a time. The order-4 model produces the most fluent output, but at the cost of memorization: having seen most of its 4-gram contexts only once in training, the model tends to reproduce verbatim sub-sequences from the corpus, which explains the repeated appearance of stereotyped openings such as `once upon a time there was`. This memorization phenomenon is a direct consequence of the sparsity problem discussed in the next section: the longer the context window, the more deterministic the estimated conditional distribution becomes, and the less the model is able to generalize to unseen continuations.

Table 3.2 shows the top predicted next tokens for three representative 4-gram contexts in the order-4 model, illustrating that the estimated distributions are sharp and concentrated on a few successors.

Table 3.2: Top next-token predictions for three 4-gram contexts (order $k = 4$). Probabilities are the empirical conditional frequencies $\hat{p}_{ij}$.

Context	Next token	$\hat{p}$
`once upon a time`	`there`	0.847
	`in`	0.127
	`a`	0.023
	`two`	0.003
`there was a little`	`boy`	0.369
	`girl`	0.357
	`dog`	0.071
	`rabbit`	0.012
	`bunny`	0.012
`one day a little`	`girl`	0.419
	`boy`	0.339
	`bird`	0.048
	`red`	0.032
	`fish`	0.016

Sparsity of the Transition Matrix

A central difficulty of higher-order Markov models is the exponential growth of the lifted state space $\widehat{I} = \mathcal{V}^k$. Table 3.3 summarizes the key statistics for each order.

Table 3.3: State-space and sparsity statistics for orders $k = 1, 2, 4$ ($|\mathcal{V}| = 2{,}940$, training corpus $\approx 81{,}600$ tokens).

Order k	$\lvert\widehat{I}\rvert = \lvert\mathcal{V}\rvert^k$	**Observed states**	**Obs. / Total**	**Matrix non zero entries**
1	2,940	2,940	100.00%	0.3115%
2	8,643,600	26,922	0.3115%	0.000216%
4	7.47×10^{13}	69,603	$\approx 0\%$	$\approx 0\%$

Already at $k = 2$ the theoretical state space ($\approx$ 8.6 million entries) vastly exceeds the number of training tokens, so only 0.31% of bigram states are observed. At $k = 4$ the state space exceeds 7×10^{13}; the code automatically switches to a dictionary representation, and the 69,603 observed states are an infinitesimal fraction of the total. Crucially, even the matrix density—the fraction of *observed* transitions that are non-zero—collapses from 0.31% at $k = 1$ to effectively zero at $k = 4$: the model becomes increasingly deterministic on the contexts it has seen, and completely silent on all others.

Information-Theoretic Evaluation

Metrics. We report two complementary quantities. The *normalized test log-likelihood* ℓ_{test}/n is the negative cross-entropy of the test k-grams under the model; higher is better, and it equals $-H(X_1 \mid X_0)$ only when $\hat{P} = P$. Whenever $\hat{p}_{\mathbf{i}j} = 0$ for a test-observed pair $(\mathbf{i}, j)$, the contribution is $-\infty$; we therefore report separately the fraction of test k-grams affected and the log-likelihood restricted to the finite contributions. The *stationary-weighted KL divergence* $\widehat{D}_{\text{KL}}$ (equation (3.28)) measures the average discrepancy between the test-empirical distribution and the model; we again report only the finite part, i.e. restricted to states where $\hat{p}_{\mathbf{i}j} > 0$.

Table 3.4: Information-theoretic evaluation on the held-out test set (200 stories, $\approx$ 32,360 test k-grams in vocabulary).

Order k	**Zero-prob frac.**	ℓ_{test}/n **(finite)**	**States with** $D_{\text{KL}} = +\infty$	$\widehat{D}_{\text{KL}}$ **(finite)**
1	23.43%	-2.4927	1,402 / 1,578	0.0192
2	58.84%	-0.7130	11,701 / 13,150	0.0560
4	90.72%	-0.0600	27,814 / 28,967	0.0173

Interpretation. The three orders exhibit a clear and coherent pattern. As k increases, the fraction of test k-grams assigned probability zero grows monotonically: from 23% at $k = 1$ to 59% at $k = 2$ to 91% at $k = 4$. The finite part of the log-likelihood improves monotonically in the opposite direction ($-2.49 \to -0.71 \to -0.06$ nats/gram), reflecting the fact that a longer context makes the estimated conditional distribution nearly deterministic on the small set of transitions the model has actually observed. The $\widehat{D}_{\text{KL}}$ finite part does not vary monotonically ($0.019 \to 0.056 \to 0.017$): at $k = 4$ the states shared between training and test tend to be highly stereotyped phrases (e.g. `once upon a time`), on which training and test distributions agree closely, whereas at $k = 2$ the overlap is larger but the per-state estimates are noisier.

In summary, higher-order models face a fundamental tension: a longer context window yields sharper and more accurate predictions on seen transitions, but simultaneously renders an ever-growing fraction of test transitions invisible. No smoothing technique within the discrete Markov framework can overcome this, because the exponential barrier $|\mathcal{V}|^k \gg N$ guarantees that most k-gram contexts will never appear in any finite corpus as k grows.

Top-s prediction accuracy. In addition to likelihood-based metrics, we evaluate the predictive performance of the first-order model through a top-s next-token prediction task. For each context i observed in the test sequence, the model produces a ranking of possible next tokens according to the estimated

transition probabilities $\hat{p}_{ij}$. A prediction is considered correct in the top-s sense if the true next token appears among the s most probable transitions.

The evaluation was performed on 32,362 context–target pairs extracted from the test corpus using a corpus of 32,363 tokens. All contexts appearing in the test set were observed in the training corpus, so no transitions were skipped due to unseen states.

Metric	**Order 1**	**Order 2**	**Order 4**
Accuracy Metrics			
Top-1 accuracy	20.81%	20.17%	6.96%
Top-3 accuracy	34.20%	29.26%	8.62%
Top-5 accuracy	41.15%	32.93%	9.02%
Sparsity & Coverage			
Unseen Contexts (Skips)	0.00%	5.12%	**80.59%**
Evaluated Pairs	32,362	32,361	32,359
Sequence Baselines			
Most Frequent Seq. (k)	"the"	"a big"	"once upon a time"
Absolute Frequency	2,050	176	104
Sequence Percentage	6.33%	0.54%	0.32%
Global Baselines			
Random ($1/\|V\|$)	0.034%	0.034%	0.034%
Majority ("the")	6.33%	6.33%	6.34%

Table 3.5: Comparative analysis of Markov models (Orders 1, 2, and 4). The table illustrates the correlation between increasing model order, the explosion of sparsity (80.59% unseen contexts for $k = 4$), and the rarefaction of stereotyped sequences.

Interpretation. The results highlight the trade-off between model complexity and data sparsity in Markov language models. A simple first-order model already captures a substantial amount of local structure in the language: the top-1 accuracy reaches 20.81% , which is significantly higher than both the uniform random baseline (0.034%) and the majority-class predictor (6.33%). This indicates that transition probabilities estimated from the training corpus contain meaningful predictive information about the next token.

Increasing the model order does not necessarily improve predictive performance. The second-order model achieves a slightly lower top-1 accuracy (20.17%), while its top-3 and top-5 accuracies decrease to 29.26% and 32.93%, respectively. Although conditioning on longer contexts should in principle provide more information, the number of possible contexts grows rapidly and many of them are not observed in the training data.

This effect becomes particularly evident for the fourth-order model. In this case, 0.59% of the contexts appearing in the test set were never observed during training, leading to a drastic reduction in predictive performance (top-1 accuracy 6.96%). As a consequence, the model performs only marginally better than the majority baseline.

