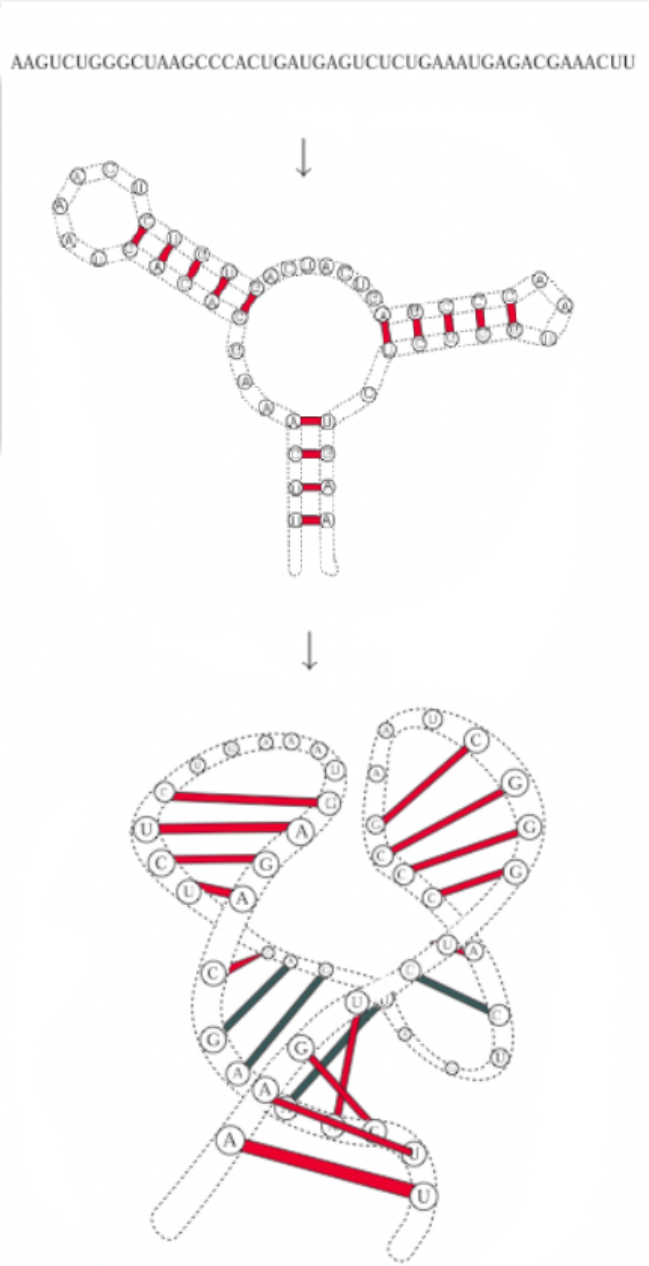
Niveles de Estructura del Ácido Ribonucleico (ARN)

Estructura primaria

- Secuencia lineal de nucleótidos unidos por enlaces fosfodiéster (puente químico).
- Compuesta por las bases nitrogenadas adenina (A), guanina (G), citosina (C) y uracilo (U).
- Define la información que porta la molécula.



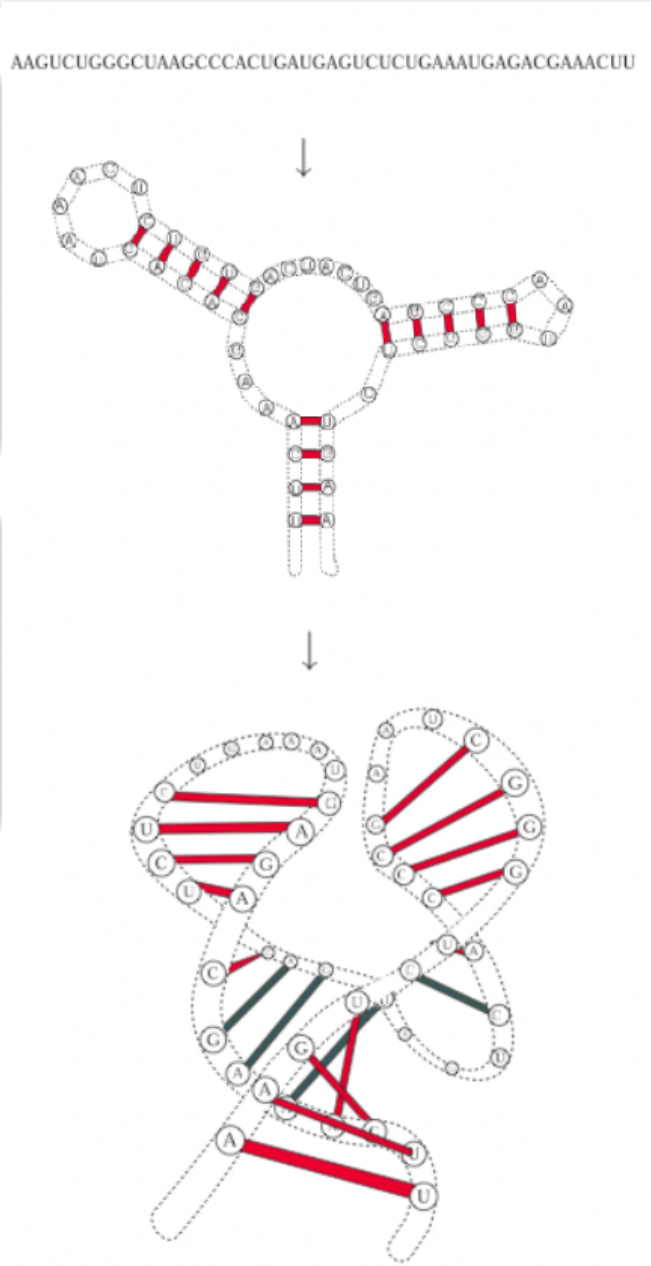
Niveles de Estructura del Ácido Ribonucleico (ARN)

Estructura primaria

- Secuencia lineal de nucleótidos unidos por enlaces fosfodiéster (puente químico).
- Compuesta por las bases nitrogenadas adenina (A), guanina (G), citosina (C) y uracilo (U).
- Define la información que porta la molécula.

Estructura secundaria

- Disposición local de la cadena simple mediante apareamientos de bases complementarias.
- Forma estructuras como horquillas, bucles y tallos, que estabilizan la molécula.



Niveles de Estructura del Ácido Ribonucleico (ARN)

Estructura primaria

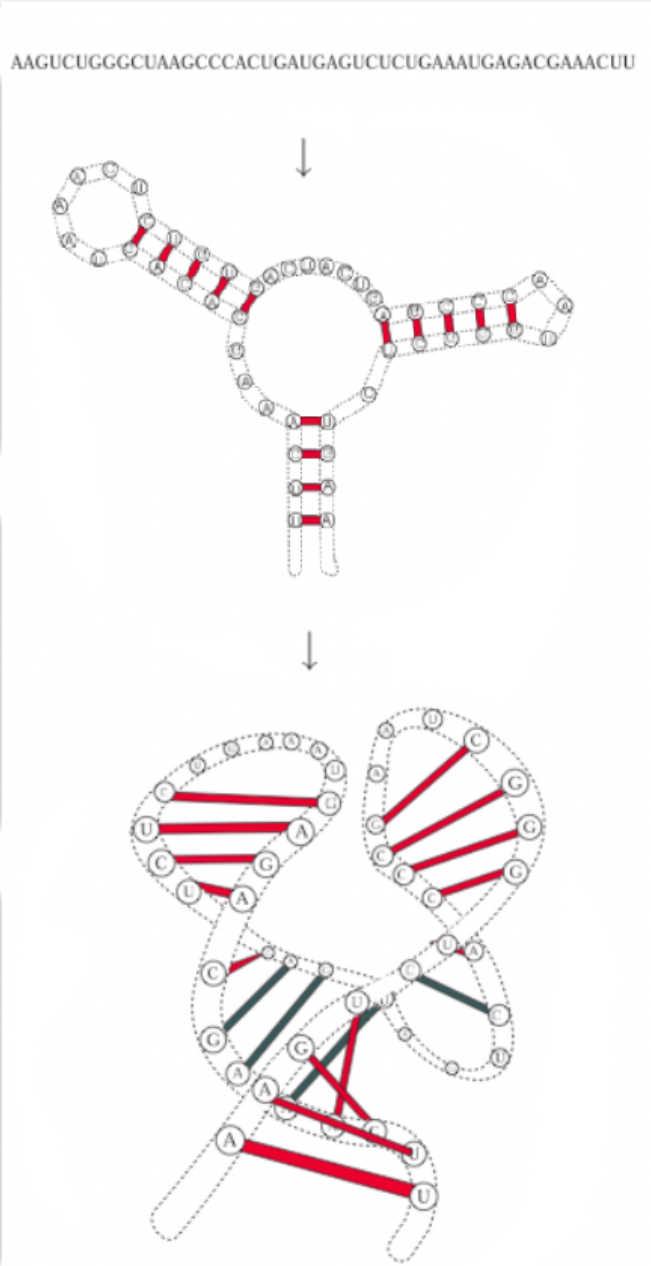
- Secuencia lineal de nucleótidos unidos por enlaces fosfodiéster (puente químico).
- Compuesta por las bases nitrogenadas adenina (A), guanina (G), citosina (C) y uracilo (U).
- Define la información que porta la molécula.

Estructura secundaria

- Disposición local de la cadena simple mediante apareamientos de bases complementarias.
- Forma estructuras como horquillas, bucles y tallos, que estabilizan la molécula.

