Spherical Codes with Fourier-Weighted Neural Networks for Mini-Scale Text Generation

Orges Leka & ChatGPT 5
(Limburg, Deutschland)
firstname.lastname@gmail.com

August 28, 2025

Abstract

We investigate the combination of maximally separated spherical codes with Fourier-Weighted Neural Networks (FWNNs) for compact text generation models. The key observation is that the cosine similarity between concatenated unit-norm code blocks is a tightly controlled affine proxy for Hamming similarity. This enables the definition of a "soft Hamming" error, whose sensitivity to single-token flips is transparent and whose concentration improves at rate $O(1/\sqrt{w})$. Embedding this surrogate into a softmax classifier yields a cross-entropy loss that is both geometrically interpretable and provably consistent: under mild identifiability and mixing assumptions, the expected loss converges as window length $w \to \infty$, and classification error vanishes for large scale parameters. FWNNs complement this framework by parameterizing weight matrices spectrally, reducing complexity while maintaining standard backpropagation. The resulting architecture thus combines (i) geometric calibration via spherical codes, (ii) sample-efficient, stable optimization via FWNNs, and (iii) a principled link between cosine similarity and discrete Hamming error. We provide both theoretical guarantees and illustrative constructions, positioning the method as a lightweight yet mathematically grounded tool for symbolic sequence modeling.

Contents

1 Introduction

Designing compact sequence models that are both *predictive* and *analytically tractable* remains a central challenge in modern machine learning. This paper studies a simple geometric recipe for symbolic sequences: map tokens to a *maximally separated spherical code*, concatenate w code vectors for a context window, ℓ_2-normalize the result, and score classes by a function of the cosine similarity to class references. The key observation is elementary yet powerful: because the blocks in the concatenation are disjoint, the cosine between two normalized windows equals the *average* of their per-position cosines. For well-separated codes (small coherence μ), this average provides an *affine, tightly controlled estimator* of position-wise Hamming similarity. In other words, the geometry on the sphere mirrors the discrete Hamming world, but with smoothness and calibrated gradients.

Building on this link, we introduce a *soft Hamming* similarity $\widehat{\alpha}$ by linearly rescaling the window cosine and clipping to $[0, 1]$. A single token flip changes the score by approximately $(1 - \mu)/w$, making the sensitivity transparent; and by Hoeffding-type concentration, the window cosine concentrates around its mean at rate $\mathcal{O}(1/\sqrt{w})$. These properties yield losses that carry a direct discrete meaning ("fraction of matches") while remaining differentiable end-to-end.

On the modeling side, we adopt a *Fourier-Weighted Neural Network* (FWNN) backbone, in which dense linear maps are parameterized by a small number of spectral coefficients rather than free entries. FWNNs keep standard backpropagation intact, but reduce parameter count and memory footprint, offering a practical path to small models without losing the geometric structure induced by the spherical code embedding.

Contributions.

- **Cosine ↔ Hamming calibration.** We formalize tight bounds that convert cosine between unit-normalized, block-concatenated code windows into estimates of position-wise Hamming similarity, with deviation controlled by code coherence μ and window length w.
- **A soft Hamming loss.** We define a cosine-calibrated soft Hamming similarity $\widehat{\alpha}$ and use it to build pairwise margin objectives and multiclass softmax scores. The resulting loss has an interpretable flip sensitivity $\approx (1 - \mu)/w$ and smooth gradients on the sphere.
- **Consistency for long windows.** Under mild identifiability (a fixed mean gap in expected Hamming similarity between the true class and competitors), boundedness, and weak dependence across positions, we prove that the expected cross-entropy converges as $w \to \infty$, and the 0–1 error vanishes when the softmax scale grows after w.
- **Parameter-efficient backbone.** We show how a spectral FWNN parameterization fits naturally with cosine-based scoring: it preserves the unit-norm geometry, simplifies gradients via the chain rule over the spectral basis, and reduces model size and compute.
- **Constructive illustrations.** We provide synthetic grammars and controlled codebooks that exhibit the predicted concentration and calibration, serving as diagnostic testbeds for loss behavior and scaling with w.

Scope and limitations. Our guarantees hinge on assumptions that are natural but not automatic. First, a *mean-separation* condition must hold: the true class should exhibit a strictly larger expected match fraction than any competitor; otherwise any score monotone in Hamming similarity is Bayes-inefficient. Second, the spherical code should be sufficiently separated (small μ) so that the affine conversion between cosine and Hamming remains well-conditioned; poor codes degrade both sensitivity and concentration. Third, weak temporal dependence is required to invoke Hoeffding/Azuma concentration for window averages. These conditions are explicit in our statements and lead to straightforward stress tests.

Why this matters. The framework offers a rare combination of interpretability and practicality. The spherical code embedding turns similarity learning into a controlled geometric problem; the soft Hamming loss makes discrete error visible in a smooth objective; and FWNNs deliver compactness without sacrificing differentiability. Together, these ingredients yield lightweight, analyzable models for symbolic data—useful when resources are limited, when losses must be audited, or when one wants a principled connection to Hamming-style criteria.

Organization. Section 4 formalizes the cosine–Hamming link and defines the soft Hamming surrogate. Section 5 develops concentration bounds and proves consistency of the softmax classifier built on $\widehat{\alpha}$. We then discuss FWNN parameterization, computational aspects, and illustrative experiments, and conclude with limitations and open problems.

2 Introduction to Spherical Codes

A *spherical code* in $\mathbb{R}^n$ is a finite set $C \subset \mathbb{S}^{n-1} := \{x \in \mathbb{R}^n : \|x\|_2 = 1\}$. Its quality is often measured by the *minimum angular distance*

$$\theta(C) \;:=\; \min_{\substack{x,y\in C\\ x\neq y}} \arccos\big(\langle x, y\rangle\big),$$

or, equivalently, by the (non-absolute) *coherence*

$$\mu(C) \;:=\; \max_{\substack{x,y\in C\\ x\neq y}} \langle x, y\rangle \qquad (\text{so that } \mu(C) = \cos\theta(C)).$$

Intuitively, a good code spreads points as evenly as possible over the unit sphere, i.e., it has large $\theta(C)$ (small $\mu(C)$).

Core optimization problems. Two equivalent viewpoints are standard:

1. *Maximal cardinality at a prescribed angle:*

$$A(n,\theta) \;:=\; \max\big\{\, |C| : C \subset \mathbb{S}^{n-1},\ \theta(C) \geq \theta \,\big\}.$$

2. *Largest achievable angle (or smallest coherence) at fixed size M:*

$$\Theta(n,M) \;:=\; \sup_{|C|=M} \theta(C), \qquad \mu^\star(n,M) \;:=\; \inf_{|C|=M} \mu(C).$$

Here $A(n,\theta)$ generalizes spherical packing, while $\Theta(n,M)$ asks how far M points can "repel" each other on $\mathbb{S}^{n-1}$.