Overall, these results illustrate a classical phenomenon in n-gram language models: while higher-order models are theoretically more expressive, their practical performance strongly depends on the availability of sufficient training data. Without smoothing or back-off strategies, the sparsity of high-order contexts severely limits their predictive capability.

Towards kernel methods. The zero-probability problem has a single root cause: the discrete state space treats every k-gram context as an atomic symbol, with no notion of similarity between contexts. The contexts `[the cat sat on]` and `[a dog lay on]` are as different as `[the cat sat on]` and `[quantum foam collapsed into]`, even though the first two share obvious semantic and syntactic structure and should predict similar next tokens.

Kernel Ridge Regression addresses this directly. By embedding each word into $\mathbb{R}^d$ via pre-trained Word2Vec vectors and applying a kernel K on context representations, the model can *share statistical strength* across similar contexts: if $\mathbf{c}$ and $\mathbf{c}'$ are close in embedding space, their predictions will be similar even if $\mathbf{c}'$ was never observed in training. The continuous output space also guarantees that the predicted next-word embedding is always finite and well-defined, sidestepping the zero-probability problem entirely.

4 Kernel Methods Approach to Natural Language Processing

4.1 Problem Formulation

We consider the problem of predicting the next word in a sequence. Given a corpus of text, we construct a dataset of consecutive word pairs:

$$\mathcal{D} = \{(w_i, w_{i+1})\}_{i=1}^{n-1} \tag{4.1}$$

where $w_i \in \mathcal{V}$ is a word from a vocabulary $\mathcal{V}$ of size $|\mathcal{V}| = V$.

Representation Space

Each word is mapped to a vector representation through an embedding function:

$$E : \mathcal{V} \to \mathbb{R}^d \tag{4.2}$$

where d is the embedding dimension.

Training Data Matrices We construct the input and target matrices:

$$X = \begin{bmatrix} E(w_1)^T \\ E(w_2)^T \\ \vdots \\ E(w_{n-1})^T \end{bmatrix} \in \mathbb{R}^{(n-1)\times d} \tag{4.3}$$

$$Y = \begin{bmatrix} E(w_2)^T \\ E(w_3)^T \\ \vdots \\ E(w_n)^T \end{bmatrix} \in \mathbb{R}^{(n-1)\times d} \tag{4.4}$$

Objective: Estimating the Conditional Expectation

Rather than learning the function $f : \mathbb{R}^d \to \mathbb{R}^d$ itself, our goal is to estimate its *expected value* under the data-generating distribution. Specifically, we seek to approximate:

$$\mathbb{E}_\pi[E(w_{t+1}) \mid E(w_t) = x] \tag{4.5}$$

where π is the invariant distribution of the underlying stochastic process generating the language data.

This formulation is motivated by the fact that, in natural language, the next word is not deterministically determined by the current word, but follows a probability distribution. The conditional expectation:

$$f^*(x) = \mathbb{E}_\pi[E(w_{t+1}) \mid E(w_t) = x] = \sum_{w\in\mathcal{V}} E(w) \cdot \mathbb{P}(w_{t+1} = w \mid w_t) \tag{4.6}$$

represents the "average" next-word embedding given the current word.

Practical interpretation: The function f^* maps the embedding of word w_t to a weighted combination of all possible next-word embeddings, where the weights are the transition probabilities in the language model.

Our approach uses Kernel Ridge Regression to estimate f^* from the observed sequence of word pairs. As we will show in section 4.2, the ergodic theorem guarantees that this empirical estimate converges to the true conditional expectation under the invariant measure.

4.2 Theoretical Foundation: Language as a Markov Chain

The i.i.d. Assumption and Its Violation

Standard machine learning theory, including the analysis of Kernel Ridge Regression, relies on the assumption that training samples are *independent and identically distributed* (i.i.d.). Under this assumption, the law of large numbers guarantees that empirical averages converge to expectations, and generalization bounds can be derived.

However, natural language data fundamentally violates this assumption:

Sequential Dependence

Our training dataset consists of consecutive word pairs (w_i, w_{i+1}) extracted from text sequences. These pairs are *not independent*:

- The pairs (w_i, w_{i+1}) and (w_{i+1}, w_{i+2}) share the word w_{i+1}
- Words in a narrative are semantically and syntactically dependent
- The context at position i influences the context at position $i + 1$

More formally, for consecutive samples (x_i, y_i) and (x_{i+1}, y_{i+1}) where $x_i = E(w_i)$ and $y_i = E(w_{i+1})$:

$$\mathbb{P}[(x_{i+1}, y_{i+1})|(x_i, y_i)] \neq \mathbb{P}[(x_{i+1}, y_{i+1})] \tag{4.7}$$

Non-identical Distribution

The distribution of word pairs varies across:

- **Position in narrative**: Beginning, middle, and end of stories have different word distributions
- **Genre**: Different story types (fairy tales, realistic fiction) have different linguistic patterns
- **Topic**: The subject matter influences vocabulary and syntax

The fundamental question: Why should Kernel Ridge Regression generalize on such dependent, non-i.i.d. data?

Markov Chain Framework for Language

The key insight is to model natural language generation as a *first-order Markov chain* on the vocabulary $\mathcal{V}$.

First-Order Markov Model

For a sequence of words $w_1, w_2, \ldots, w_T$, we assume the Markov property:

$$\mathbb{P}(w_{t+1}|w_1, \ldots, w_t) = \mathbb{P}(w_{t+1}|w_t) \tag{4.8}$$

This defines a time-homogeneous Markov chain (following Definition 1.2) with:

- **State space**: $I = \mathcal{V}$ (the vocabulary)
- **Transition probabilities**: $p_{ij} = \mathbb{P}(w_{t+1} = j|w_t = i)$ for $i, j \in \mathcal{V}$
- **Transition matrix**: $P = (p_{ij})_{i,j\in\mathcal{V}}$

Interpretation: The sequence of words $(w_1, w_2, \ldots, w_T)$ is a trajectory of this Markov chain. Each transition from w_t to w_{t+1} follows the probability distribution $p_{w_t,\cdot}$.

Ergodicity of Language

For our theoretical framework to apply, we make the following assumption about the structure of natural language:

Assumption 1 (Ergodicity of Language)**.** *The Markov chain $(w_t)_{t\geq 0}$ on the vocabulary $\mathcal{V}$ is:*

1. ***Irreducible**: Every word can eventually lead to every other word (following Definition 1.8)*
2. ***Aperiodic**: The chain does not exhibit strict periodic behavior (following Definition 1.9)*
3. ***Positive recurrent**: The expected return time to any state is finite (following Definition 1.19)*

Justification

Irreducibility: In natural language, with sufficient context and creativity, any word can follow any other word (though with varying probabilities). For example, even uncommon transitions like "dog" $\rightarrow$ "quantum" are possible in appropriate contexts.

Aperiodicity: Language does not have deterministic cycles. Sentence boundaries, topic shifts, and the inherent variability in human expression prevent periodic behavior.

Positive recurrence: Common words like "the", "and", "is" recur frequently, and even less common words appear regularly in a large corpus, ensuring finite expected return times.

Invariant Distribution and Convergence

Under Assumption 1, Theorem 1.20 guarantees:

Theorem 4.1 (Existence and Uniqueness of Invariant Distribution)**.** *The Markov chain (w_t) admits a unique invariant probability distribution π on $\mathcal{V}$ such that:*

$$\pi_j = \sum_{i\in\mathcal{V}} \pi_i p_{ij} \quad \textit{for all } j \in \mathcal{V} \tag{4.9}$$

Moreover, for any initial distribution, the chain converges to π:

$$\lim_{t\to\infty} \mathbb{P}(w_t = j | w_0 = i) = \pi_j \tag{4.10}$$

Interpretation: The invariant distribution π represents the *long-run frequency* of words in natural language. It is the stationary distribution that the chain converges to regardless of where it starts.