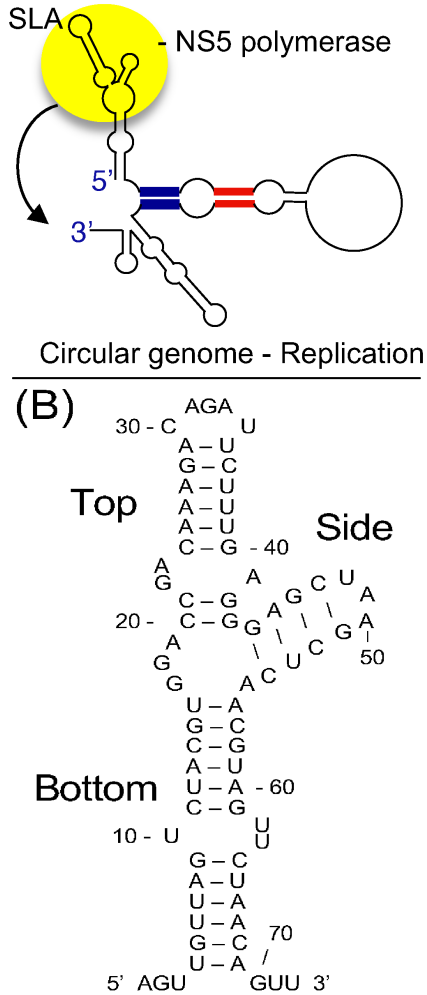
Estructura terciaria

- Plegamiento tridimensional de la molécula completa, resultado de interacciones entre regiones distantes.
- Permite la formación de formas funcionales complejas.



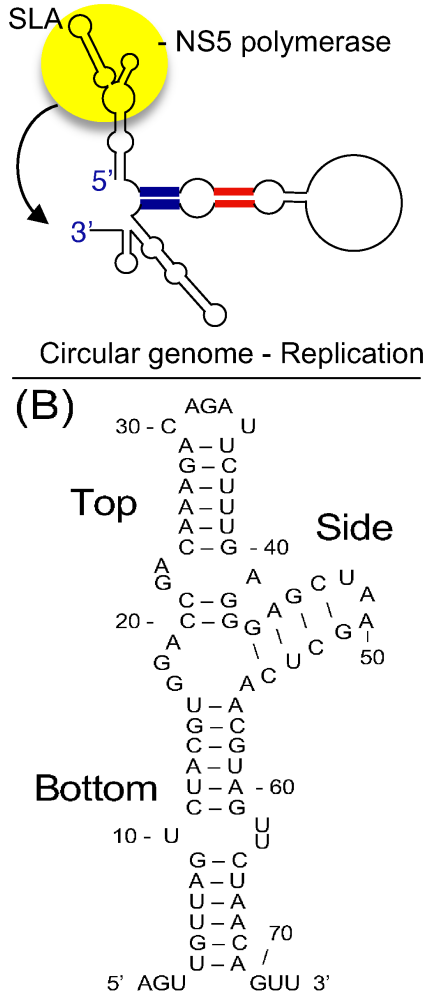
Contexto y Relevancia: La Estructura Stem-Loop A(SLA)

- En virus como el del Dengue, el ARN se pliega en estructuras funcionales. Una de las más importantes es el **Stem-Loop A (SLA)**.
- **¿Por qué es crucial?** Actúa como un promotor indispensable para la replicación del virus al interactuar con la proteína no estructural 5 (non-structural protein 5 -NS5).
- **La clave del problema:** Su función depende de su estructura 3D, no de la secuencia exacta de nucleótidos.
- **El desafío:** Las secuencias de ARN que contienen SLA tienen diferentes longitudes ('L'), que en nuestro caso varían entre 70-90 nucleótidos. Los algoritmos de clasificación clásicos requieren entradas de tamaño fijo.



Contexto y Relevancia: La Estructura Stem-Loop A(SLA)

- En virus como el del Dengue, el ARN se pliega en estructuras funcionales. Una de las más importantes es el **Stem-Loop A (SLA)**.
- **¿Por qué es crucial?** Actúa como un promotor indispensable para la replicación del virus al interactuar con la proteína no estructural 5 (non-structural protein 5 -NS5).
- **La clave del problema:** Su función depende de su estructura 3D, no de la secuencia exacta de nucleótidos.
- **El desafío:** Las secuencias de ARN que contienen SLA tienen diferentes longitudes ('L'), que en nuestro caso varían entre 70-90 nucleótidos. Los algoritmos de clasificación clásicos requieren entradas de tamaño fijo.



Objetivo del Proyecto

Detectar si una secuencia de ARN puede adoptar esta estructura SLA es clave para el desarrollo de nuevas terapias antivirales.

Pregunta central

El Enfoque Tradicional y sus Límites

- El método clásico es la **predicción de estructura secundaria**, usando herramientas como ‘Mfold‘ o ‘ViennaRNA‘.
  - *¿Cómo funcionan?* Se basan en **métodos termodinámicos**: busca la estructura más estable minimizando la energía libre.

Pregunta central

El Enfoque Tradicional y sus Límites

- El método clásico es la **predicción de estructura secundaria**, usando herramientas como ‘Mfold‘ o ‘ViennaRNA‘.
  - *¿Cómo funcionan?* Se basan en **métodos termodinámicos**: busca la estructura más estable minimizando la energía libre.
- **Limitaciones:** Es un proceso computacionalmente costoso y con dificultades para modelar estructuras complejas.

Pregunta central

El Enfoque Tradicional y sus Límites

- El método clásico es la **predicción de estructura secundaria**, usando herramientas como ‘Mfold‘ o ‘ViennaRNA‘.
  - *¿Cómo funcionan?* Se basan en **métodos termodinámicos**: busca la estructura más estable minimizando la energía libre.
- **Limitaciones:** Es un proceso computacionalmente costoso y con dificultades para modelar estructuras complejas.

La Pregunta Central

¿Podemos determinar la presencia de **SLA** directamente desde la secuencia (estructura primaria), **sin reconstruir explícitamente** la estructura?

Aparición de modelos fundacionales

- Inspirados en el éxito de los LLMs en el lenguaje natural (ej. BERT, GPT), han surgido **Modelos Fundacionales Biológicos**.
- Se pre-entrenan en corpus masivos de secuencias de ADN, ARN o proteínas.
- **Objetivo:** Aprender representaciones numéricas (*embeddings*) que capturen la 'gramática' y la 'semántica' del lenguaje biológico.
- Estas representaciones contienen información sobre la estructura y la función de forma **implícita**.

RNA-FM (explicación técnica y por qué usarlo)

- Es un modelo de lenguaje basado en la arquitectura **Transformer-Encoder (BERT)**.
- Pre-entrenado con Modelado de Lenguaje Enmascarado (MLM) **auto-supervisado**, a partir de la base de datos **RNAcentral** (23.7 millones de secuencias de ARN no codificante).
- **Tarea:** El modelo aprende a predecir nucleótidos faltantes basándose en su contexto.
- **Resultado:** Genera un vector de **640 dimensiones** para cada nucleótido logrando representar no solo la secuencia sino la estructura de la molécula.

RNA-FM (explicación técnica y por qué usarlo)

- Es un modelo de lenguaje basado en la arquitectura **Transformer-Encoder (BERT)**.
- Pre-entrenado con Modelado de Lenguaje Enmascarado (MLM) **auto-supervisado**, a partir de la base de datos **RNAcentral** (23.7 millones de secuencias de ARN no codificante).
- **Tarea:** El modelo aprende a predecir nucleótidos faltantes basándose en su contexto.
- **Resultado:** Genera un vector de **640 dimensiones** para cada nucleótido logrando representar no solo la secuencia sino la estructura de la molécula.

¿Por qué RNA-FM para este proyecto?

Es una herramienta que nos proporciona la representación numérica con información estructural del ARN.

Idea central: de secuencia a imagen

Hipótesis del Proyecto

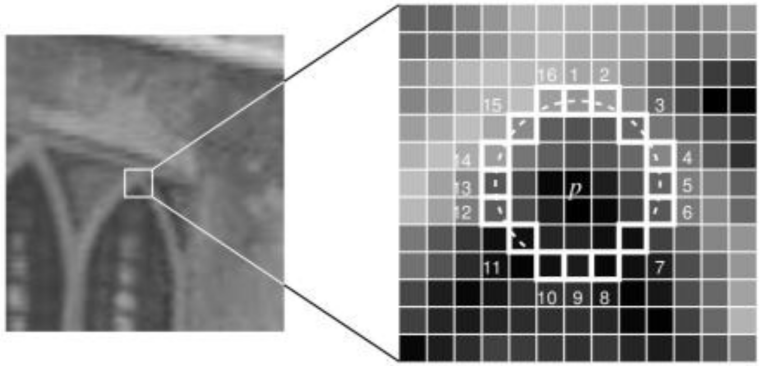
Dado que **RNA-FM** genera embeddings con información estructural implícita, al normalizar los valores resultantes podemos tratar estos embeddings de longitud variable como **imágenes** para luego aplicar una secuencia de métodos de **visión por computadora** y así extraer un vector de características de longitud fija, resolviendo el desafío técnico que representa longitud variable de las secuencias ('L') y permitiendo la clasificación con métodos tradicionales.

Detector FAST: Features from Accelerated Segment Test

Detección de 'esquinas' de alta velocidad

Mecanismo:

1. Se selecciona un píxel candidato p.
2. Se examinan los 16 píxeles en un círculo de radio 3 a su alrededor.
3. Si existen N píxeles *consecutivos* que cumplen la condición (ej. $N = 9$), p es marcado como keypoint.