Examples and basic constructions.

- **Simplex code:** For $M = n + 1$ there is a regular simplex with pairwise inner products $-1/n$. Hence $\mu = -1/n$ and $\theta = \arccos(-1/n)$.
- **Orthoplex code:** The $2n$ coordinate axes $\{\pm e_i\}_{i=1}^n$ satisfy $\max_{x \neq y} \langle x, y \rangle = 0$, so $\theta = \pi/2$.
- **Kissing number:** The kissing number $\tau(n)$ equals $A(n, \pi/3)$, since tangency directions have mutual angle $\arccos(1/2) = \pi/3$. Classical values include $\tau(2) = 6$, $\tau(3) = 12$, $\tau(4) = 24$, $\tau(8) = 240$, and $\tau(24) = 196{,}560$.

Bounds. Determining $A(n, \theta)$ or $\Theta(n, M)$ is difficult, but several general bounds are known:

- *Spherical cap packing* (volume argument) yields elementary upper bounds via the surface area of caps of radius $\theta/2$.
- *Linear programming* (Delsarte–Levenshtein) gives strong upper bounds using positivity of certain spherical polynomials.
- *Semidefinite programming* further sharpens these bounds in many fixed dimensions and angles.
- *Welch bound* (for M unit vectors in $\mathbb{R}^n$) on the *absolute* coherence $\nu(C) := \max_{x \neq y} |\langle x, y \rangle|$:

$$\nu(C) \;\geq\; \sqrt{\frac{M - n}{n(M - 1)}},$$

  with equality precisely for *equiangular tight frames* when they exist.

Connections and applications. Spherical codes arise in many areas:

- **Communication theory:** Signal constellations on the sphere (e.g., phase constellations and higher-dimensional analogues), beamforming, and quantization under AWGN.
- **Frame theory & signal processing:** Design of low-coherence dictionaries for sparse recovery and compressed sensing.
- **Geometry & combinatorics:** Links to lattice vectors (e.g., E_8 and the Leech lattice), spherical t-designs, and packing/covering problems.

Summary. Spherical codes formalize the task of distributing finitely many points as evenly as possible on $\mathbb{S}^{n-1}$. The subject blends deep geometric questions with practical design in communications. Today's toolkit combines explicit constructions, asymptotic probabilistic methods, and powerful convex bounds (LP/SDP).

3 Introduction to Fourier Weighted Neural Networks

Modern neural networks achieve remarkable accuracy across vision, language, and tabular domains, but their dense parameterization imposes substantial runtime and memory costs, which can hinder deployment on resource-constrained hardware. Fourier Weighted Neural Networks (FWNNs) address this tension by *constructing* layer weights from a compact Fourier basis rather than learning each entry of a dense matrix independently. This spectral parameterization preserves standard backpropagation while drastically reducing the number of free parameters and improving training stability. :contentReference[oaicite:0]index=0

Core idea. Instead of storing a full weight matrix $W \in \mathbb{R}^{H\times S}$, FWNNs generate each entry from a truncated cosine series

$$W_{rs} \;=\; \sum_{j=-R}^{R} c_j \,\cos(j\,\theta_{rs}), \qquad \theta_{rs} \;=\; \frac{r+s}{N-1}\,\pi,$$

where $\{c_j\}_{j=-R}^{R}$ are shared Fourier coefficients, R is the maximal mode ("spectral rank"), and N normalizes indices. This replaces $\mathcal{O}(HS)$ parameters by only $(2R+1)$ coefficients (plus biases), enforcing a smooth, low-frequency structure over anti-diagonals of W. Because $|\cos(\cdot)| \le 1$, entries are bounded, which in turn yields well-conditioned gradients. :contentReference[oaicite:1]index=1

Optimization and stability. A practical advantage of the cosine-based construction is that gradients are naturally *bounded and non-vanishing*. Consequently, FWNNs tolerate substantially higher learning rates without destabilizing training, often accelerating convergence relative to dense baselines while keeping the usual forward/backward mechanics unchanged (the gradient w.r.t. c_j aggregates dense gradients against the fixed cosine basis). :contentReference[oaicite:2]index=2

Computational footprint. The spectral parameterization yields linear scaling in both runtime and memory with respect to the number of layers and the spectral rank:

$$\text{complexity} \;=\; \mathcal{O}\big((2R+1)\times \#\text{Layers}\big),$$

a favorable contrast to the quadratic storage of dense layers. This economy also improves cache locality (via reusable cosine bases) and serves as an implicit regularizer through a low-frequency bias. :contentReference[oaicite:3]index=3

Empirical evidence. Across a range of standard classification and regression tasks (e.g., Breast Cancer, Iris, Wine, Digits, Diabetes, and California Housing), FWNNs maintain competitive accuracy and error metrics while using far fewer parameters; experiments further corroborate the stability of large learning rates and rapid loss reduction under the bounded-gradient regime. :contentReference[oaicite:4]index=4

Summary. FWNNs recast weight learning as spectral coefficient estimation on a unit cosine basis. The result is a compact, hardware-friendly layer family with: (i) strong parameter efficiency, (ii) stable, large-step optimization due to bounded gradients, and (iii) linear scaling in depth and spectral rank—without sacrificing performance on diverse benchmarks. :contentReference[oaicite:5]index=5

4 Why Maximally Separated Spherical Codes with FWNNs?

Setting. Let $P = [p_1, \dots, p_K] \in \mathbb{R}^{d\times K}$ be a *maximally separated spherical code*: $\|p_i\|_2 = 1$ and

$$\mu \;\stackrel{\text{def}}{=}\; \max_{i\neq j} |\langle p_i, p_j\rangle| \quad \text{is minimal over all } K\text{-point codebooks on } \mathbb{S}^{d-1}.$$

Given a token window $(i_1, \dots, i_w)$ we map it to the block-concatenation

$$x \;=\; \phi(i_{1:w}) \;=\; [\,p_{i_1}; p_{i_2}; \dots; p_{i_w}\,] \in \mathbb{R}^{dw}, \qquad \hat{x} \;=\; \frac{x}{\|x\|_2} \in \mathbb{S}^{dw-1}.$$

For two windows $i_{1:w}$ and $j_{1:w}$ define

$$\rho_t \;\stackrel{\text{def}}{=}\; \langle p_{i_t}, p_{j_t}\rangle, \qquad \alpha \;\stackrel{\text{def}}{=}\; \frac{1}{w}\#\{t:\, i_t = j_t\}.$$