The Ergodic Theorem

The central result that resolves the i.i.d. violation is the **Ergodic Theorem for Markov Chains** (Theorem 1.23):

Interpretation

This theorem provides the theoretical foundation for statistical learning on sequential data:

1. **Almost sure convergence**: The convergence holds with probability 1, not just in expectation
2. **Law of Large Numbers for dependent data**: The ergodic theorem is a generalization of the law of large numbers that applies to Markov chains

Remark 4.2 (Limits of the ergodic justification)**.** *The Ergodic Theorem guarantees the convergence of the empirical risk $\hat{L}(f) \to \mathbb{E}_\pi[\ell(w_{t+1}, f(w_t))]$ for any* fixed *function $f \in \mathcal{H}$. This is not sufficient to justify the generalization of the empirical minimizer $\hat{f}^*$, which depends on the training trajectory $(w_1, \ldots, w_n)$ and for which the empirical risk is systematically optimistic. A rigorous treatment would require additional arguments—based on the chain's mixing time or the algorithmic stability of KRR—which are beyond the scope of this work.*

4.3 Word Embedding Methods

The choice of embedding function $E : \mathcal{V} \to \mathbb{R}^d$ fundamentally determines the input representation for our model. We consider two embedding methods that represent different points in the trade-off between interpretability and expressiveness.

One-Hot Encoding

One-hot encoding is the simplest word representation, treating each word as an atomic unit with no inherent similarity to other words.

Definition. Let $\mathcal{V} = \{v_1, v_2, \ldots, v_V\}$ be the vocabulary of size V. The one-hot encoding of word v_i is the vector:

$$E(v_i) = e_i \in \mathbb{R}^V \tag{4.11}$$

where e_i is the i-th canonical basis vector:

$$e_i = (0, \ldots, 0, \underbrace{1}_{i\text{-th position}}, 0, \ldots, 0)^T \tag{4.12}$$

Properties:

- **Orthogonality:** $e_i^T e_j = \delta_{ij}$ (Kronecker delta), implying all words are equidistant
- **Sparsity:** Each vector has exactly one non-zero entry
- **Dimensionality:** $d = V$, which can be very large ($V \sim 10^4$ to 10^5 in practice)
- **No semantic structure:** The representation does not capture any notion of word similarity or meaning

Kernel implications: For the linear kernel:

$$K_{lin}(e_i, e_j) = e_i^T e_j = \delta_{ij} \tag{4.13}$$

This means the kernel only activates when words are identical, with no generalization across vocabulary items.

Advantages:

- Simple and deterministic
- No additional training required
- Exact word matching

Limitations:

- No notion of word similarity (e.g., "cat" and "dog" are as different as "cat" and "democracy")
- High dimensionality leads to curse of dimensionality
- Cannot handle out-of-vocabulary words

Word2Vec Embeddings

[Mikolov et al., 2013a]

Word2Vec is a family of neural network models that learn dense, low-dimensional word representations by exploiting the distributional hypothesis: words that appear in similar contexts tend to have similar meanings.

Overview. Word2Vec learns an embedding function:

$$E : \mathcal{V} \to \mathbb{R}^d \quad \text{with } d \ll V \tag{4.14}$$

where typically $d \in [50, 300]$, by training on the task of predicting words from their contexts (or vice versa).

Architecture variants: Word2Vec has two main architectures:

- **Continuous Bag-of-Words (CBOW):** Predicts a target word from its context
- **Skip-gram:** Predicts context words from a target word

We focus on the CBOW model, which is the variant used in our implementation.

CBOW Model Formulation

Given a text corpus represented as a sequence of words $w_1, w_2, \ldots, w_T$, the CBOW model aims to predict a target word w_t from its surrounding context words.

Context window: For a target word w_t, we define the context as the c words before and after it:

$$\text{Context}(w_t) = \{w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}\} \tag{4.15}$$

Model parameters: The model learns two sets of embeddings:

- **Input embeddings:** $W_{in} \in \mathbb{R}^{V \times d}$, where row i is the embedding $v_i \in \mathbb{R}^d$ of word v_i as input
- **Output embeddings:** $W_{out} \in \mathbb{R}^{d \times V}$, used for the prediction layer

Forward pass: Given a context $\{w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}\}$:

1. Extract context word embeddings: $v_{t-c}, \ldots, v_{t-1}, v_{t+1}, \ldots, v_{t+c}$
2. Compute context representation (average):

$$h = \frac{1}{2c} \sum_{j \in \{-c,\ldots,-1,1,\ldots,c\}} v_{t+j} \in \mathbb{R}^d \tag{4.16}$$

3. Compute scores for all words in vocabulary:

$$u = W_{out}^T h \in \mathbb{R}^V \tag{4.17}$$

4. Apply softmax to obtain probability distribution:

$$P(w_i|\text{Context}(w_t)) = \frac{\exp(u_i)}{\sum_{j=1}^{V} \exp(u_j)} = \text{softmax}(u)_i \tag{4.18}$$

Training Objective

The model is trained to maximize the log-likelihood of predicting target words from their contexts across the entire corpus of T words:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^{T} \log P(w_t|\text{Context}(w_t)) \tag{4.19}$$

Equivalently, we minimize the negative log-likelihood:

$$\mathcal{L}_{CBOW} = -\frac{1}{T} \sum_{t=1}^{T} \log P(w_t|\text{Context}(w_t)) \tag{4.20}$$

Expanding the softmax:

$$\mathcal{L}_{CBOW} = -\frac{1}{T} \sum_{t=1}^{T} \left[u_{w_t} - \log \sum_{j=1}^{V} \exp(u_j) \right] \tag{4.21}$$

where u_{w_t} denotes the score for the target word w_t.

Computational Challenges and Negative Sampling

The main computational bottleneck is the softmax normalization:

$$\sum_{j=1}^{V} \exp(u_j) \tag{4.22}$$

which requires $O(V)$ operations per training example. For large vocabularies ($V \sim 10^5$), this becomes prohibitive.

[Mikolov et al., 2013b]

Negative Sampling. Computing the softmax normalisation over the full vocabulary $V \sim 10^5$ at every training step costs $O(V)$, which is computationally prohibitive. Negative Sampling addresses this by replacing the softmax objective with a binary classification problem: for each target word w_t, a small set of m *negative* words $w_{\text{neg}}^{(1)}, \ldots, w_{\text{neg}}^{(m)}$ is sampled, and the model is trained to assign high score to the true word and low score to the negatives:

$$\log \sigma(u_{w_t}) + \sum_{i=1}^{m} \mathbb{E}\Big[\log \sigma\Big(-u_{w_{\text{neg}}^{(i)}}\Big)\Big], \tag{4.23}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the sigmoid function and $u_w = W_{\text{out}}^{\top} h$ is the score of word w. This reduces the per-step cost to $O(m)$ with $m \approx 5$–20.

The negative words are drawn from the unigram distribution raised to the power $3/4$:

$$P_n(w) \propto \text{freq}(w)^{3/4}. \tag{4.24}$$

This choice mediates between two unsatisfactory extremes: sampling proportionally to $\text{freq}(w)$ would cause very frequent tokens such as *the* or *a* to dominate almost every negative sample, providing little informative signal; uniform sampling, on the other hand, would oversample rare words that the model has barely encountered. The exponent $3/4$ compresses the frequency distribution, reducing the dominance of high-frequency words while still respecting the overall corpus statistics. Concretely, if $\text{freq}(\textit{the}) = 1000$ and $\text{freq}(\textit{unicorn}) = 1$, proportional sampling would select *the* a thousand times more often than *unicorn*, whereas under P_n the ratio drops to $1000^{3/4} \approx 178$. This choice is empirically motivated [Mikolov et al., 2013b] and has been shown to outperform both alternatives in practice.