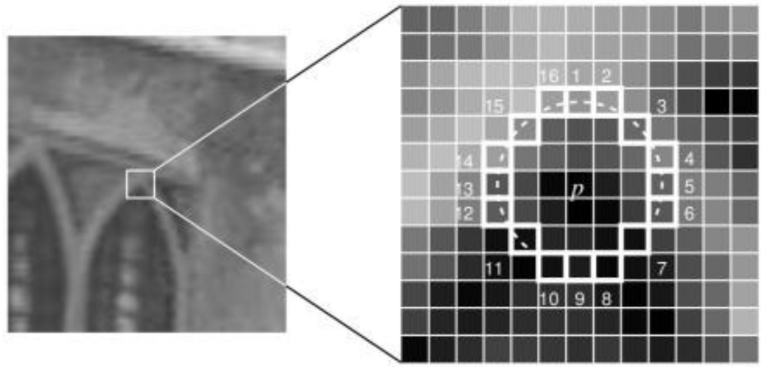


Detector FAST: Features from Accelerated Segment Test

Detección de 'esquinas' de alta velocidad

Mecanismo:

1. Se selecciona un píxel candidato p.
2. Se examinan los 16 píxeles en un círculo de radio 3 a su alrededor.
3. Si existen N píxeles *consecutivos* que cumplen la condición (ej. $N = 9$), p es marcado como keypoint.



Ventaja Principal

Es órdenes de magnitud más rápido que otros métodos, ya que solo realiza comparaciones de intensidad simples.

Herramientas (2/3): El estándar SIFT (1/2)

Paso 1: Espacio Escala Gaussiano

- Se aplica una serie de filtros Gaussianos (G_σ) con desenfoque creciente para crear diferentes .ºctavas"de imágenes.
- Dentro de cada octava, se reduce la resolución (subsampling) para construir una pirámide de imágenes.
- **Objetivo:** Detectar características de interés sin importar su tamaño (escala) en la imagen original.

Herramientas (2/3): El estándar SIFT (1/2)

Paso 1: Espacio Escala Gaussiano

- Se aplica una serie de filtros Gaussianos (G_σ) con desenfoque creciente para crear diferentes .ºctavas"de imágenes.
- Dentro de cada octava, se reduce la resolución (subsampling) para construir una pirámide de imágenes.
- **Objetivo:** Detectar características de interés sin importar su tamaño (escala) en la imagen original.

Paso 2: Detección de Keypoints (Puntos Clave)

- Se calculan las "Diferencias de Gaussianas"(DoG) restando imágenes consecutivas en la pirámide de escala.
- Los keypoints se localizan como máximos/mínimos locales en esta pirámide DoG, comparando cada píxel con sus 26 vecinos (en el espacio 3D).
- **Objetivo:** Encontrar puntos estables y distintivos que puedan ser localizados de forma fiable en diferentes escalas.

Herramientas (2/3): El estándar SIFT (2/2)

Paso 3: Asignación de Orientación

- Para cada keypoint, se calcula la dirección y magnitud del gradiente en una ventana a su alrededor (16x16 píxeles).
- Se crea un histograma de orientaciones (36 bins). La orientación dominante del keypoint es el pico más alto de este histograma.

Herramientas (2/3): El estándar SIFT (2/2)

Paso 3: Asignación de Orientación

- Para cada keypoint, se calcula la dirección y magnitud del gradiente en una ventana a su alrededor (16x16 píxeles).
- Se crea un histograma de orientaciones (36 bins). La orientación dominante del keypoint es el pico más alto de este histograma.

Paso 4: Creación del Descriptor

- Se toma una ventana de 16x16 alrededor del keypoint y se rota según su orientación dominante.
- Esta ventana se divide en una cuadrícula de 4x4 subregiones. Para cada una, se crea un histograma de 8 orientaciones.
- Se concatenan los 16 histogramas para formar un vector de 128 dimensiones (16 regiones $\times$ 8 orientaciones), su "huella digital".

Herramientas (2/3): El estándar SIFT (2/2)

Paso 3: Asignación de Orientación

- Para cada keypoint, se calcula la dirección y magnitud del gradiente en una ventana a su alrededor (16x16 píxeles).
- Se crea un histograma de orientaciones (36 bins). La orientación dominante del keypoint es el pico más alto de este histograma.

Paso 4: Creación del Descriptor

- Se toma una ventana de 16x16 alrededor del keypoint y se rota según su orientación dominante.
- Esta ventana se divide en una cuadrícula de 4x4 subregiones. Para cada una, se crea un histograma de 8 orientaciones.
- Se concatenan los 16 histogramas para formar un vector de 128 dimensiones (16 regiones $\times$ 8 orientaciones), su "huella digital".

Conclusión

SIFT es extremadamente **robusto** pero computacionalmente **intensivo**. Su resultado es un conjunto de descriptores invariantes a escala, rotación, traslación y cambios de iluminación.

BRIEF: Binary Robust Independent Elementary Features

Identificador Binario

Para esto se construye un vector de bits basado en simples comparaciones de intensidad entre pares de píxeles.

Mecanismo:

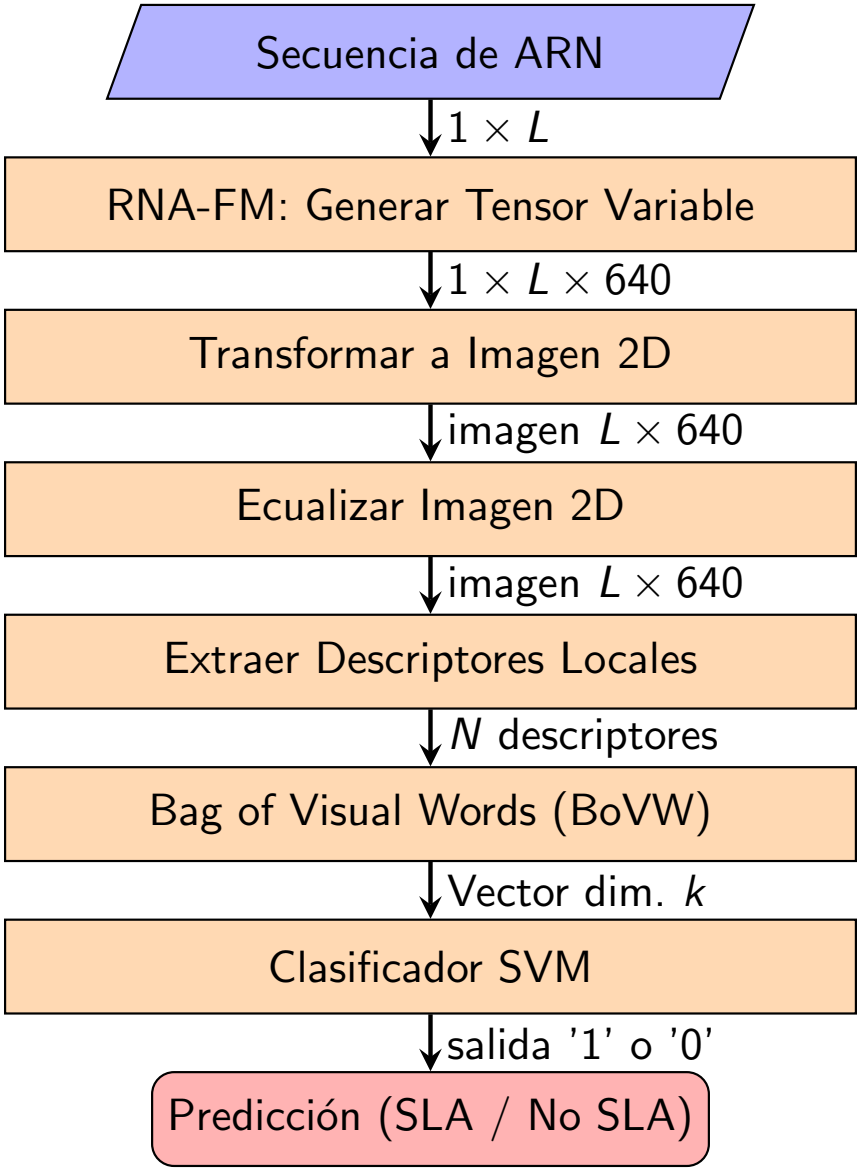
1. Se aplica un filtro Gaussiano a la imagen para reducir el ruido.
2. Se define un conjunto de n_d pares de píxeles (p_1, p_2) en una ventana alrededor del keypoint.
3. Para cada par, se realiza el test:

$$\tau(p_1, p_2) := \begin{cases} 1 & \text{si } I(p_1) < I(p_2), \\ 0 & \text{en otro caso.} \end{cases}$$

4. El descriptor es el vector binario de n_d bits resultante (por ejemplo, 256 bits).

(+) El almacenamiento y la comparación (distancia de Hamming) son extremadamente rápidos.
(-) No es inherentemente invariante a la rotación (aunque en nuestro caso no es un problema).

Diagrama de la secuencia de métodos



Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

---

2. Embedding: Del RNA-FM obtenemos una matriz de reales ($L \times 640$).

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

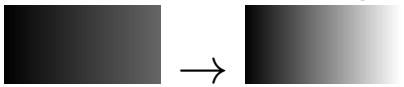
---

2. Embedding: Del RNA-FM obtenemos una matriz de reales ($L \times 640$).

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

---

3. Imagen + Ecualización: Los valores de la matriz se normalizan entre 0 y 255 para obtener una imagen en escala de grises que luego se ecualiza.



Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

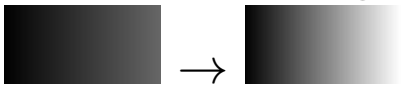
---

2. Embedding: Del RNA-FM obtenemos una matriz de reales ($L \times 640$).

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

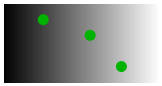
---

3. Imagen + Ecualización: Los valores de la matriz se normalizan entre 0 y 255 para obtener una imagen en escala de grises que luego se ecualiza.