Because the blocks are disjoint in the concatenation, we have

$$\cos(\hat{x}, \hat{y}) \;=\; \frac{\langle x, y\rangle}{\|x\|_2 \, \|y\|_2} \;=\; \frac{1}{w} \sum_{t=1}^{w} \rho_t. \tag{1}$$

4.1 From Cosine to (Soft) Hamming Similarity

Maximal separation implies $|\rho_t| \le \mu$ for $i_t \neq j_t$ and $\rho_t = 1$ if $i_t = j_t$. Thus

$$\underbrace{-\mu + \alpha(1+\mu)}_{\text{all mismatches at } -\mu} \;\le\; \cos(\hat{x}, \hat{y}) \;\le\; \underbrace{\mu + \alpha(1-\mu)}_{\text{all mismatches at } +\mu} \quad . \tag{2}$$

Inverting the bounds yields a tight conversion between cosine and *Hamming similarity* α:

$$\frac{\cos(\hat{x}, \hat{y}) - \mu}{1-\mu} \;\le\; \alpha \;\le\; \frac{\cos(\hat{x}, \hat{y}) + \mu}{1+\mu}. \tag{3}$$

Hence, with small μ (max-sep codes), $\cos(\hat{x}, \hat{y})$ is an *affine, tightly controlled estimator* of the position-wise Hamming similarity. In particular, a single-position change modifies the cosine by about $(1-\mu)/w$:

$$\Delta \cos \;\approx\; \frac{1-\mu}{w}. \tag{4}$$

Concentration for long windows. Assume the pairs (i_t, j_t) are independent draws from some distribution on $\{1, \dots, K\}^2$ with $\mathbb{E}[\rho_t] = \bar{\rho}$ and $-\mu \le \rho_t \le 1$. By Hoeffding,

$$\Pr\Big(|\cos(\hat{x}, \hat{y}) - \bar{\rho}| \ge \varepsilon\Big) \;\le\; 2\exp\!\left(-\frac{2w\varepsilon^2}{(1+\mu)^2}\right), \tag{5}$$

i.e., $\cos(\hat{x}, \hat{y})$ concentrates at rate $\mathcal{O}(1/\sqrt{w})$. For random, unaligned windows with $\Pr[i_t = j_t] = 1/K$ and near-orthogonal mismatches ($\mathbb{E}[\rho_t \mid i_t \neq j_t] \approx 0$), we have $\bar{\rho} \approx 1/K$; thus typical random pairs are nearly orthogonal in the concatenated, normalized space.

4.2 A "Soft Hamming" Error and Calibrated Loss

Using (3), define the cosine-calibrated *soft Hamming similarity*

$$\widehat{\alpha}(\hat{x}, \hat{y}) \;=\; \mathrm{clip}\!\left(\frac{\cos(\hat{x}, \hat{y}) - \mu}{1-\mu},\, 0,\, 1\right), \qquad \widehat{D}_H(\hat{x}, \hat{y}) \;\stackrel{\text{def}}{=}\; 1 - \widehat{\alpha}(\hat{x}, \hat{y}), \tag{6}$$

where $\widehat{D}_H$ is the corresponding *soft Hamming distance* (soft error). By (4), $\widehat{D}_H$ changes by $\approx (1-\mu)/w$ under a single-position flip, giving a transparent sensitivity scale.

Pairwise loss. For a positive pair $(\hat{x}, \hat{y}^+)$ and a set of negatives $\{\hat{y}_j^-\}$ one may use a margin loss on the soft errors,

$$\mathcal{L}_{\text{pair}} \;=\; \big[\, \widehat{D}_H(\hat{x}, \hat{y}^+) - \widehat{D}_H(\hat{x}, \hat{y}^-) + \delta \,\big]_+, \tag{7}$$

with margin $\delta > 0$ (and averaging over negatives j). Equivalently, using similarities: $\mathcal{L}_{\text{pair}} = \big[\, \widehat{\alpha}(\hat{x}, \hat{y}^-) - \widehat{\alpha}(\hat{x}, \hat{y}^+) + \delta \,\big]_+$.

Multiclass loss (softmax on soft Hamming). If each class c provides a reference $\hat{y}_c$ (prototype or feature), define logits by a scaled similarity

$$s_c(\hat{x}) \;=\; \beta\, \widehat{\alpha}(\hat{x}, \hat{y}_c), \qquad \mathcal{L}_{\text{CE}} \;=\; -\log \frac{e^{s_y(\hat{x})}}{\sum_c e^{s_c(\hat{x})}}, \tag{8}$$

where $\beta > 0$ is a temperature. This implements a cross-entropy whose scores are *calibrated estimates of Hamming similarity* between the input window and class references.

Calibration properties. By (3) and (5), $\widehat{\alpha}$ is an affine, high-probability estimator of the true Hamming similarity α with deviation controlled by μ and concentration improving as $\mathcal{O}(1/\sqrt{w})$. Thus $\widehat{D}_H$ is a Lipschitz surrogate of the Hamming error that (i) has a known flip sensitivity $\approx (1-\mu)/w$, and (ii) yields well-behaved gradients through the cosine on the unit sphere.

4.3 Why an FWNN Backbone?

Let the last linear map be parameterized spectrally as in FWNN:

$$W_{rs} \;=\; \sum_{k=-R}^{R} c_k \, \varphi_k(r,s),$$

with structured basis (e.g., Fourier/circulant/Toeplitz). This yields:

- **Parameter efficiency.** Dense $W \in \mathbb{R}^{H_{\text{out}} \times H_{\text{in}}}$ needs $H_{\text{out}} H_{\text{in}}$ parameters, whereas FWNN needs $\mathcal{O}(R) + \mathcal{O}(H_{\text{out}})$ coefficients and biases; typically $R \ll \min(H_{\text{in}}, H_{\text{out}})$.
- **Fast inference.** Convolutive/circulant structure implies Wv can be applied in $\mathcal{O}(H \log H)$ via FFTs rather than $\mathcal{O}(H^2)$.
- **Simple gradients.** With a cosine-based score (e.g., (8)), the feature gradient has the standard L2-normalized form $\frac{\partial \mathcal{L}}{\partial f} = \frac{1}{\|f\|_2}(I - \hat{f}\hat{f}^\top)\, g(\hat{f})$, so the FWNN coefficients follow by the chain rule $\frac{\partial \mathcal{L}}{\partial c_k} = \sum_{r,s} \frac{\partial \mathcal{L}}{\partial W_{rs}} \, \varphi_k(r,s)$.