Properties of Learned Embeddings

After training, the learned input embeddings $E(w) = v_w$ exhibit remarkable properties:

1. Cosine similarity captures semantic relatedness:

$$\text{sim}(w, w') = \frac{E(w)^T E(w')}{\|E(w)\| \, \|E(w')\|} \tag{4.25}$$

Words with similar meanings have high cosine similarity.

2. Linear structure in the embedding space: The embeddings capture semantic relationships through vector arithmetic. Famous examples include:

$$E(\text{"king"}) - E(\text{"man"}) + E(\text{"woman"}) \approx E(\text{"queen"}) \tag{4.26}$$

$$E(\text{"Paris"}) - E(\text{"France"}) + E(\text{"Italy"}) \approx E(\text{"Rome"}) \tag{4.27}$$

3. Dimensionality reduction: $d \ll V$ provides a compact representation while preserving semantic structure.

4. Continuous space: Unlike one-hot encoding, Word2Vec maps words into a continuous space, enabling interpolation and generalization.

Advantages for Kernel Ridge Regression

Using Word2Vec embeddings as input to our Kernel Ridge Regression model provides several benefits:

1. **Semantic smoothing:** Words with similar meanings have similar embeddings, allowing the kernel to generalize across similar contexts even if exact word sequences were not observed in training.

2. **Dimensionality reduction:** $d \ll V$ reduces the input dimension, making kernel computations more efficient and reducing the risk of overfitting.

3. **Handling rare words:** Word2Vec embeddings are learned from all contexts a word appears in, providing robust representations even for less frequent words.

4. **Continuous optimization:** The smooth embedding space enables gradient-based optimization and interpolation between word meanings.

5. **Transfer learning:** Pre-trained Word2Vec embeddings can be used, eliminating the need to train embeddings from scratch on small datasets.

Application to Kernel Ridge Regression

Training Data as Markov Trajectory

Our training dataset consists of N consecutive word pairs:

$$\mathcal{D} = \{(w_i, w_{i+1})\}_{i=1}^{N} \tag{4.28}$$

This is *not* a collection of i.i.d. samples. Rather, it is a *trajectory* of the Markov chain: a single realization of the stochastic process $(w_1, w_2, \ldots, w_N, w_{N+1})$.

After applying the embedding function:

$$\{(x_i, y_i)\}_{i=1}^{N} = \{(E(w_i), E(w_{i+1}))\}_{i=1}^{N} \tag{4.29}$$

The pairs (x_i, y_i) follow the trajectory of the embedded Markov chain.

Empirical Risk as Time Average

In Kernel Ridge Regression, we minimize the empirical risk:

$$R_{\text{emp}}(f) = \frac{1}{N}\sum_{i=1}^{N} \|y_i - f(x_i)\|^2 = \frac{1}{N}\sum_{i=1}^{N} \|E(w_{i+1}) - f(E(w_i))\|^2 \tag{4.30}$$

Key observation: This is a *time average* of the squared error function:

$$g(w_i, w_{i+1}) = \|E(w_{i+1}) - f(E(w_i))\|^2 \tag{4.31}$$

along the Markov trajectory $(w_1, \ldots, w_{N+1})$.

Convergence to Expected Risk

By the ergodic theorem (taking g to be the function above), as $N \to \infty$:

$$R_{\text{emp}}(f) = \frac{1}{N}\sum_{i=1}^{N} \|E(w_{i+1}) - f(E(w_i))\|^2 \xrightarrow{\text{a.s.}} \mathbb{E}_\pi \left[\|E(w_{t+1}) - f(E(w_t))\|^2\right] \tag{4.32}$$

where the expectation is taken with respect to the invariant distribution π of the Markov chain.

Interpretation: Even though our training samples are not i.i.d., the empirical risk converges almost surely to the expected risk under the invariant distribution. This is the *population risk* we aim to minimize.

Estimating the Conditional Expectation

Recall that our objective is to estimate:

$$f^*(x) = \mathbb{E}_\pi[E(w_{t+1}) \mid E(w_t) = x] \tag{4.33}$$

By the ergodic theorem, the Kernel Ridge Regression solution:

$$\hat{f}_N = \arg\min_{f \in \mathcal{H}} \left\{ \frac{1}{N}\sum_{i=1}^{N} \|E(w_{i+1}) - f(E(w_i))\|^2 + \lambda \|f\|_{\mathcal{H}}^2 \right\} \tag{4.34}$$

converges to the minimizer of the expected risk under π as $N \to \infty$, which is precisely the conditional expectation f^*.

Summary: The ergodic theorem guarantees that:

1. Kernel Ridge Regression learns an approximation to the conditional expectation $\mathbb{E}_\pi[E(w_{t+1}) \mid E(w_t)]$
2. Generalization is theoretically justified despite the violation of the i.i.d. assumption

Summary of Theoretical Foundation

We have established:

1. **Language is Markovian**: Natural language sequences can be modeled as a first-order Markov chain on the vocabulary
2. **Ergodicity holds**: Under reasonable assumptions (irreducibility, aperiodicity, positive recurrence), the chain admits a unique invariant distribution π
3. **KRR is justified**: The empirical risk in Kernel Ridge Regression converges to the expected risk under π

Computational Solution via Gradient Descent

Having established the theoretical foundation for applying Kernel Ridge Regression to sequential language data, we now address the practical computation of the solution.

Closed-Form Solution and Its Limitations

Recall that the Kernel Ridge Regression objective for scalar-valued outputs is:

$$\min_{\alpha \in \mathbb{R}^N} \frac{1}{N} \|y - K\alpha\|_2^2 + \lambda \alpha^T K \alpha \tag{4.35}$$

where $K \in \mathbb{R}^{N \times N}$ is the kernel matrix and $y \in \mathbb{R}^N$ is the vector of target values.

For vector-valued outputs $Y \in \mathbb{R}^{N \times D}$ (where D is the embedding dimension), we work with the coefficient matrix $A \in \mathbb{R}^{N \times D}$, and the objective becomes:

$$\min_{A \in \mathbb{R}^{N \times D}} \mathcal{L}(A) = \frac{1}{N} \|Y - KA\|_F^2 + \lambda \operatorname{tr}(A^T K A) \tag{4.36}$$

The closed-form solution is:

$$A^* = (K + N\lambda I_N)^{-1} Y \tag{4.37}$$

However, this direct approach has significant computational drawbacks:

- **Cubic complexity**: Computing the inverse via Cholesky decomposition requires $O(N^3)$ operations
- **Quadratic memory**: Storing the full kernel matrix K requires $O(N^2)$ memory
- **Numerical instability**: For large N or ill-conditioned K, direct inversion can be numerically unstable
- **No monitoring**: The black-box solution provides no insight into convergence or overfitting during optimization

For large-scale applications, these limitations motivate an iterative approach.

Gradient Descent Formulation

Gradient Computation

To apply gradient descent, we compute the gradient of the objective (4.36) with respect to A.