---

4. Descriptores: Se extraen puntos clave (keypoints) y se generan descriptores.



Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

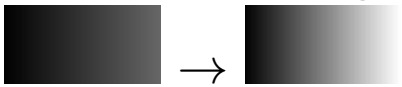
---

2. Embedding: Del RNA-FM obtenemos una matriz de reales ($L \times 640$).

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

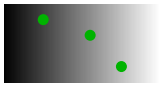
---

3. Imagen + Ecualización: Los valores de la matriz se normalizan entre 0 y 255 para obtener una imagen en escala de grises que luego se ecualiza.



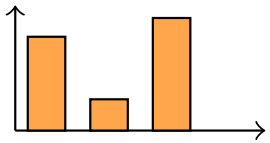
---

4. Descriptores: Se extraen puntos clave (keypoints) y se generan descriptores.



---

5. Vector Fijo: Se crea un vector-histograma con BoVW.



Un Ejemplo Paso a Paso

1. Entrada: Una secuencia de ARN ($1 \times L$).

`A G U C A G U...`

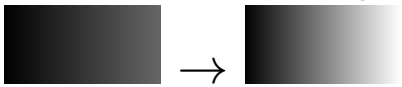
---

2. Embedding: Del RNA-FM obtenemos una matriz de reales ($L \times 640$).

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

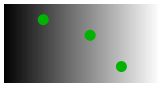
---

3. Imagen + Ecualización: Los valores de la matriz se normalizan entre 0 y 255 para obtener una imagen en escala de grises que luego se ecualiza.



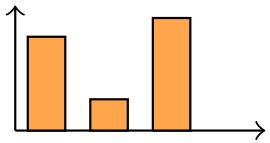
---

4. Descriptores: Se extraen puntos clave (keypoints) y se generan descriptores.



---

5. Vector Fijo: Se crea un vector-histograma con BoVW.



---

6. Predicción: El clasificador SVM da un resultado (SLA/no-SLA).

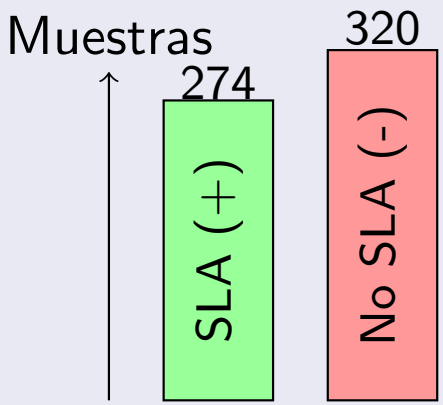
Nuestro Dataset en Detalle

1. Características Generales

- **Origen:** Secuencias de ARN de flavivirus, compiladas y etiquetadas por el laboratorio de virología de la UNAHUR.
- **Dataset Final: 594 secuencias únicas** tras el proceso de limpieza.
- **División (Split):** Se utilizó un 80 % (475) para entrenamiento y un 20 % (119) para prueba.

2. Distribución de Clases

- 274 Positivas (SLA)
- 320 Negativas (No SLA)



El dataset está razonablemente balanceado.

3. Longitud de las Secuencias ('L')

La longitud es muy variable, un desafío clave.

- **Secuencias Positivas:** Longitud entre 61 y 76 nucleótidos.
- **Secuencias Negativas:** Longitud entre 63 y 97 nucleótidos.

Para prestar atención

Esta variabilidad refuerza la necesidad de un método que genere vectores de longitud fija.

Paso 1: Generación de embeddings con RNA-FM

Todo comienza con una secuencia de nucleótidos.

`GUGUCAGUAUGAACAGGUAGAAAAAAGGACAGGU...`

Cada secuencia de longitud ‘L‘ se convierte con **RNA-FM** en un tensor de ‘1 x L x 640‘.

- ‘1‘: Lote de una secuencia.
- ‘L‘: Longitud variable de la secuencia.
- ‘640‘: Dimensiones del embedding por cada nucleótido.

Paso 1: Generación de embeddings con RNA-FM

Todo comienza con una secuencia de nucleótidos.

`GUGUCAGUAUGAACAGGUAGAAAAAAGGACAGGU...`

Cada secuencia de longitud ‘L‘ se convierte con **RNA-FM** en un tensor de ‘1 x L x 640‘.

- ‘1‘: Lote de una secuencia.
- ‘L‘: Longitud variable de la secuencia.
- ‘640‘: Dimensiones del embedding por cada nucleótido.

Conversión a Matriz Visual

- El tensor se reinterpreta como una matriz de ‘L x 640‘ de números reales.

$$\begin{pmatrix} 0{,}12 & -0{,}45 & \dots & 0{,}88 \\ -0{,}91 & 0{,}03 & \dots & -0{,}21 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

Paso 2: Conversión a Imagen (1/2)

La matriz obtenida en el paso anterior se normaliza con valores entre 0 y 255 y se visualiza como una imagen.

Paso 2: Conversión a Imagen (1/2)

La matriz obtenida en el paso anterior se normaliza con valores entre 0 y 255 y se visualiza como una imagen.

Dicha normalización se efectuó de dos formas

- **Normalización Local:** Se lleva a cabo tomando el mínimo y máximo de cada imagen
- **Normalización Global:** Se realizó tomando el valor máximo y mínimo de todos los embeddings de los datos de entrenamiento generados por RNA-FM.

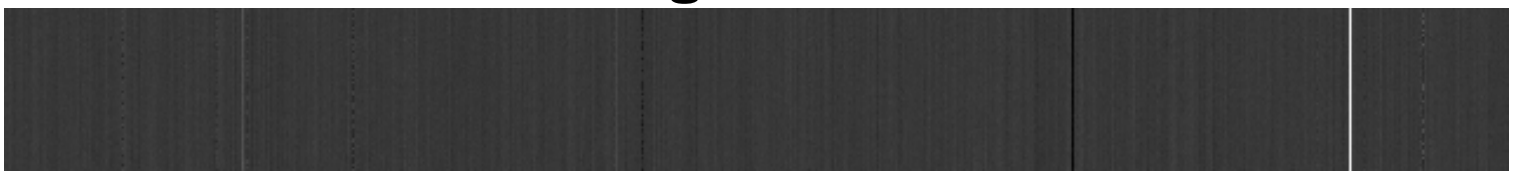
Paso 2: Conversión a Imagen (1/2)

La matriz obtenida en el paso anterior se normaliza con valores entre 0 y 255 y se visualiza como una imagen.

Dicha normalización se efectuó de dos formas

- **Normalización Local:** Se lleva a cabo tomando el mínimo y máximo de cada imagen
- **Normalización Global:** Se realizó tomando el valor máximo y mínimo de todos los embeddings de los datos de entrenamiento generados por RNA-FM.

Imagen Inicial



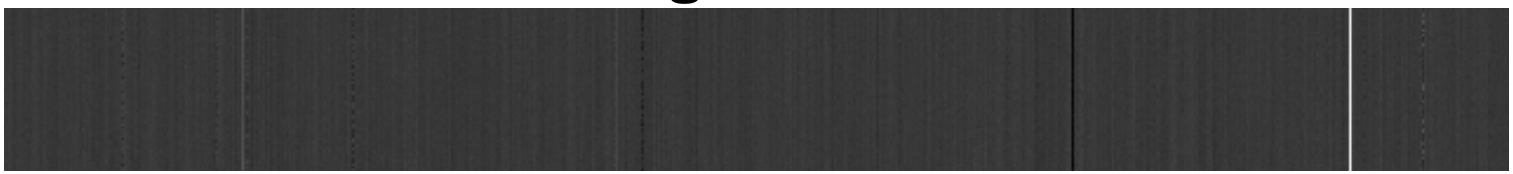
Paso 2: Conversión a Imagen (1/2)

La matriz obtenida en el paso anterior se normaliza con valores entre 0 y 255 y se visualiza como una imagen.