4.4 Takeaways

1. **Max-sep codes ⇒ cosine ≈ (soft) Hamming.** Via (3), the window cosine tightly estimates the fraction of *position-wise matches*, with deviation controlled by μ.
2. **Long windows concentrate.** By (5), $\cos(\hat{x}, \hat{y})$ concentrates at rate $\mathcal{O}(1/\sqrt{w})$; single-token flips change the cosine only by $\approx (1-\mu)/w$ (4).
3. **Soft Hamming error is a calibrated surrogate.** The quantity $\widehat{D}_H = 1 - \widehat{\alpha}$ provides a geometrically meaningful, Lipschitz surrogate for Hamming distance, usable in pairwise margins (7) or multiclass CE (8).
4. **FWNN brings efficiency without sacrificing geometry.** Spectral parameterization reduces memory/compute and meshes naturally with cosine-based objectives on unit-norm inputs.

5 Consistency of a Soft-Hamming Softmax with Unit-Norm Code Windows

Setting (unit-norm windows from a spherical code). Let $P = [p_1, \ldots, p_K] \in \mathbb{R}^{d \times K}$ be a *maximally separated spherical code* with $\|p_i\|_2 = 1$ and

$$\mu \;\stackrel{\text{def}}{=}\; \max_{i \neq j} |\langle p_i, p_j \rangle| \qquad \text{(small)}.$$

Given a token window $(i_1, \ldots, i_w)$, encode it by block-concatenation and unit-normalization

$$x = \phi(i_{1:w}) = [\, p_{i_1}; \ldots; p_{i_w} \,] \in \mathbb{R}^{dw}, \qquad \hat{x} = \frac{x}{\|x\|_2} \in \mathbb{S}^{dw-1}.$$

For two windows $i_{1:w}$, $j_{1:w}$ write $\rho_t = \langle p_{i_t}, p_{j_t} \rangle$ and define the (position-wise) *Hamming similarity*

$$\alpha(i_{1:w}, j_{1:w}) = \frac{1}{w} \#\{t : i_t = j_t\}.$$

Because blocks are disjoint, the cosine between normalized concatenations is the average of the per-position cosines:

$$\cos(\hat{x}, \hat{y}) = \frac{\langle x, y \rangle}{\|x\|_2 \|y\|_2} = \frac{1}{w} \sum_{t=1}^{w} \rho_t. \tag{9}$$

Maximal separation gives for each position: $\rho_t = 1$ if $i_t = j_t$ and $\rho_t \in [-\mu, \mu]$ if $i_t \neq j_t$. Hence the window cosine obeys the tight bounds

$$-\mu + \alpha(1 + \mu) \;\leq\; \cos(\hat{x}, \hat{y}) \;\leq\; \mu + \alpha(1 - \mu), \tag{10}$$

which invert to an affine sandwich for α:

$$\frac{\cos(\hat{x}, \hat{y}) - \mu}{1 - \mu} \;\leq\; \alpha \;\leq\; \frac{\cos(\hat{x}, \hat{y}) + \mu}{1 + \mu}. \tag{11}$$

We use the (clipped) *soft Hamming similarity* estimator

$$\widehat{\alpha}(\hat{x}, \hat{y}) \;=\; \mathrm{clip}\Big(\frac{\cos(\hat{x}, \hat{y}) - \mu}{1 - \mu},\, 0,\, 1 \Big), \qquad \widehat{D}_H(\hat{x}, \hat{y}) = 1 - \widehat{\alpha}(\hat{x}, \hat{y}). \tag{12}$$

Classifier and loss. Assume each class $c \in \{1, \ldots, K\}$ has a fixed reference (prototype) $\hat{y}_c$ in the same space. We define *softmax scores* from soft Hamming similarity,

$$s_c(\hat{x}) = \beta\, \widehat{\alpha}(\hat{x}, \hat{y}_c), \qquad \mathcal{L}_{\mathrm{CE}}(\hat{x}, y) = -\log \frac{e^{s_y(\hat{x})}}{\sum_{j=1}^{K} e^{s_j(\hat{x})}}, \tag{13}$$

where $\beta > 0$ is a scale (inverse temperature). This makes the CE-loss depend monotonically on estimated Hamming similarity.

5.1 Assumptions and concentration

We analyze the regime of long windows $w \to \infty$. Write $\mathbb{E}[\cdot]$ for expectation over the joint randomness of windows and their labels.

(A1) Identifiability by Hamming means. There exists $\delta > 0$ such that for (almost) every input with true class y,

$$\Delta \;\stackrel{\mathrm{def}}{=}\; \mathbb{E}[\alpha(\hat{x}, \hat{y}_y)] - \max_{j \neq y} \mathbb{E}[\alpha(\hat{x}, \hat{y}_j)] \;\geq\; \delta.$$

Intuitively, the true class yields a strictly larger *expected* fraction of position-wise matches than any competitor.

(A2) Boundedness and weak dependence. For each fixed pair $(\hat{x}, \hat{y}_c)$, the summands ρ_t in (9) are bounded in $[-\mu, 1]$ and either independent across t or sufficiently weakly dependent (e.g., a bounded-difference martingale or a geometrically mixing process) so that a Hoeffding/Azuma-type inequality holds:

$$\Pr\left(\left| \frac{1}{w} \sum_{t=1}^{w} \rho_t - \mathbb{E}[\rho_t] \right| \geq \varepsilon \right) \;\leq\; 2 \exp\left(-\frac{2w\,\varepsilon^2}{(1 + \mu)^2} \right).$$

(A3) Small coherence. $\mu < 1$ is fixed and small so that the affine map in (11) is well-conditioned; a single flip changes the cosine by $\approx (1 - \mu)/w$.

By (11) and (A2), the estimator $\widehat{\alpha}(\hat{x}, \hat{y}_c)$ concentrates around its mean at rate $\mathcal{O}(1/\sqrt{w})$ for each class c:

$$\Pr\big(|\widehat{\alpha}(\hat{x}, \hat{y}_c) - \mathbb{E}[\widehat{\alpha}(\hat{x}, \hat{y}_c)]| \geq \epsilon\big) \;\leq\; 2\exp\left(-\frac{2w\,(1-\mu)^2\,\epsilon^2}{(1+\mu)^2}\right). \tag{14}$$

(We used that $\widehat{\alpha}$ is an affine transform of the average $\frac{1}{w}\sum \rho_t$ followed by clipping.)