Expanding the data term:

$$\frac{1}{N} \|Y - KA\|_F^2 = \frac{1}{N} \operatorname{tr}\left[(Y - KA)^T (Y - KA)\right] \tag{4.38}$$

Using matrix calculus identities:

$$\frac{\partial}{\partial A} \left[\frac{1}{N} \|Y - KA\|_F^2 \right] = -\frac{2}{N} K^T (Y - KA) \tag{4.39}$$

$$\frac{\partial}{\partial A} \left[\lambda \operatorname{tr}(A^T K A) \right] = 2\lambda K A \tag{4.40}$$

Therefore, the gradient of $\mathcal{L}(A)$ is:

$$\nabla_A \mathcal{L}(A) = -\frac{2}{N} K^T (Y - KA) + 2\lambda KA \tag{4.41}$$

This can be equivalently written as:

$$\nabla_A \mathcal{L}(A) = -\frac{2}{N} K^T Y + \frac{2}{N} K^T KA + 2\lambda KA \tag{4.42}$$

Update Rule
The gradient descent update at iteration t is:

$$A^{(t+1)} = A^{(t)} - \eta \nabla_A \mathcal{L}(A^{(t)}) \tag{4.43}$$

where $\eta > 0$ is the learning rate.
Substituting (4.41):

$$A^{(t+1)} = A^{(t)} + \frac{2\eta}{N} K^T (Y - KA^{(t)}) - 2\eta\lambda KA^{(t)} \tag{4.44}$$

Advantages of Gradient Descent

The iterative approach offers several benefits:

1. **Continuous monitoring**: We can track training and validation loss at each epoch, enabling detection of overfitting or convergence issues
2. **Early stopping**: By monitoring validation performance, we can halt training when generalization begins to degrade:

$$\text{Stop if } \mathcal{L}_{\text{val}}(A^{(t)}) > \mathcal{L}_{\text{val}}(A^{(t-1)}) \text{ for } k \text{ consecutive iterations} \tag{4.45}$$

3. **Regularization control**: The interplay between learning rate η and regularization λ can be tuned adaptively
4. **Scalability**: For very large N, mini-batch or stochastic variants can be employed
5. **Numerical stability**: Small step sizes avoid the numerical instability of direct matrix inversion

Convergence Analysis

Under mild conditions, gradient descent converges to the global minimum of (4.36).

Theorem 4.3 (Convergence of Gradient Descent for KRR)**.** *Let K be positive semidefinite with $\lambda > 0$. Then for sufficiently small learning rate η, the gradient descent iterates* (4.43) *converge to the closed-form solution:*

$$\lim_{t \to \infty} A^{(t)} = A^* = (K + N\lambda I_N)^{-1} Y \tag{4.46}$$

Proof sketch. The objective $\mathcal{L}(A)$ is strongly convex in A since:

$$\nabla_A^2 \mathcal{L}(A) = \frac{2}{N} K^T K + 2\lambda K \succeq 2\lambda K \succ 0 \tag{4.47}$$

when K is positive definite (guaranteed by $\lambda > 0$). Strong convexity ensures that gradient descent with appropriate step size converges linearly to the unique global minimum. □

Learning Rate Selection

To determine the appropriate learning rate for gradient descent, we analyze the Hessian of the objective function and derive a theoretical bound.

Hessian and Lipschitz constant. Consider the Kernel Ridge Regression objective:

$$\mathcal{L}(A) = \frac{1}{N}\|KA - Y\|_F^2 + \lambda \operatorname{tr}(A^T K A) \tag{4.48}$$

Since the loss is quadratic in A, its Hessian is constant and given by:

$$\nabla_A^2 \mathcal{L}(A) = \frac{2}{N} K^T K + 2\lambda K \tag{4.49}$$

Because the kernel matrix K is symmetric positive semidefinite, we have $K^T = K$ and therefore:

$$\nabla_A^2 \mathcal{L}(A) = 2\left(\frac{1}{N} K^2 + \lambda K\right) \tag{4.50}$$

Spectral bound. Let μ_i denote the eigenvalues of K. The eigenvalues of the Hessian are then:

$$\lambda_i\big(\nabla_A^2 \mathcal{L}\big) = 2\left(\frac{1}{N}\mu_i^2 + \lambda \mu_i\right) \tag{4.51}$$

The Lipschitz constant of the gradient is given by the largest eigenvalue:

$$L = \lambda_{\max}\big(\nabla_A^2 \mathcal{L}\big) = 2\left(\frac{1}{N}\|K\|_2^2 + \lambda \|K\|_2\right) \tag{4.52}$$

where $\|K\|_2$ denotes the spectral norm (largest eigenvalue) of K.

Learning rate bound. For gradient descent applied to a function with L-Lipschitz gradient, the step size must satisfy:

$$0 < \eta < \frac{2}{L} \tag{4.53}$$

Substituting the expression for L, we obtain:

$$0 < \eta < \frac{2}{2\left(\frac{1}{N}\|K\|_2^2 + \lambda\|K\|_2\right)} = \frac{1}{\frac{1}{N}\|K\|_2^2 + \lambda\|K\|_2} \tag{4.54}$$

Multiplying numerator and denominator by N yields:

$$\boxed{0 < \eta < \frac{N}{\|K\|_2^2 + N\lambda\|K\|_2}} \tag{4.55}$$

This theoretical bound ensures convergence of gradient descent to the global minimum.

Implementation Algorithm

Algorithm 1 Gradient Descent for Kernel Ridge Regression

Require: Training data $(X_{\text{train}}, Y_{\text{train}})$, validation data $(X_{\text{val}}, Y_{\text{val}})$
Require: Kernel type, regularization λ, learning rate η, max epochs T, patience k
Compute kernel matrices: $K_{\text{train}} = \text{Kernel}(X_{\text{train}}, X_{\text{train}})$, $K_{\text{val}} = \text{Kernel}(X_{\text{val}}, X_{\text{train}})$
Initialize: $A^{(0)} \sim \mathcal{N}(0, 0.01^2 I)$
best_loss $= \infty$, patience_counter $= 0$
for $t = 0, 1, \ldots, T-1$ **do**
  Compute gradient: $\nabla = -\frac{2}{N} K_{\text{train}}^T (Y_{\text{train}} - K_{\text{train}} A^{(t)}) + 2\lambda K_{\text{train}} A^{(t)}$
  Update: $A^{(t+1)} = A^{(t)} - \eta \nabla$
  Compute square losses:
    $\mathcal{L}_{\text{train}}(A^{(t+1)}) = \frac{1}{N} \|Y_{\text{train}} - K_{\text{train}} A^{(t+1)}\|^2$
    $\mathcal{L}_{\text{val}}(A^{(t+1)}) = \frac{1}{N_{\text{val}}} \|Y_{\text{val}} - K_{\text{val}} A^{(t+1)}\|^2$
  if $\mathcal{L}_{\text{val}}(A^{(t+1)}) <$ best_loss **then**
    best_loss $= \mathcal{L}_{\text{val}}(A^{(t+1)})$, $A_{\text{best}} = A^{(t+1)}$, patience_counter $= 0$
  else
    patience_counter $=$ patience_counter $+ 1$
  end if
  if patience_counter $\geq k$ **then**
    break ▷ Early stopping
  end if
end for
return A_{best}

Decoding: From Prediction to Token

The output of KRR is a vector $\hat{y} = f(\mathbf{c}_t) \in \mathbb{R}^d$, which is a continuous point in the embedding space. Since $\hat{y}$ does not correspond in general to any word in the vocabulary $\mathcal{V}$, a *decoding step* is required to map it back to a discrete token. The decoding procedure depends on the choice of encoding.

One-hot encoding. When $E(w_i) = e_i \in \mathbb{R}^{|\mathcal{V}|}$ is the i-th standard basis vector, the embedding matrix $\mathcal{E} = I_{|\mathcal{V}|}$ is the identity. The KRR output $\hat{y} \in \mathbb{R}^{|\mathcal{V}|}$ can therefore be interpreted directly as a score vector over the vocabulary, and decoding reduces to taking the argmax:

$$\hat{w}_{t+1} = \arg\max_{w_i \in \mathcal{V}} \hat{y}_i.$$

No geometric search is needed: the i-th component of $\hat{y}$ is simply the score assigned to word w_i.

Word2Vec encoding. When $E(w_i) \in \mathbb{R}^d$ is a dense embedding with $d \ll |\mathcal{V}|$, the prediction $\hat{y} \in \mathbb{R}^d$ is a point in a continuous, low-dimensional space and cannot be interpreted componentwise as a score over the vocabulary. Decoding requires finding the vocabulary word whose embedding is closest to $\hat{y}$.