Dicha normalización se efectuó de dos formas

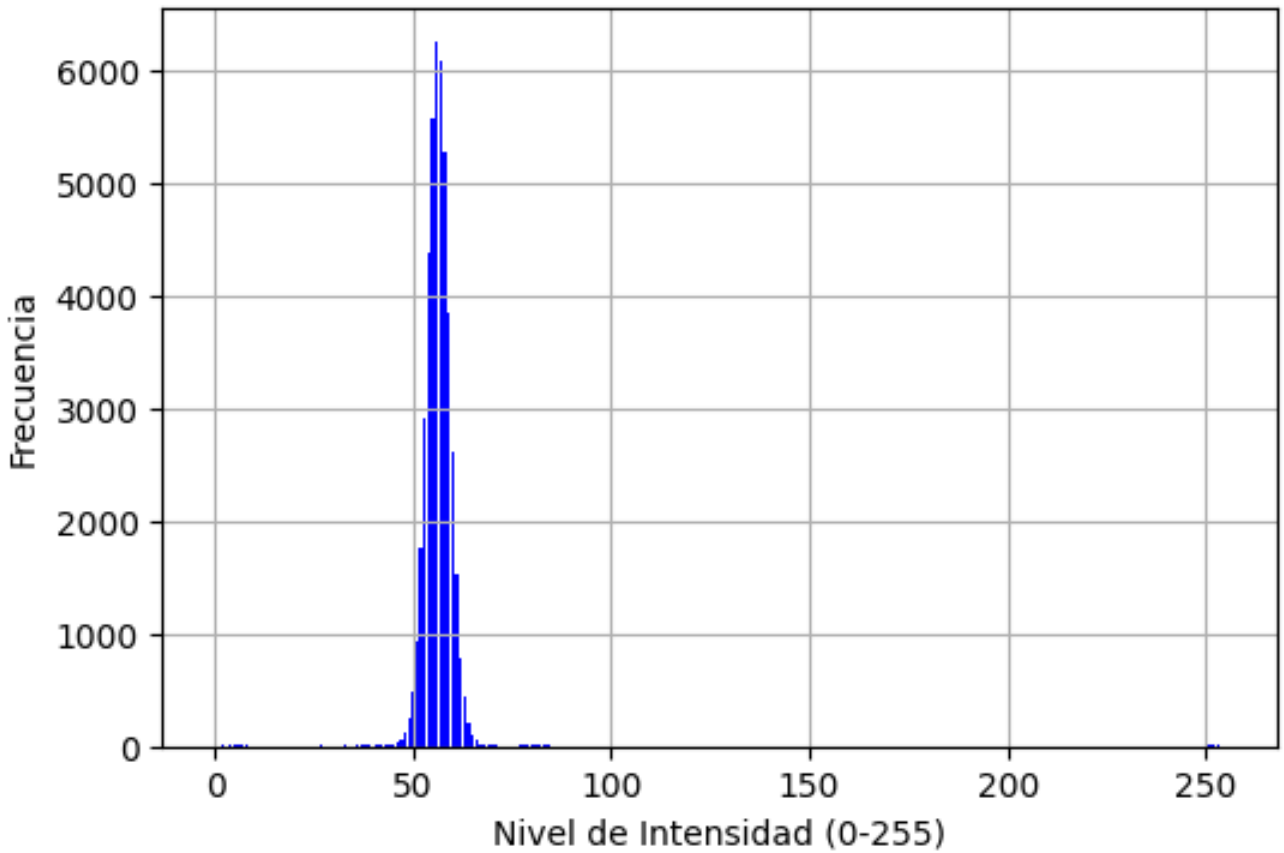
- **Normalización Local:** Se lleva a cabo tomando el mínimo y máximo de cada imagen
- **Normalización Global:** Se realizó tomando el valor máximo y mínimo de todos los embeddings de los datos de entrenamiento generados por RNA-FM.

Imagen Inicial



Problema: Bajo contraste.

Paso 2: Conversión a Imagen (2/2)



Histograma de intensidades concentradas

Problema: La mayoría de los píxeles tienen intensidades bajas, concentradas en un rango estrecho $\Rightarrow$ imagen apagada.

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

1. Cálculo de la probabilidad de intensidad:

$$p(r_j) = \frac{n_j}{n};$$ Donde:

- n_j: número de píxeles con intensidad r_j
- n: número total de píxeles en la imagen

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

1. Cálculo de la probabilidad de intensidad:

$$p(r_j) = \frac{n_j}{n};$$ Donde:

- n_j: número de píxeles con intensidad r_j
- n: número total de píxeles en la imagen

$\Rightarrow$ Esto representa una **frecuencia relativa**, base del histograma normalizado.

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

1. Cálculo de la probabilidad de intensidad:

$$p(r_j) = \frac{n_j}{n};$$ Donde:

- n_j: número de píxeles con intensidad r_j
- n: número total de píxeles en la imagen

$\Rightarrow$ Esto representa una **frecuencia relativa**, base del histograma normalizado.

2. Cálculo de la función de distribución acumulada (CDF):

$$s_i = \sum_{j=0}^{i} p(r_j)$$

La CDF indica la acumulación progresiva de probabilidades hasta el nivel i.

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

1. Cálculo de la probabilidad de intensidad:

$$p(r_j) = \frac{n_j}{n};$$ Donde:

- n_j: número de píxeles con intensidad r_j
- n: número total de píxeles en la imagen

$\Rightarrow$ Esto representa una **frecuencia relativa**, base del histograma normalizado.

2. Cálculo de la función de distribución acumulada (CDF):

$$s_i = \sum_{j=0}^{i} p(r_j)$$

La CDF indica la acumulación progresiva de probabilidades hasta el nivel i.

3. Transformación del nivel de gris:

$$s_i' = (L-1) \cdot s_i$$

Con $L = 256$, se reescala al rango completo de niveles [0–255].

Paso 3: Ecualización del histograma (1/2)

¿Qué es la ecualización del histograma?

- **Objetivo:** Redistribuir los niveles de gris para mejorar el contraste.

1. Cálculo de la probabilidad de intensidad:

$$p(r_j) = \frac{n_j}{n};$$ Donde:

- n_j: número de píxeles con intensidad r_j
- n: número total de píxeles en la imagen

$\Rightarrow$ Esto representa una **frecuencia relativa**, base del histograma normalizado.

2. Cálculo de la función de distribución acumulada (CDF):

$$s_i = \sum_{j=0}^{i} p(r_j)$$

La CDF indica la acumulación progresiva de probabilidades hasta el nivel i.

3. Transformación del nivel de gris:

$$s_i' = (L-1) \cdot s_i$$

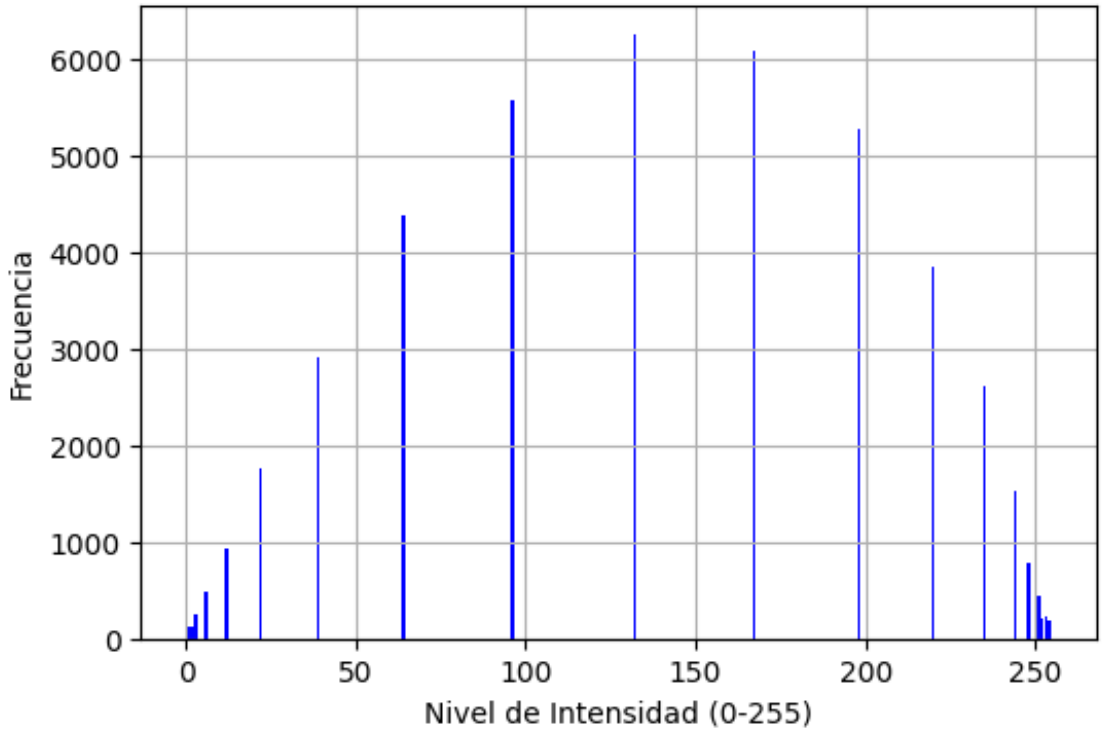
Con $L = 256$, se reescala al rango completo de niveles [0–255].

4. Resultado: Niveles de gris más dispersos $\Rightarrow$ mayor contraste y detalles visibles.

Paso 3: Ecualización del histograma (2/2)

Imagen después de ecualizar

Mejora significativa en el contraste

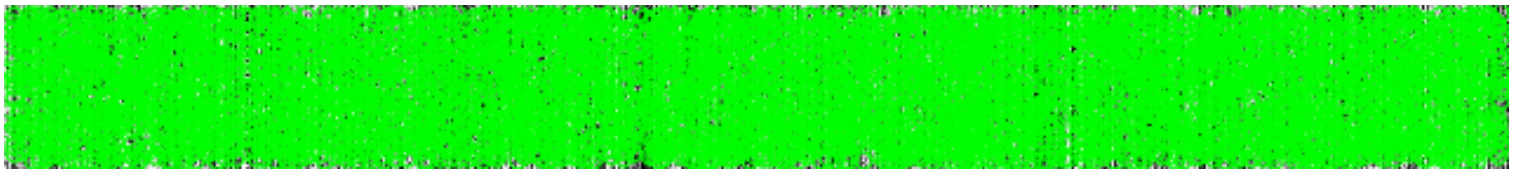


Histograma redistribuido en todo el rango [0, 255]

Paso 4: Detección y Descripción de Características

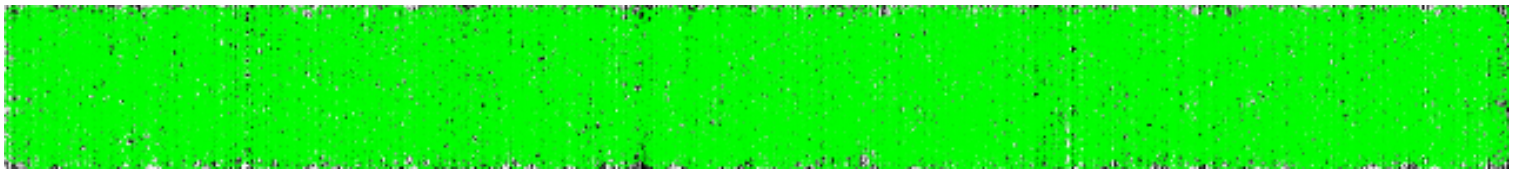
1. Detección de Puntos Clave

- Un **detector** examina la imagen en busca de 'esquinas' o regiones con alto contenido informativo como se muestra en la figura, en este caso, el detector FAST genera 3620 keypoints que se muestran en verde.
- Genera un conjunto de coordenadas (x, y).