5.2 Main result: expected loss converges as $w \to \infty$

Theorem 1 (Consistency of soft-Hamming softmax)**.** *Suppose* (A1)–(A3) *hold. Fix any* $\beta > 0$ *and let* $\mathcal{L}_{\mathrm{CE}}$ *be given by* (13)*. Then there exist constants* $c = c(\mu, \delta) > 0$ *and* $C = C(K, \beta)$ *such that, for all window lengths* w*,*

$$\mathbb{E}\big[\mathcal{L}_{\mathrm{CE}}(\hat{x}, y)\big] \;\leq\; \log\Big(1 + (K-1)e^{-\beta\delta/2}\Big) \;+\; C\,e^{-c\,w}.$$

In particular,

$$\lim_{w\to\infty} \mathbb{E}\big[\mathcal{L}_{\mathrm{CE}}(\hat{x}, y)\big] \;\leq\; \log\Big(1 + (K-1)e^{-\beta\delta/2}\Big),$$

and letting $\beta \to \infty$ *after* $w \to \infty$ *yields vanishing* 0–1 *error.*

Proof. We show that, with high probability for large w, the true class y has a *score gap* of at least $\beta\delta/2$ against every competitor, which makes the cross-entropy uniformly small on that event.

Step 1 (means are separated). By (A1), writing $\overline{\alpha}_c = \mathbb{E}[\alpha(\hat{x}, \hat{y}_c)]$,

$$\overline{\alpha}_y - \overline{\alpha}_j \;\geq\; \delta \qquad \text{for all } j \neq y.$$

Step 2 (estimators concentrate). Let $\overline{\widehat{\alpha}}_c = \mathbb{E}[\widehat{\alpha}(\hat{x}, \hat{y}_c)]$. Since $\widehat{\alpha}$ is an affine image of the average cosine (plus clipping), (14) gives, for any $\epsilon > 0$,

$$\Pr\big(\,|\widehat{\alpha}_c - \overline{\widehat{\alpha}}_c| \geq \epsilon\,\big) \;\leq\; 2\exp\left(-\frac{2w\,(1-\mu)^2\,\epsilon^2}{(1+\mu)^2}\right).$$

Moreover, because $\widehat{\alpha}$ and α are linked by (11), $\overline{\widehat{\alpha}}_y - \overline{\widehat{\alpha}}_j$ is bounded below by an affine image of $\overline{\alpha}_y - \overline{\alpha}_j$, hence also $\geq c_0\,\delta$ for some $c_0 \in (0, 1]$ depending only on μ. (With small μ, we can take $c_0 \approx 1$ and drop this technicality; to keep the exposition simple we keep writing δ below.)

Step 3 (a good event). Fix $\epsilon = \delta/4$. Define the event

$$\mathcal{E} \;=\; \bigcap_{j=1}^{K} \Big\{\; |\widehat{\alpha}_j - \overline{\widehat{\alpha}}_j| < \epsilon \;\Big\}.$$

By a union bound and (14),

$$\Pr(\mathcal{E}^c) \;\leq\; 2K\exp\left(-\frac{2w\,(1-\mu)^2\,\epsilon^2}{(1+\mu)^2}\right) \;=\; 2K\exp\left(-\frac{w\,(1-\mu)^2\,\delta^2}{8(1+\mu)^2}\right) \;\stackrel{\text{def}}{=}\; e^{-c\,w},$$

with $c = \frac{(1-\mu)^2\,\delta^2}{8(1+\mu)^2} - \frac{\log(2K)}{w}$; for all large w, c is positive and we can absorb constants into e^{-cw}.

On $\mathcal{E}$ we have, for all $j \neq y$,

$$\widehat{\alpha}_y - \widehat{\alpha}_j \;\geq\; (\overline{\widehat{\alpha}}_y - \epsilon) - (\overline{\widehat{\alpha}}_j + \epsilon) \;=\; (\overline{\widehat{\alpha}}_y - \overline{\widehat{\alpha}}_j) - 2\epsilon \;\geq\; \delta - 2\cdot\frac{\delta}{4} \;=\; \frac{\delta}{2}.$$

Therefore, the *score gap* satisfies

$$s_y - s_j = \beta(\widehat{\alpha}_y - \widehat{\alpha}_j) \;\geq\; \beta\,\frac{\delta}{2} \qquad (j \neq y) \quad \text{on } \mathcal{E}.$$

Step 4 (cross-entropy bound). On $\mathcal{E}$,

$$\mathcal{L}_{\mathrm{CE}} = \log\left(1 + \sum_{j \neq y} e^{-(s_y - s_j)}\right) \;\leq\; \log\Big(1 + (K-1)e^{-\beta\delta/2}\Big)\,.$$

Outside $\mathcal{E}$ we can use the trivial bound $0 \leq s_c \leq \beta$ (since $0 \leq \widehat{\alpha} \leq 1$), hence $\mathcal{L}_{\mathrm{CE}} \leq \log\sum_c e^{s_c} \leq \log\Big(Ke^{\beta}\Big) = \beta + \log K$.

Step 5 (take expectations). Split by $\mathbb{1}_{\mathcal{E}}$:

$$\mathbb{E}[\mathcal{L}_{\mathrm{CE}}] \;\leq\; \log\Big(1 + (K-1)e^{-\beta\delta/2}\Big) \cdot \Pr(\mathcal{E}) \;+\; (\beta + \log K) \cdot \Pr(\mathcal{E}^c).$$

Using $\Pr(\mathcal{E}^c) \leq e^{-cw}$ finishes the claim with $C = \beta + \log K$. Finally, taking $\beta \to \infty$ after $w \to \infty$ forces the error probability under the argmax rule to 0 (the margin in scores explodes), yielding 0–1 consistency. □

Remark 1 (About the FWNN backbone)**.** *The theorem does not require universal approximation by the backbone: it relies on the* input-level geometry *(unit-norm code blocks and cosine ↔ Hamming via* (11)*). An FWNN layer that linearly mixes coordinates (followed by an L2-normalization before scoring) preserves the boundedness and the Lipschitz dependence of the cosine, so the concentration and gap arguments remain valid as long as the mixing does not systematically collapse the expected gap in (A1).*

5.3 When the conclusion can fail (counterexamples)

The assumptions are essential:

- **(C1) No mean gap ($\Delta = 0$).** Two classes y and j may have the same expected match fraction (e.g., identical unigrams but different higher-order structure). Then *any* score that is a monotone function of α is Bayes-inefficient: no matter how large w is, the expected CE cannot converge to the Bayes optimum.
- **(C2) Large coherence ($\mu \approx 1$).** If the code is poorly separated, the affine map in (11) is ill-conditioned and a single flip changes the cosine by $\approx (1-\mu)/w \ll 1/w$. Concentration becomes weak and the finite-w probability of misranking may not decay fast enough for the bound to be useful.
- **(C3) Strong temporal dependence.** If the per-position cosines ρ_t exhibit long-range dependence or adversarial phase behavior, Hoeffding/Azuma bounds do not apply; the average need not concentrate at rate $\mathcal{O}(1/\sqrt{w})$, and the score gaps may not stabilize.
- **(C4) Head misspecification.** If scores are not an increasing function of a statistic aligned with class separation (e.g., using a non-monotone transform of the cosine), the argmax can be systematically biased even as $w \to \infty$.