Let $\mathcal{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ be the embedding matrix, where the i-th row $\mathcal{E}[i] = E(w_i)$. Since all Word2Vec embeddings are ℓ^2-normalised after training ($\|E(w_i)\| = 1$ for all $w_i \in \mathcal{V}$), cosine similarity reduces to the standard inner product:

$$\text{sim}(\hat{y},\, E(w_i)) = \frac{\hat{y}^\top E(w_i)}{\|\hat{y}\|\, \|E(w_i)\|} = \frac{\hat{y}^\top E(w_i)}{\|\hat{y}\|}.$$

The cosine similarity scores $s_i = \text{sim}(\hat{y}, E(w_i))$ thus constitute a natural ranking of the vocabulary with respect to $\hat{y}$.

Greedy decoding and its limitations. The simplest decoding strategy is *greedy decoding*, which always selects the word with the highest similarity score:

$$\hat{w}_{t+1} = \arg\max_{w \in \mathcal{V}} \mathcal{E}\, \frac{\hat{y}}{\|\hat{y}\|},$$

computed as a single matrix-vector product $\mathcal{E}\,\tilde{y}$, where $\tilde{y} = \hat{y}/\|\hat{y}\|$, at cost $O(|\mathcal{V}| \cdot d)$. While computationally efficient, greedy decoding has a well-known drawback: by always picking the single most likely token, it tends to produce repetitive and degenerate sequences, since the model repeatedly falls into the same high-probability regions of the embedding space.

Sampling-based decoding. A more flexible alternative is to treat the similarity scores as logits of a probability distribution over the vocabulary and *sample* from it rather than taking the argmax. Concretely, given the scores $s_i = \tilde{y}^\top E(w_i)$, one first optionally restricts attention to the K highest-scoring words (*top-K sampling*), and then defines a distribution via a softmax with temperature $\tau > 0$:

$$P_\tau(w_i \mid \hat{y}) = \frac{\exp(s_i/\tau)}{\sum_{j \in \mathcal{K}} \exp(s_j/\tau)},$$

where $\mathcal{K}$ is the set of top-K candidates (or the full vocabulary if $K = |\mathcal{V}|$). The next token is then drawn from this distribution:

$$\hat{w}_{t+1} \sim P_\tau(\,\cdot \mid \hat{y}).$$

The temperature τ controls the sharpness of the distribution:

- $\tau \to 0$: the distribution concentrates on the argmax, recovering greedy decoding
- $\tau = 1$: raw softmax over the cosine scores
- $\tau > 1$: flatter distribution, more diversity at the cost of lower average quality

The top-K restriction prevents sampling from the long tail of words with near-zero similarity to $\hat{y}$, which would otherwise introduce incoherent tokens regardless of temperature. In our experiments we use $\tau = 0.8$ and $K = 50$ as default values, which produce more varied and natural-sounding sequences than greedy decoding while avoiding the degeneration caused by sampling from the full vocabulary.

Remark 4.4. *In the one-hot setting, sampling-based decoding is also applicable: since $\hat{y} \in \mathbb{R}^{|\mathcal{V}|}$ is already a score vector over the vocabulary, one can directly apply a softmax with temperature τ to obtain a distribution and sample from it, without any nearest-neighbour search. The scores $\hat{y}_i$ are real-valued outputs of KRR and lack the geometric interpretation of cosine similarities, but the sampling procedure is otherwise identical to the Word2Vec case.*

4.4 Extension to n-gram Contexts

Motivation

The formulation presented so far relies on *bigram* conditioning: the prediction of w_{t+1} depends solely on the immediately preceding word w_t. This corresponds to a first-order Markov assumption and has well-known limitations in capturing long-range syntactic and semantic dependencies. Natural language exhibits dependencies that extend over several tokens—agreement phenomena, coreference, idiomatic expressions, and topical coherence all suggest that richer contexts improve predictive quality [Jurafsky and Martin, 2025].

The n-gram extension addresses this by conditioning on the $n-1$ preceding words:

$$\mathbb{P}(w_{t+1} \mid w_1, \ldots, w_t) \approx \mathbb{P}(w_{t+1} \mid w_{t-n+2}, \ldots, w_t), \tag{4.56}$$

which defines an $(n-1)$-th order Markov chain. The kernel regression framework must then be extended to accommodate vector-valued context representations built from a *window* of embeddings rather than a single embedding.

Problem Reformulation

Given a corpus $w_1, w_2, \ldots, w_T$, we now construct the dataset of n-gram pairs:

$$\mathcal{D}_n = \big\{(\mathbf{c}_t,\, E(w_{t+1}))\big\}_{t=n-1}^{T-1}, \tag{4.57}$$

where $\mathbf{c}_t$ is a fixed-dimensional *context representation* derived from the window $\big(w_{t-n+2}, \ldots, w_t\big) \in \mathcal{V}^{n-1}$.

The regression target remains the embedding of the next word, and the goal is to estimate:

$$f^*(x) = \mathbb{E}_\pi\big[E(w_{t+1}) \mid \mathbf{c}_t = x\big], \tag{4.58}$$

where π is now the invariant distribution of the $(n-1)$-th order chain, which can be reduced to a first-order chain on the extended state space $\mathcal{V}^{n-1}$.

The theoretical justification via the ergodic theorem carries over unchanged, since an $(n-1)$-th order Markov chain on $\mathcal{V}$ is equivalent to a first-order chain on $\mathcal{V}^{n-1}$, and ergodicity of the original chain implies ergodicity of the lifted chain

Context Aggregation Strategies

The central design choice is the mapping:

$$\phi : \mathcal{V}^{n-1} \longrightarrow \mathbb{R}^{d'}, \quad \phi(w_{t-n+2}, \ldots, w_t) = \mathbf{c}_t, \tag{4.59}$$

which compresses the context window into a single vector $\mathbf{c}_t \in \mathbb{R}^{d'}$ that is then fed as input to the kernel. We now survey the principal choices.

1 Uniform Weighted Average (Arithmetic Mean)

Definition.

$$\mathbf{c}_t^{\text{avg}} = \frac{1}{n-1} \sum_{k=1}^{n-1} E(w_{t-k+1}) \in \mathbb{R}^d. \tag{4.60}$$

This is the approach adopted in our implementation.

Advantages.

- **Fixed output dimension**: $d' = d$ regardless of n, keeping the kernel matrix size independent of context length.
- **Permutation invariance**: The representation is invariant to the order of context words, which is desirable for bag-of-words style kernels.
- **Efficiency**: Requires $O(n \cdot d)$ operations per sample, which is linear in n.
- **Smoothness**: Averaging reduces variance in the representation, making the kernel matrix better conditioned.

Disadvantages.

- **Loss of positional information**: The relative order of words in the window is entirely discarded. The contexts "cat chased the" and "the chased cat" yield identical representations.
- **Equal contribution of distant words**: Words far from the target contribute equally to those immediately preceding it, which contradicts the empirical observation that proximal words are more predictive.

2 Exponentially Decaying Weighted Average

Definition. Assign geometrically decreasing weights to words further from the target:

$$\mathbf{c}_t^{\text{exp}} = \frac{\sum_{k=1}^{n-1} \gamma^{k-1} E(w_{t-k+1})}{\sum_{k=1}^{n-1} \gamma^{k-1}} \in \mathbb{R}^d, \quad \gamma \in (0,1), \tag{4.61}$$

so that w_t (the most recent word, $k=1$) receives weight 1, w_{t-1} receives weight γ, and so on.

Advantages.