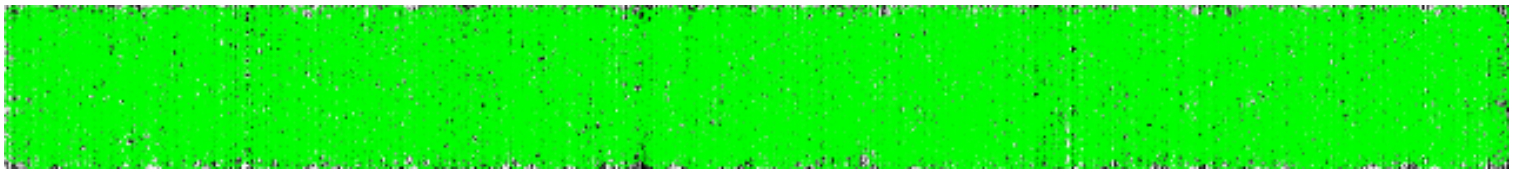
Paso 4: Detección y Descripción de Características

1. **Detección de Puntos Clave**
   - Un **detector** examina la imagen en busca de 'esquinas' o regiones con alto contenido informativo como se muestra en la figura, en este caso, el detector FAST genera 3620 keypoints que se muestran en verde.
   - Genera un conjunto de coordenadas (x, y).
2. **Descripción de Características**
   - Para cada punto, un **descriptor** crea una 'firma' numérica (vector) cuya longitud depende del método empleado y que captura la apariencia local del respectivo keypoint.



Paso 4: Detección y Descripción de Características

1. **Detección de Puntos Clave**
   - Un **detector** examina la imagen en busca de 'esquinas' o regiones con alto contenido informativo como se muestra en la figura, en este caso, el detector FAST genera 3620 keypoints que se muestran en verde.
   - Genera un conjunto de coordenadas (x, y).
2. **Descripción de Características**
   - Para cada punto, un **descriptor** crea una 'firma' numérica (vector) cuya longitud depende del método empleado y que captura la apariencia local del respectivo keypoint.



Nuevo Desafío

Cada imagen ahora tiene un **número variable** de descriptores. ¿Cómo podemos usar esto como entrada para un clasificador que requiere un vector de tamaño fijo?

Paso 5: La Solución - Bag of Visual Words (BoVW)

El Problema a Resolver

Cada imagen tiene un número **variable** de descriptores. Necesitamos una representación de **longitud fija (k)** para el clasificador.

Paso 5: La Solución - Bag of Visual Words (BoVW)

El Problema a Resolver

Cada imagen tiene un número **variable** de descriptores. Necesitamos una representación de **longitud fija (k)** para el clasificador.

La Analogía: 'Bolsa de Palabras' en Texto

- Imagina analizar un documento.
- No importa el orden de las palabras, solo **cuántas veces** aparece cada una.
- **Diccionario**: ['ARN', 'virus', ...]
- **Vector**: [5 veces, 3 veces, ...]

Nuestra Adaptación: 'Bolsa de Palabras Visuales'

- Nuestras 'palabras' son las características locales.
- Crearemos un **diccionario** de estas 'palabras visuales'.
- El resultado será un **histograma** de frecuencias.

Paso 5: La Solución - Bag of Visual Words (BoVW)

El Problema a Resolver

Cada imagen tiene un número **variable** de descriptores. Necesitamos una representación de **longitud fija (k)** para el clasificador.

La Analogía: 'Bolsa de Palabras' en Texto

- Imagina analizar un documento.
- No importa el orden de las palabras, solo **cuántas veces** aparece cada una.
- **Diccionario**: ['ARN', 'virus', ...]
- **Vector**: [5 veces, 3 veces, ...]

Nuestra Adaptación: 'Bolsa de Palabras Visuales'

- Nuestras 'palabras' son las características locales.
- Crearemos un **diccionario** de estas 'palabras visuales'.
- El resultado será un **histograma** de frecuencias.

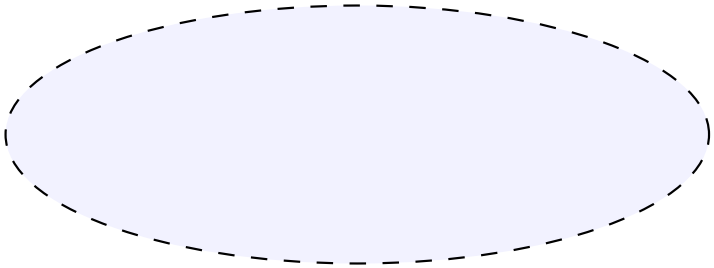
El Proceso Completo tiene dos fases:

Fase 1 (Entrenamiento): Construir el vocabulario visual.
Fase 2 (Inferencia): Codificar una nueva imagen usando ese vocabulario.

BoVW - Fase 1: Construcción del Vocabulario (Entrenamiento)

Imágenes de Entrenamiento

Img 1 | Img 2 | Img 3 | ...

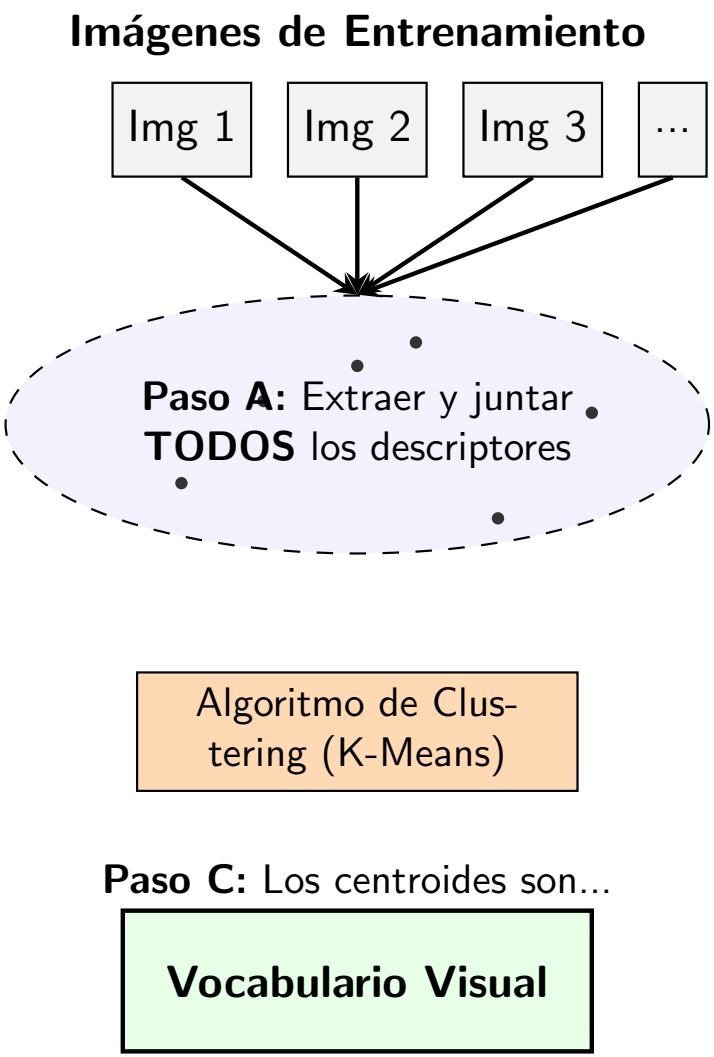


Algoritmo de Clustering (K-Means)

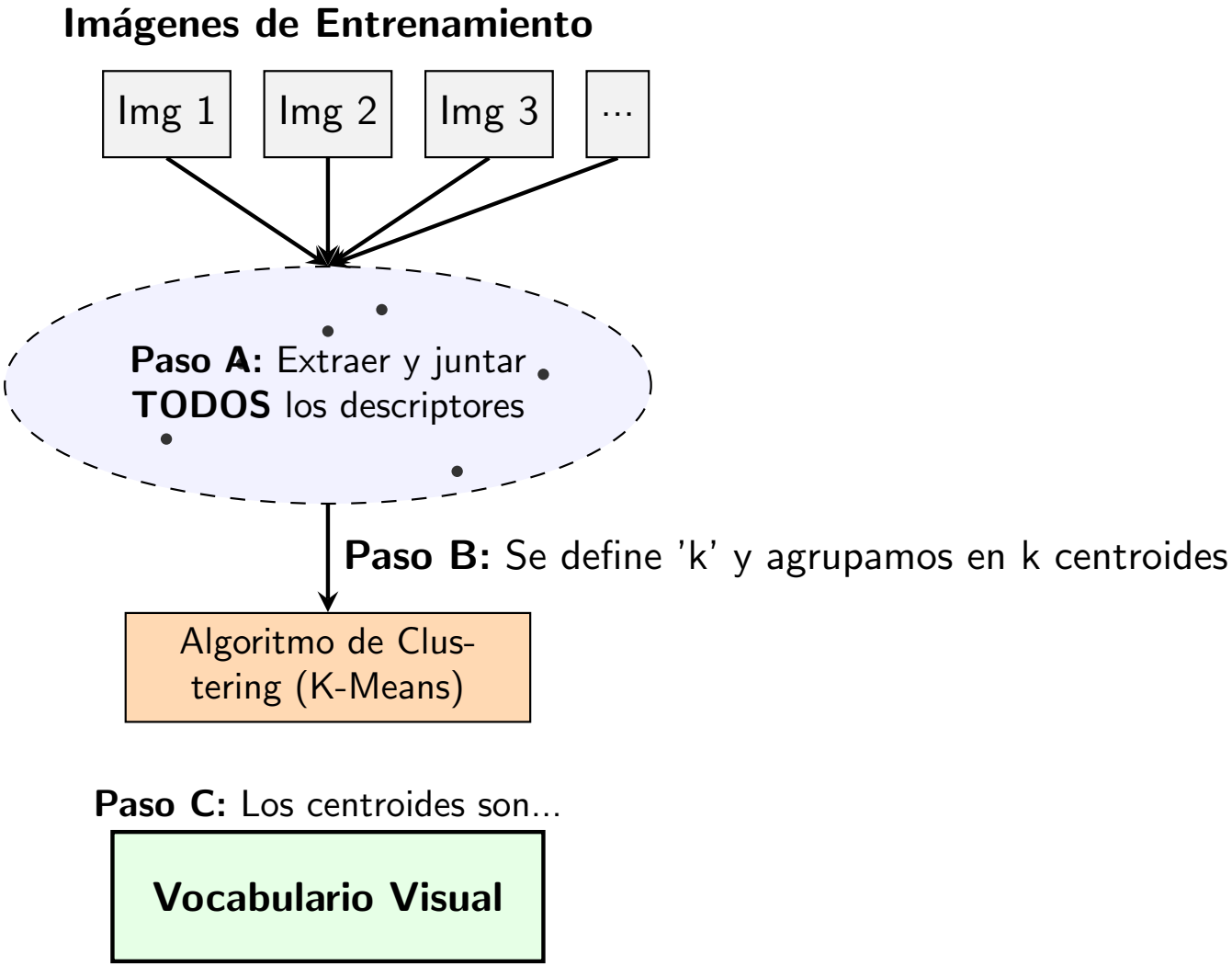
Paso C: Los centroides son...