5.4 Takeaways

1. Unit-norm concatenations from a well-separated spherical code yield a cosine that is an affine, tightly controlled proxy for Hamming similarity; its variance shrinks as $\mathcal{O}(1/w)$.
2. A simple softmax with logits $s_c = \beta\,\widehat{\alpha}(\hat{x}, \hat{y}_c)$ is *consistent*: under a fixed mean gap and mild dependence, the expected CE converges to a small limit and 0–1 error vanishes as $w \to \infty$ and then $\beta \to \infty$.

3. Universal approximation by the backbone is *not* required for this conclusion; the argument is geometric and concentration-based at the input level.

4. Violate the assumptions (no mean gap, large μ, strong dependence), and convergence can fail—use these cases as stress tests.

6 Hilbert Space Interpretation for $w \to \infty$

Corollary 1 (Unit-norm limit vectors in ℓ^2)**.** *For each finite window length w, the concatenated and normalized vector*

$$\hat{x}^{(w)} \;=\; \frac{[\,p_{i_1};\, p_{i_2};\, \ldots;\, p_{i_w}\,]}{\|[\,p_{i_1};\, \ldots;\, p_{i_w}\|_2} \;\in\; \mathbb{R}^{dw}$$

is a unit vector in the Hilbert space $\mathbb{R}^{dw}$, i.e. $\|\hat{x}^{(w)}\|_2 = 1$.

As $w \to \infty$, the ambient dimension dw also diverges. We may embed all $\hat{x}^{(w)}$ into the infinite-dimensional Hilbert space

$$\ell^2(\mathbb{N};\mathbb{R}^d) \;=\; \left\{(u_1, u_2, \ldots) : u_t \in \mathbb{R}^d,\, \sum_{t=1}^{\infty} \|u_t\|_2^2 < \infty\right\}.$$

Then every $\hat{x}^{(w)}$ is still a unit-norm element of ℓ^2.

If the sequence of tokens (i_t) is generated by a stationary ergodic process, the law of large numbers implies that for any other window sequence (j_t),

$$\langle \hat{x}^{(w)}, \hat{y}^{(w)} \rangle \;\longrightarrow\; \mathbb{E}[\rho_t] \qquad \text{as } w \to \infty,$$

where $\rho_t = \langle p_{i_t}, p_{j_t} \rangle$. In other words, the family $\{\hat{x}^{(w)}\}_w$ remains on the unit sphere in Hilbert space, and while the vectors themselves spread over more coordinates, their inner products converge to deterministic limits given by the expected (soft) Hamming similarity.

Intuition. For small w, $\hat{x}^{(w)}$ is just a normalized vector in some finite-dimensional space. As w grows, the vector lives in higher and higher dimensions, but always has length 1. In the infinite limit, the vector itself does not settle to a single finite object, but any inner product between two such vectors *does* converge to a well-defined expected value. Thus the "geometry" stabilizes even though the coordinates keep expanding.

7 Possible Applications

The proposed framework of spherical codes with FWNN backbones has potential in several domains:

- **Mini-scale text generation.** By calibrating similarity directly against Hamming error, the model is well-suited for lightweight language models on constrained hardware, such as mobile devices or embedded systems.

- **Communication systems.** The same spherical code geometry underlies error-correcting signal constellations; the soft-Hamming classifier can naturally extend to joint source–channel coding and symbol recovery tasks.

- **Biological sequence modeling.** DNA and protein strings are inherently discrete; mapping them through maximally separated codes yields interpretable similarity measures with calibrated gradients.

- **Information retrieval.** Matching queries against a large database can be phrased as nearest-neighbor search under Hamming-like metrics; the FWNN-based surrogate provides differentiable, trainable scoring functions.
- **Resource-efficient deep learning.** FWNN layers compress dense transformations into a handful of Fourier coefficients, offering practical gains in model size and inference speed without sacrificing accuracy.

8 Application: Single-Instance Sampling of Polyphonic Music via Spherical Codes

Overview. From a *single* symbolic input (e.g., MIDI) we build a self-contained pipeline: (i) tokenize the piece and write the tokens to a plain-text file; (ii) train and sample a *Fourier-weighted neural network* (FWNN) on *spherical codes* to produce a *new* token text file; (iii) decode that new token file back into polyphonic music.

Step 1: Tokenization from a single piece (write tokens to text). We extract polyphonic intervals $\mathcal{I} = \{(v, s, e, \nu)\}$, with voice v, onsets/offsets $s, e \in \mathbb{R}_{\geq 0}$ (quarter lengths), and $\nu = ((p_i, d_i, \alpha_i, r_i))_{i=1}^{m}$ (MIDI pitch p_i, local duration d_i, velocity α_i, rest flag r_i). An interval graph G is built from temporal overlap and pruned by a chord/voice kernel; its connected components $C_1, \ldots, C_M$ are our atomic units. Each component is embedded by a *Fourier-weighted* encoder onto the unit sphere, $\mathbf{z} = E_\theta(C) \in \mathbb{S}^{d-1}$. A spherical codebook $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\} \subset \mathbb{S}^{d-1}$ defines tokens by nearest-codeword quantization:

$$k = \arg\max_j \langle \mathbf{z}, \mathbf{c}_j \rangle, \quad \tau_k \text{ is the token.}$$

We write the *chronological* token sequence to a text file (one space-separated line). For decoding later, we also persist a vocabulary in which each token τ_k indexes the set ("bucket") of components assigned to $\mathbf{c}_k$; optionally, buckets are refined by a timing-invariant kernel threshold $K(C, C') > \delta$.

Step 2: Train & sample spherical codes with FWNN (produce a new token file). We train a next-token FWNN on the *text* tokens. Let the context window collect the last w codewords $\mathbf{c}_{k_{t-w}}, \ldots, \mathbf{c}_{k_{t-1}}$ (concatenated and normalized) as input $\mathbf{x}_t$. The FWNN outputs logits that respect the spherical geometry by using cosine similarity to a learnable class embedding $E \in \mathbb{R}^{K \times d}$ (or re-using $\mathcal{C}$) with a coherence-aware scale; a standard cross-entropy trains the model to predict k_t. At inference, we autoregress over tokens with temperature/top-k (or nucleus) sampling to generate a *new* token sequence, which we again write to a text file.