- **Recency bias**: The most recent word contributes most, consistent with the Markovian structure of language.
- **Fixed dimension**: As with the uniform average, $d' = d$.
- **Soft windowing**: By varying γ, one interpolates between a bigram model ($\gamma \to 0$) and a uniform average ($\gamma \to 1$).

Disadvantages.

- **Additional hyperparameter**: The decay rate γ must be tuned, adding to the cross-validation budget alongside λ and the kernel parameters.
- **Positional information still lost**: Although magnitudes differ, the relative order of words is not explicitly encoded.

3. Concatenation

Definition. Stack all context embeddings into a single vector:

$$\mathbf{c}_t^{\text{cat}} = \left[E(w_{t-n+2})^T \,\|\, E(w_{t-n+3})^T \,\|\, \cdots \,\|\, E(w_t)^T\right]^T \in \mathbb{R}^{(n-1)d}, \tag{4.62}$$

where $\|$ denotes vector concatenation.

Advantages.

- **Full positional information**: Each position in the window is encoded in a distinct block of the concatenated vector, so word order is completely preserved.
- **No information loss**: Concatenation is an injective map; no information is discarded by aggregation.
- **Expressive kernel**: A kernel on $\mathbb{R}^{(n-1)d}$ can capture interactions *between* positions that averaging cannot.

Disadvantages.

- **Dimension growth**: The input dimension grows as $(n-1)d$, which linearly inflates the cost of kernel evaluations and exacerbates the curse of dimensionality for large n or d.
- **Fixed context length**: The model is tied to a fixed n; sequences shorter than $n-1$ require padding with a special token, introducing an artificial distributional artifact.
- **Positional rigidity**: A phrase shifted by one position within the window produces an entirely different representation, reducing generalization across positions.

Formal Comparison

Table 4.1 summarises the properties of the aggregation strategies discussed above.

Table 4.1: Comparison of context aggregation strategies for n-gram Kernel Ridge Regression.

Method	Output dim	Order info	Hyperparams	Convexity
Uniform avg (4.60)	d	No	None	Yes
Exponential avg (4.61)	d	Partial	γ	Yes
Concatenation (4.62)	$(n-1)d$	Yes	None	Yes

Updated Training Matrices

For any fixed aggregation ϕ, the input matrix is:

$$X_n = \begin{bmatrix} \phi(w_1, \ldots, w_{n-1})^T \\ \phi(w_2, \ldots, w_n)^T \\ \vdots \\ \phi(w_{T-n+1}, \ldots, w_{T-1})^T \end{bmatrix} \in \mathbb{R}^{(T-n+1)\times d'}, \tag{4.63}$$

while the target matrix remains:

$$Y_n = \begin{bmatrix} E(w_n)^T \\ E(w_{n+1})^T \\ \vdots \\ E(w_T)^T \end{bmatrix} \in \mathbb{R}^{(T-n+1)\times d}. \tag{4.64}$$

The kernel matrix $K \in \mathbb{R}^{(T-n+1)\times(T-n+1)}$ is then computed from the rows of X_n using the chosen kernel function, and the remainder of the KRR algorithm (Section 4) proceeds unchanged.

Remark on the uniform average kernel. When the linear kernel $K_{\text{lin}}(\mathbf{c}, \mathbf{c}') = \mathbf{c}^T\mathbf{c}'$ is used with the uniform average (4.60), the kernel evaluates to:

$$K_{\text{lin}}\big(\mathbf{c}_t^{\text{avg}},\, \mathbf{c}_s^{\text{avg}}\big) = \frac{1}{(n-1)^2} \sum_{k=1}^{n-1} \sum_{j=1}^{n-1} E(w_{t-k+1})^T E(w_{s-j+1}), \tag{4.65}$$

which decomposes into a sum of pairwise inner products between all pairs of context words from the two windows—a form of unordered n-gram interaction kernel.

4.5 Experimental Results

Experimental Setup

All experiments use the same *TinyStories* corpus [Eldan and Li, 2023] as Chapter 3, but with a strictly separated data pipeline to avoid leakage between the embedding training, the KRR training, and the evaluation.

Data pipeline. Word2Vec embeddings are trained on a large corpus of 10,000 stories. The KRR model is then trained on a disjoint set of 500 stories (stories 0–499), capped at 20,000 samples to keep the Kernel matrix $K \in \mathbb{R}^{N\times N}$ computationally tractable (≈ 3.2 GB in `float32`). The validation set is drawn from stories 1,000–1,099 (100 stories), and the test set from stories 600–799 (200 stories), ensuring no story appears in more than one split.

Preprocessing and vocabulary. Text is lowercased and stripped of non-alphabetic characters. The vocabulary is built from the KRR training stories with a minimum frequency threshold of 3; words present in the training corpus but absent from the Word2Vec model are discarded. Table 4.2 summarises the resulting dataset sizes.

Table 4.2: Dataset statistics for the KRR experiments. Sample counts vary across n-gram orders because contexts requiring n consecutive in-vocabulary words become increasingly rare for larger n.

Quantity	**Bigram**	**Trigram**	**Pentagram**
Stories for Word2Vec		10,000	
Stories for KRR training		500 (stories 0–499)	
Stories for KRR validation		100 (stories 1,000–1,099)	
Stories for KRR test		200 (stories 600–799)	
Vocabulary size $\lvert\mathcal{V}\rvert$		487	
Word2Vec embedding dimension d		100	
Training samples (after cap)	20,000	20,000	20,000
Validation samples	12,322	10,435	7,560
Test samples	23,642	20,098	14,733

Model configurations. We evaluate three context sizes corresponding to the n-gram orders introduced in Section 4.4:

- **Bigram** ($n = 1$): the context is the single preceding word w_t; the input to KRR is $\mathbf{c}_t = E(w_t) \in \mathbb{R}^{100}$.
- **Trigram** ($n = 2$): the context is (w_{t-1}, w_t); the input is the uniform average $\mathbf{c}_t = \frac{1}{2}(E(w_{t-1}) + E(w_t)) \in \mathbb{R}^{100}$.
- **Pentagram** ($n = 4$): the context is $(w_{t-3}, w_{t-2}, w_{t-1}, w_t)$; the input is the uniform average of 4 embeddings $\mathbf{c}_t \in \mathbb{R}^{100}$.

In all cases we use the **linear kernel** $K(\mathbf{c}, \mathbf{c}') = \mathbf{c}^T\mathbf{c}'$ and the **MSE loss**. The regularisation parameter is $\lambda = 0.05$. The learning rate is set to $0.9\,\eta_{\max}$ where

$$\eta_{\max} = \frac{N}{\|K\|_2^2 + N\lambda\|K\|_2} \tag{4.66}$$

is the spectral bound derived in Section 4.3. Early stopping monitors the validation loss with patience 200.

Gradient Descent: Loss Curves

Figure 4.1 show the training and validation **MSE loss** curves over the gradient descent iterations for each of the three models.