Vocabulario Visual

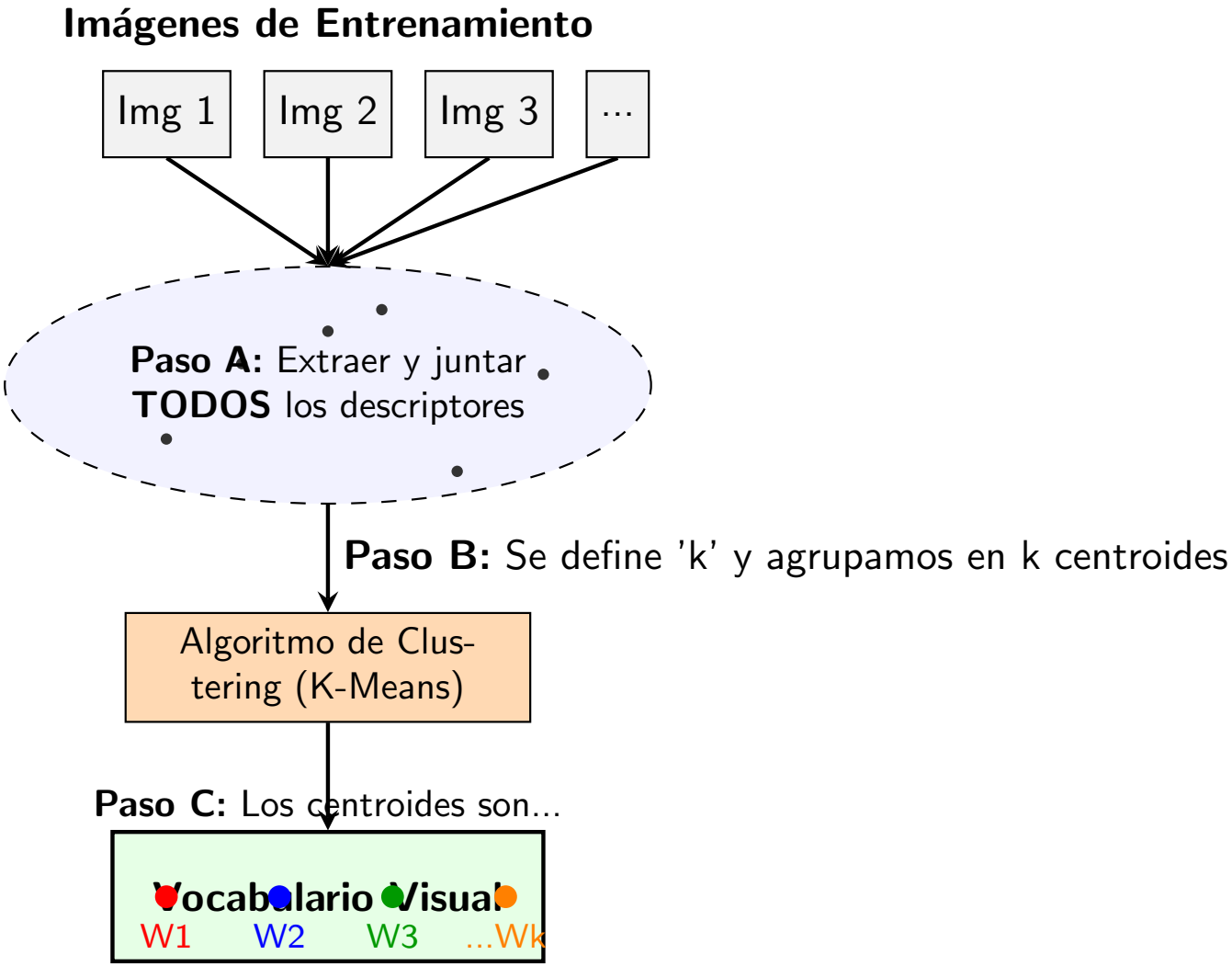
BoVW - Fase 1: Construcción del Vocabulario (Entrenamiento)



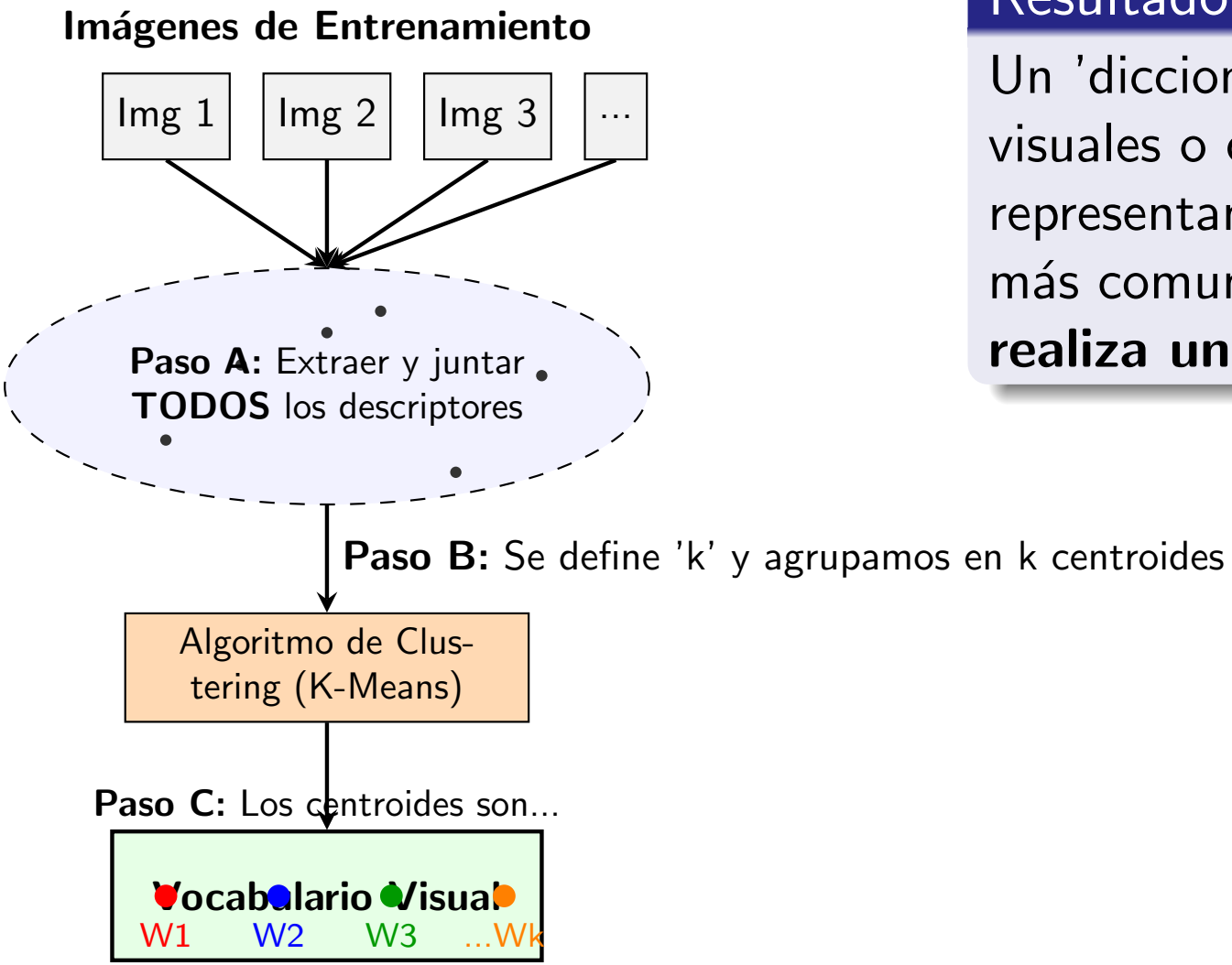
BoVW - Fase 1: Construcción del Vocabulario (Entrenamiento)



BoVW - Fase 1: Construcción del Vocabulario (Entrenamiento)



BoVW - Fase 1: Construcción del Vocabulario (Entrenamiento)