Step 3: Decode the new tokens back to music. Given the new token text file $\tau_{t_1}, \ldots, \tau_{t_N}$, we map each token to a concrete component by selecting from its bucket: either *random* (seeded) or *round-robin* (modulo) for diversity without repetition. Inside each chosen component $\hat{C}_n$, relative micro-timing is preserved by local onset subtraction $t \mapsto t - t_{\min}(\hat{C}_n)$, and components are concatenated by a running offset (optionally with a gap):

$$(s, e) \mapsto \big(s - t_{\min}(\hat{C}_n) + \text{offset},\ e - t_{\min}(\hat{C}_n) + \text{offset}\big), \quad \text{offset} \leftarrow \text{offset} + \text{len}(\hat{C}_n) + \text{gap}.$$

We then emit note-on/off per voice to a Standard MIDI File (PPQ grid) or stream to a live MIDI port.

Why this split works. Separating *token writing* (Step 1), *FWNN training/sampling* on spherical codes (Step 2), and *decoding* (Step 3) cleanly divides concerns: the first step fixes the discretization (geometry-aware, style-preserving), the second learns sequence structure purely over *text tokens*, and the third restores the original intra-component timing while controlling phrasing via the global offset and gap.

Reproducibility. We log: encoder checkpoint θ, codebook $\mathcal{C}$ (and version), token text files (original and generated), FWNN checkpoint and sampling temperature/top-k, the bucket refinement threshold δ, and the within-bucket selection mode (random/round-robin) with RNG seed. This suffices to exactly regenerate MIDI from the generated token file.

"'latex

9 Further Ideas: Kernel-Based Tokenization on Arbitrary Semantic Spaces

Setup. Let (X, k) be any finite or countably-finite semantic space equipped with a positive definite (p.d.) kernel $k : X \times X \to [-1, 1]$. (If k is not already bounded, replace it by the cosine-normalized kernel $\tilde{k}(x, y) = \frac{k(x,y)}{\sqrt{k(x,x)\,k(y,y)}} \in [-1, 1]$.) Assume we are given a single training sequence $(x_t)_{t=1}^{T}$ with $x_t \in X$.

Kernel-threshold tokenization. Fix a similarity threshold $\delta \in (0, 1)$ and form the graph $G_\delta = (V, E)$ with $V = \{x_t : 1 \le t \le T\}$ and

$$E = \big\{\{u, v\} \subseteq V \ : \ k(u, v) > \delta\big\}.$$

The connected components of G_δ define a quotient $V/\!\sim_\delta$. Each equivalence class becomes a *token*; write $\pi_\delta : V \to \mathcal{T}$ for the map from elements to tokens. Optionally pick a representative $r(\tau) \in \tau$ (e.g. the medoid that maximizes $\sum_{x\in\tau} k(x, \cdot)$) for fast decoding. This generalizes the music "connected-components" idea: two items x, y receive the same token whenever there exists a path $x = x_0, \ldots, x_m = y$ with $k(x_i, x_{i+1}) > \delta$ for all i.

Learning on spherical codes (FWNN back-end). Embed tokens into $\mathbb{S}^{d-1}$ by constructing a spherical code $\{c_\tau \in \mathbb{S}^{d-1} : \tau \in \mathcal{T}\}$ with large minimal pairwise angle. We then train a lightweight Fourier-Weighted Neural Network (FWNN) classifier $f_\theta : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{T}|}$ with targets c_τ (cosine or cross-entropy loss) for *next-token prediction*: given a context window around time t, the model predicts $\pi_\delta(x_{t+1})$.[1]

Generative procedure. After training, generation proceeds in two steps. (i) From the current context, sample a token $\hat{\tau}_{t+1}$ according to the FWNN scores. (ii) Decode to an element $\hat{x}_{t+1} \in \hat{\tau}_{t+1}$, e.g. by drawing uniformly from the class, by choosing the class representative $r(\hat{\tau}_{t+1})$, or by a soft kernel-weighted pick within the class. Iterating yields a sequence $\hat{x}_1, \hat{x}_2, \ldots$ in X given a seed $\hat{x}_1$. Hence, any finite p.d. semantic space (X, k) together with a single observed sequence induces a generative next-element model.

Examples.

- **Word–context prototypes.** Let X be the set of representative contexts for word types derived from BPE tokens (left/right slots in a fixed window). Choose a context kernel

[1] Any standard context featurization is admissible: kernel mean embeddings; Nyström/RFF features of k; or direct RKHS features via $k(\cdot, x_j)$ on a dictionary.

k (content overlap × positional similarity), normalize to $[-1, 1]$, and cluster by $k > \delta$. Each word obtains one or several context tokens; FWNN learns to predict the next token and thereby the next BPE/word. This supports polysemy naturally by allowing multiple tokens per word.

- **Logical semantic spaces.** Let X be a finite set of logical objects (e.g. predicates, formulas, or world–formula pairs) with a p.d. semantic kernel k that captures logical proximity (e.g. agreement across a model set, or a proof-theoretic overlap). The same thresholding/tokenization yields symbols for "semantically similar" objects; FWNN then models next-token dynamics in logical sequences.

Why this works. Because k is p.d., there exists a feature map $\phi : X \to \mathcal{H}$ with $k(x, y) = \langle \phi(x), \phi(y) \rangle$. Thresholding at $\delta > 0$ groups elements with acute angles in $\mathcal{H}$; connected components create coherent clusters that are robust to small perturbations (transitive closure). The spherical-code target set for tokens gives large inter-class margins while keeping the classifier small (FWNN uses a compact Fourier basis).

Practical notes.

1. **Normalization.** Use cosine normalization so that $k(x, x) = 1$ and $k \in [-1, 1]$; then δ has a direct geometric meaning.
2. **Controlling granularity.** Tune δ or require $k(x, y) > \delta + \gamma$ with margin $\gamma > 0$; or use size-capped components to avoid giant classes.
3. **Representatives.** Medoids maximize within-class kernel sum and serve as canonical decodings; multiple representatives per token can better cover multi-modal classes.
4. **Efficiency.** Build the similarity graph with approximate neighbors in the RKHS (Nyström/RFF + ANN). All steps are streaming: tokens can be updated online as new items arrive.

Outlook. The scheme turns *any* p.d. semantic space into a symbolic process: kernel-threshold tokenization yields a compact alphabet; spherical codes give a geometric target for classification; FWNN provides a fast predictor. This unifies music components, word–context prototypes, and logical objects under the same kernel-to-token pipeline, enabling one-sequence training and generation on top of (X, k).

10 Why the kernel→token→FWNN pipeline works (theory & heuristics)

Setting. Let (X, k) be a finite (or countably finite) set with a positive–definite kernel $k : X \times X \to [-1, 1]$. W.l.o.g. assume cosine normalization so that $k(x, y) = \langle \phi(x), \phi(y) \rangle$ for an embedding $\phi : X \to \mathcal{H}$ with $\|\phi(x)\| = 1$ for all $x \in X$. Fix a threshold $\delta \in (0, 1)$ and form the similarity graph G_δ on the observed items (nodes are the distinct elements seen in a single training sequence) with an undirected edge $\{x, y\}$ iff $k(x, y) > \delta$. Tokens are the connected components of G_δ. Tokens are embedded as unit vectors on the sphere (a spherical code), and a lightweight FWNN classifier is trained to predict the next token from context features.