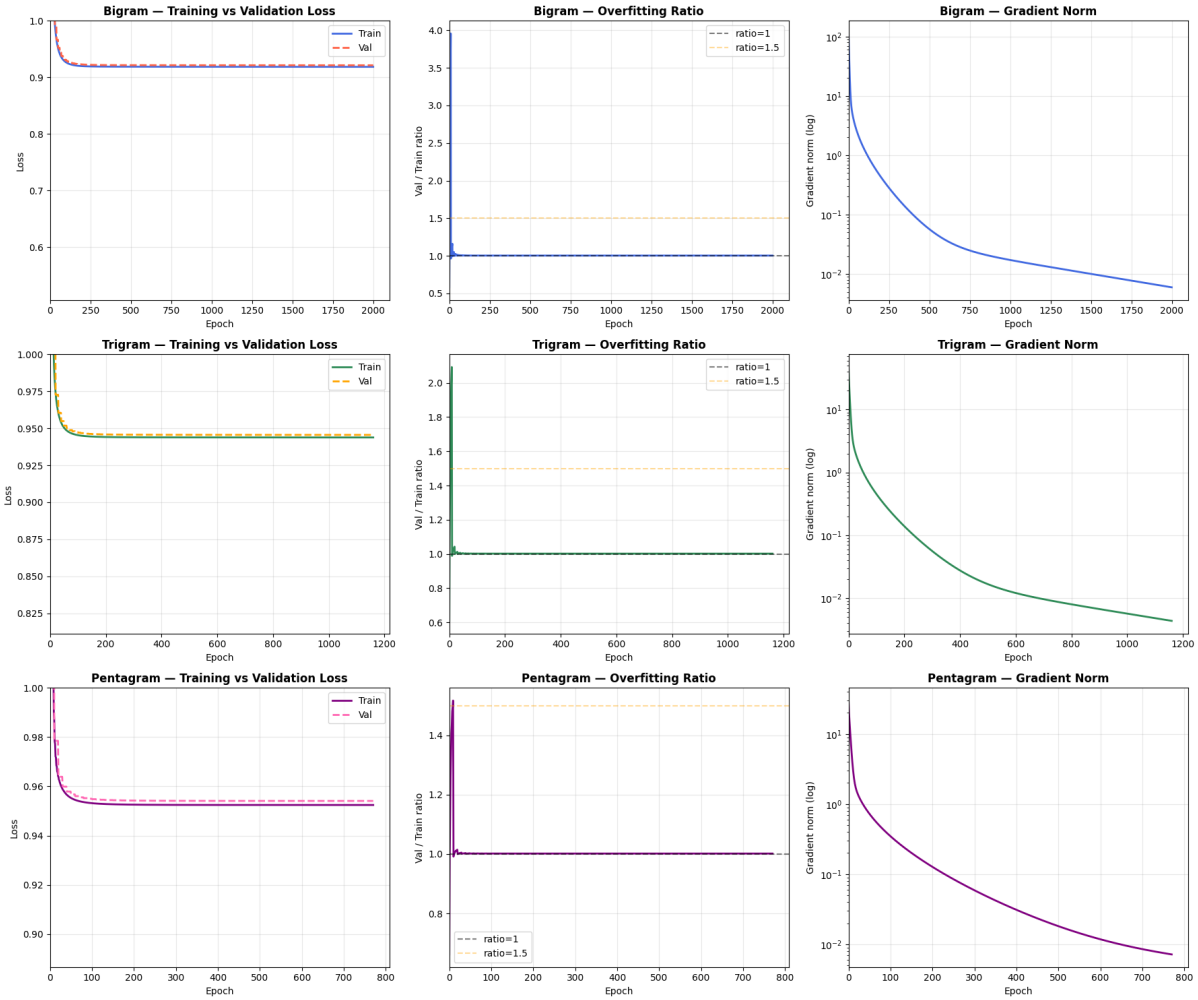

Figure 4.1: Training and validation MSE loss curves for the **Bigram**, **Trigram**, and **Pentagram** KRR models.

Table 4.3 records the key convergence statistics for all three models.

Table 4.3: Convergence statistics for the three KRR models trained with **MSE loss**. "Epochs" is the number of gradient descent steps actually run before early stopping.

Model	Epochs	Train loss	Val loss (best)	Grad norm (final)
Bigram	1,960	0.918615	0.921367	0.005947
Trigram	1,160	0.943840	0.945558	0.004364
Pentagram	770	0.952423	0.954079	0.007168

Top-s Prediction Accuracy

We evaluate the predictive performance of all three models using the **top-s accuracy** (as defined in Chapter 2). Tables 4.4 and 4.5 report the results on the test set and on the training and validation sets respectively, together with the two baselines.

Table 4.4: Top-s next-token prediction accuracy on the **test set** for the three KRR models. Baselines are the same for all orders.

Metric	**Bigram** (23,642)	**Trigram** (20,098)	**Pentagram** (14,733)
Top-1	18.67%	12.81%	7.93%
Top-3	29.25%	21.35%	15.76%
Top-5	34.57%	26.38%	19.75%
Random ($1/\|\mathcal{V}\| = 487$)		0.21%	
Majority ("the")		7.11%	

Table 4.5: Top-s accuracy on the **training** and **validation** sets.

	Bigram		**Trigram**		**Pentagram**	
Metric	Train	Val	Train	Val	Train	Val
Top-1	18.04%	17.59%	12.26%	12.03%	8.13%	7.86%
Top-3	28.17%	28.14%	20.68%	19.78%	15.54%	15.13%
Top-5	33.52%	33.33%	25.92%	24.90%	19.32%	18.86%

Bigram. The bigram model is evaluated on 23,642 (context, target) pairs extracted from the test corpus. It achieves a top-1 accuracy of 18.67% and a top-5 accuracy of 34.57%, substantially outperforming both the random baseline (0.21%) and the majority-class predictor (7.11%). The close agreement between training (18.04%), validation (17.59%) and test (18.67%) top-1 accuracies indicates that the model generalises well without overfitting.

Trigram. The trigram model achieves a top-1 accuracy of 12.81% and a top-5 accuracy of 26.38% on the test set. Performance is consistently lower than the bigram model across all values of s, suggesting that the uniform-average aggregation of two embeddings introduces a loss of discriminative information that outweighs the benefit of the wider context window.

Pentagram. The pentagram model achieves the lowest accuracy across all metrics, with a top-1 of 7.93% and a top-5 of 19.75%. The monotonic degradation from bigram to pentagram is consistent with the hypothesis that uniform averaging over longer contexts progressively smooths out the embedding signal, pushing the context vector towards the mean of the embedding space and reducing its predictive content.

Discussion. The results reveal a clear trade-off: richer context does not translate into higher accuracy under uniform-average aggregation. The bigram model, which feeds the raw embedding of the preceding word directly to KRR, retains the full information of that embedding, whereas the trigram and pentagram models compress multiple words into a single vector of the same dimension $d = 100$. This compression is lossy: the linear kernel $K(\mathbf{c}, \mathbf{c}') = \mathbf{c}^T\mathbf{c}'$ cannot recover the individual contributions of each context word from their average. A natural direction for improvement would be to concatenate context embeddings rather than average them, at the cost of a higher-dimensional input space, or to adopt a weighted aggregation scheme that assigns greater importance to the most recent word.

Python code

Bibliography

Mauricio A. Alvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: a review, 2012. URL https://arxiv.org/abs/1106.6251.

Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English? *arXiv preprint arXiv:2305.07759*, 2023. Dataset utilizzato per gli esperimenti.

F. Fagnola and E. Sasso. Catene di markov discrete, 2017.

Dan Jurafsky and James H. Martin. Speech and language processing (3rd ed. draft), 2025. URL https://web.stanford.edu/~jurafsky/slp3/. Online manuscript released August 24, 2025.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations (ICLR)*, 2013a. Paper originale di Word2Vec (CBOW e Skip-gram).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119, 2013b. Introduce negative sampling e altre ottimizzazioni.

L. Rosasco. Statistical learning theory, 2023. URL https://www.dropbox.com/scl/fi/fej5t9ya6qn0yex9rt8kw/ScribesSLT.pdf?e=1&rlkey=68yw874ybgtkn1x8m18mbnvfw&dl=0.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

Wikipedia contributors. Kullback–leibler divergence — Wikipedia, the free encyclopedia, 2026. URL https://en.wikipedia.org/w/index.php?title=Kullback%E2%80%93Leibler_divergence&oldid=1340419888. [Online; accessed 13-March-2026].

Jianhong Xu. Can a higher order markov chain be treated as a first order markov chain?, 2025. URL https://arxiv.org/abs/2512.01969.