We first give a *recoverability* statement for the tokenization step, then explain why the spherical-code/FWNN back-end admits large margins and consistent learning from a single (long) sequence.

A cluster-recoverability guarantee. Assume that X admits a latent partition $\{C_1, \ldots, C_M\}$ ("semantic classes"), well-separated in the RKHS.

Assumption A (spherical cluster model). There exist unit vectors $\mu_1, \ldots, \mu_M \in \mathcal{H}$ and a radius $r \in (0, 1)$ such that for every $x \in C_j$,

$$\|\phi(x) - \mu_j\| \leq r,$$

and inter-class center similarity is bounded by $\gamma < 1$:

$$\langle \mu_j, \mu_\ell \rangle \leq \gamma \qquad (j \neq \ell).$$

This is a standard "small within-class, large between-class angle" condition. It is satisfied, e.g., when classes are tight in cosine geometry (same meaning or function) and different classes point to sufficiently different directions.

Proposition 1 (Tokenization correctness under a margin)**.** *Under Assumption A, if the threshold δ satisfies*

$$\boxed{1 - 2r^2 \;\geq\; \delta \;>\; \gamma + 2r + r^2}$$

then for any finite sample whose distinct elements cover a subset of $\bigcup_j C_j$, the graph G_δ has no inter-class edges and every within-class pair is connected; hence the connected components of G_δ coincide with (the observed parts of) the true classes $\{C_j\}$.

Proof sketch. For unit vectors u, v we have $\langle u, v \rangle = 1 - \frac{1}{2}\|u - v\|^2$. If $x, y \in C_j$ then $\|\phi(x) - \phi(y)\| \leq 2r$, hence $\langle \phi(x), \phi(y) \rangle \geq 1 - \frac{1}{2}(2r)^2 = 1 - 2r^2$; this is the *within-class lower bound.* If $x \in C_j$, $y \in C_\ell$ with $j \neq \ell$, then by polarization

$$\langle \phi(x), \phi(y) \rangle = \langle \mu_j, \mu_\ell \rangle + \langle \phi(x) - \mu_j, \mu_\ell \rangle + \langle \mu_j, \phi(y) - \mu_\ell \rangle + \langle \phi(x) - \mu_j, \phi(y) - \mu_\ell \rangle,$$

whose absolute correction terms are bounded by r, r, and r^2 respectively. Thus $\langle \phi(x), \phi(y) \rangle \leq \gamma + 2r + r^2$, the *between-class upper bound.* Choosing δ strictly between these bounds forbids inter-class edges while ensuring every within-class pair is (directly) adjacent. Connectivity in each C_j follows immediately. □

Consequences. (i) Tokenization is *stable*: as long as perturbations keep r and γ within the margin, the components (tokens) do not change. (ii) The choice of δ controls granularity: larger δ yields finer tokens; smaller δ yields coarser tokens. (iii) The argument works for any p.d. kernel $k \in [-1, 1]$ after cosine normalization, hence applies to connected components of music, to word–context prototypes, and to logical semantic objects alike.

Why spherical codes for token targets. Let $\{c_\tau \in \mathbb{S}^{d-1}\}$ be the token targets with minimal pairwise inner product $\max_{\tau \neq \tau'} \langle c_\tau, c_{\tau'} \rangle \leq \beta < 1$ (i.e., a spherical code with angular margin $m = \arccos \beta$). When training a cosine/softmax classifier $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$ to map context features z to scores $\langle f_\theta(z), c_\tau \rangle$, the induced inter-class *angular margin* translates into a margin in probability space, which tightens standard generalization bounds for margin-based losses. Heuristically: the larger m, the smaller the overlap of class score distributions, and the lower the required model capacity to separate tokens—precisely why spherical codes are well suited as targets.

Why FWNN suffices. FWNN realises a structured linear operator with bounded spectrum (controlled by its few Fourier coefficients) followed by a smooth nonlinearity. Two key properties make it effective here: (i) *Lipschitz control:* bounded weights keep the cosine scores stable under small input shifts (robustness to contextual noise); (ii) *Bias toward smooth decision boundaries:* with spherical targets, FWNN needs only to align a few Fourier modes with the principal directions of the token clusters—far fewer degrees of freedom than a dense layer of the same width, yet retaining universal approximation over the sphere at mild depth.

From one sequence to a predictor (consistency heuristic). Assume the token sequence (τ_t) induced by $\pi_\delta(x_t)$ is generated by a stationary, β-mixing process with finite memory m (a standard assumption for language/music-style sequences). Then empirical conditional frequencies $\hat{p}(\tau_{t+1} \mid \tau_{t-m+1:t})$ computed along one *long* sample converge to the Bayes conditionals. Any sufficiently expressive predictor fed with context features that determine $\tau_{t-m+1:t}$ (e.g. kernel features, Nyström/RFF embeddings, or learned context encodings) will therefore achieve vanishing excess risk as sequence length grows. In practice, FWNN learns a smooth approximation of these conditionals with the spherical-code margin making the classification easier (higher signal-to-noise ratio between tokens).

Generation is coherent by construction. At inference, we (i) sample a next token according to the learned scores and (ii) decode it into an element of its component (representative/medoid or kernel-weighted sampling within the class). Because every class is a δ-tight component in the cosine geometry, replacements are constrained to *semantically similar* items ($k > \delta$). Thus the generated sequence stays in-distribution w.r.t. the kernel geometry learned from the single training trajectory—exactly the intended "same meaning, new surface" effect.

Examples (instantiations of X, k).

- **Word–context prototypes:** X are representative contexts of word types (built from BPE tokens). k combines soft content overlap with positional similarity; δ clusters contexts into sense-specific tokens; FWNN on spherical codes predicts the next token; decoding yields the next word (or BPE) as a member of the predicted token.
- **Logical semantic spaces:** X are formulas/predicates or world–formula pairs; k measures semantic proximity (agreement across a model set, proof overlap). The same pipeline yields next-formula prediction that respects logical similarity.

Summary. The PD kernel places X on the unit sphere of an RKHS; thresholding at δ recovers semantically coherent components under a mild angular-margin assumption; spherical codes turn tokens into well-separated targets; FWNN learns a smooth, margin-friendly next-token map from a single long sequence. Taken together, these steps yield a principled and practical generative method on arbitrary (X, k).