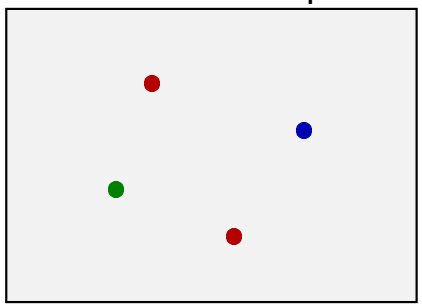
Resultado de la Fase 1

Un 'diccionario' con k palabras visuales o centroides que representan las características más comunes. **Este proceso se realiza una sola vez.**

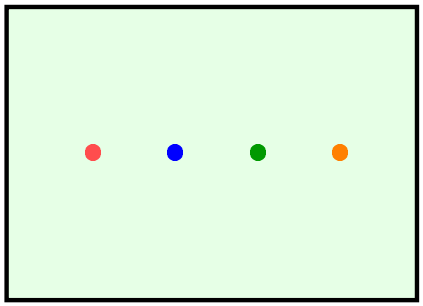
BoVW - Fase 2: Codificación de una Nueva Imagen (Inferencia)

Nueva Imagen

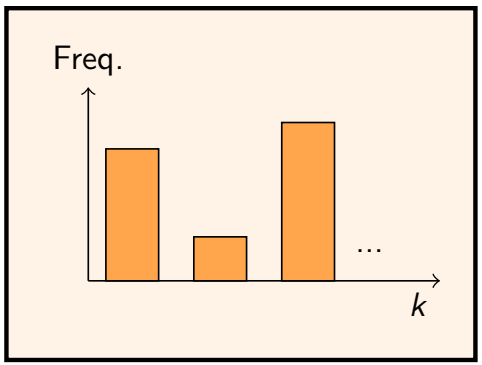
A: Extraer descriptores



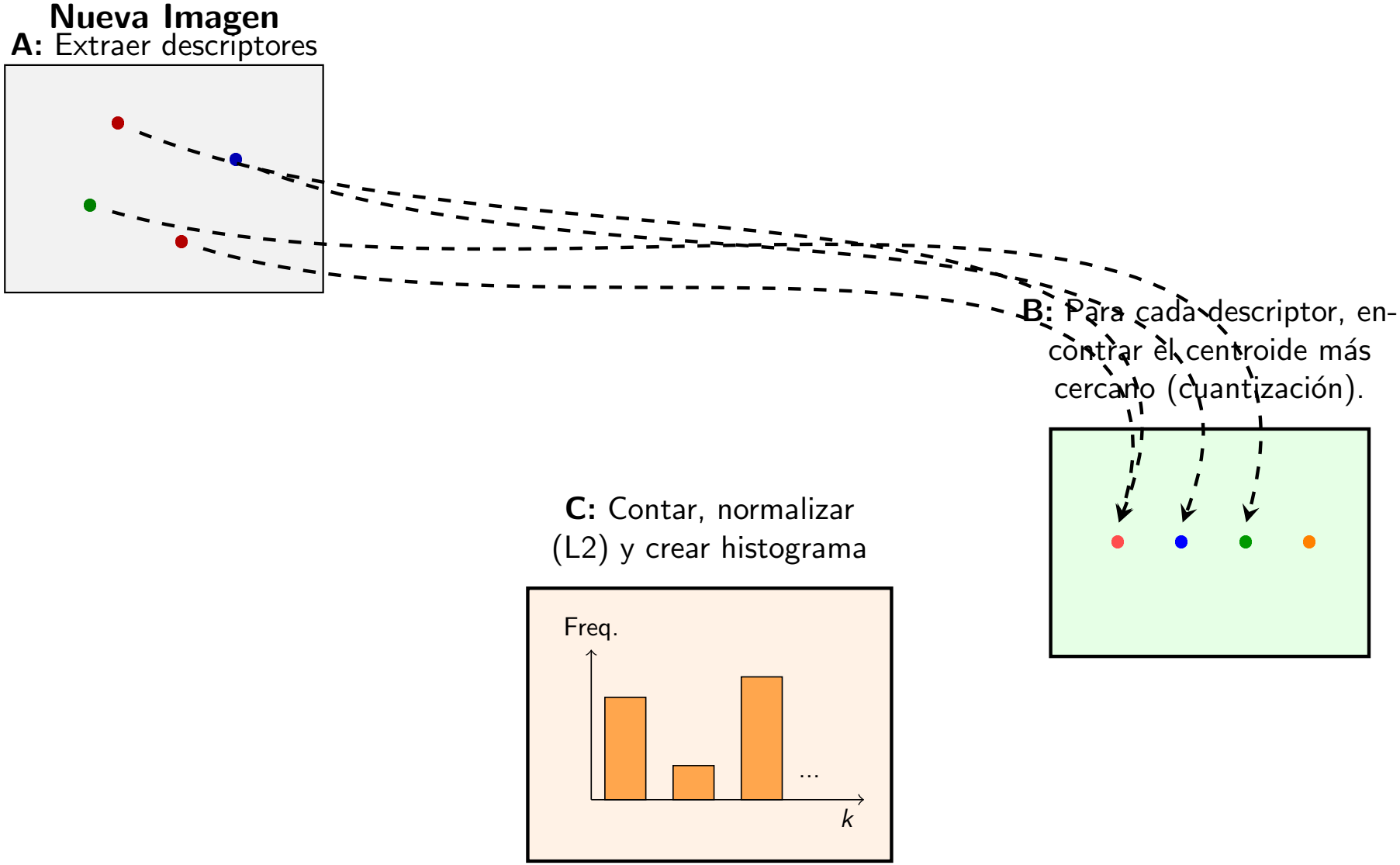
B: Para cada descriptor, encontrar el centroide más cercano (cuantización).



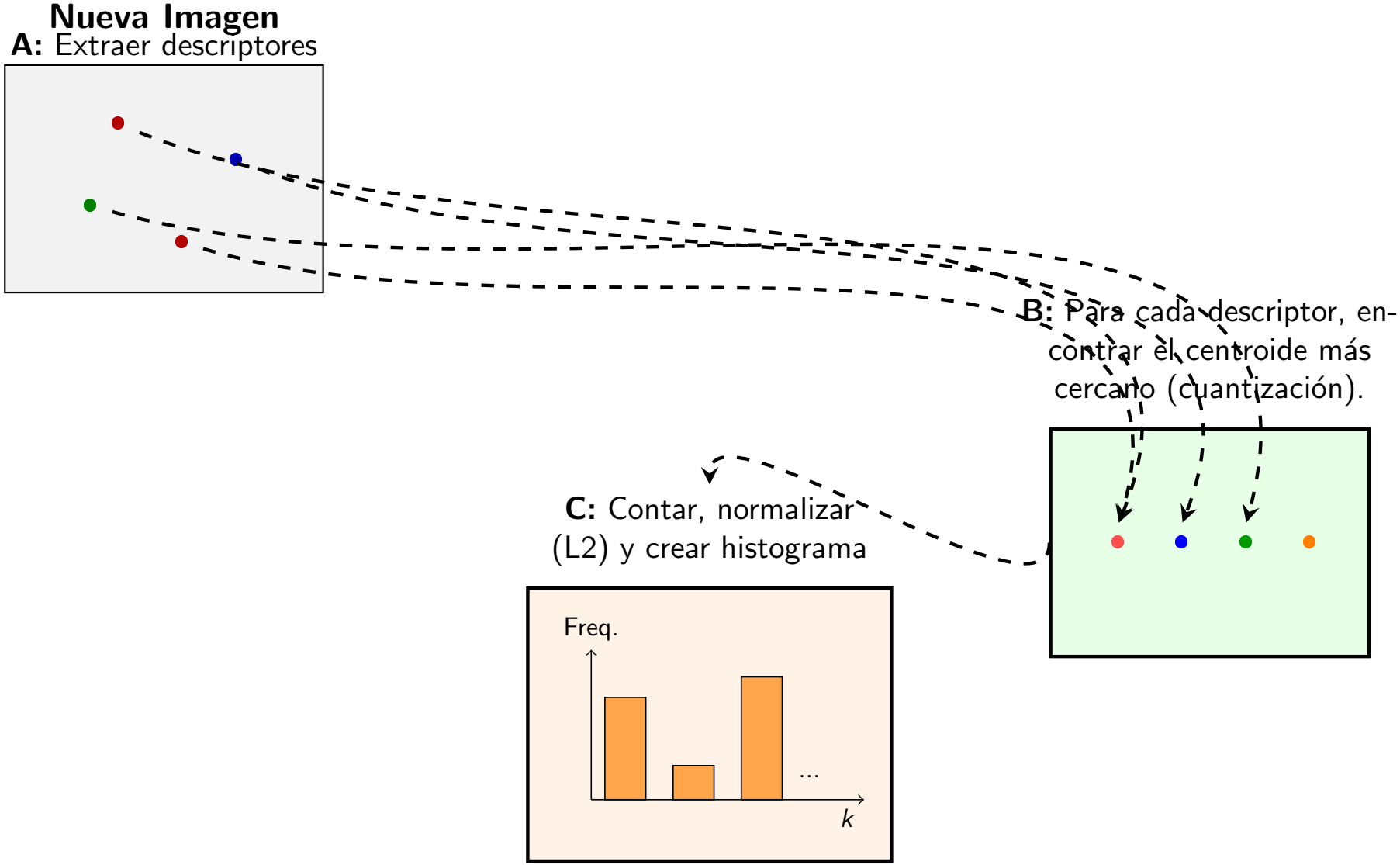
C: Contar, normalizar (L2) y crear histograma



BoVW - Fase 2: Codificación de una Nueva Imagen (Inferencia)

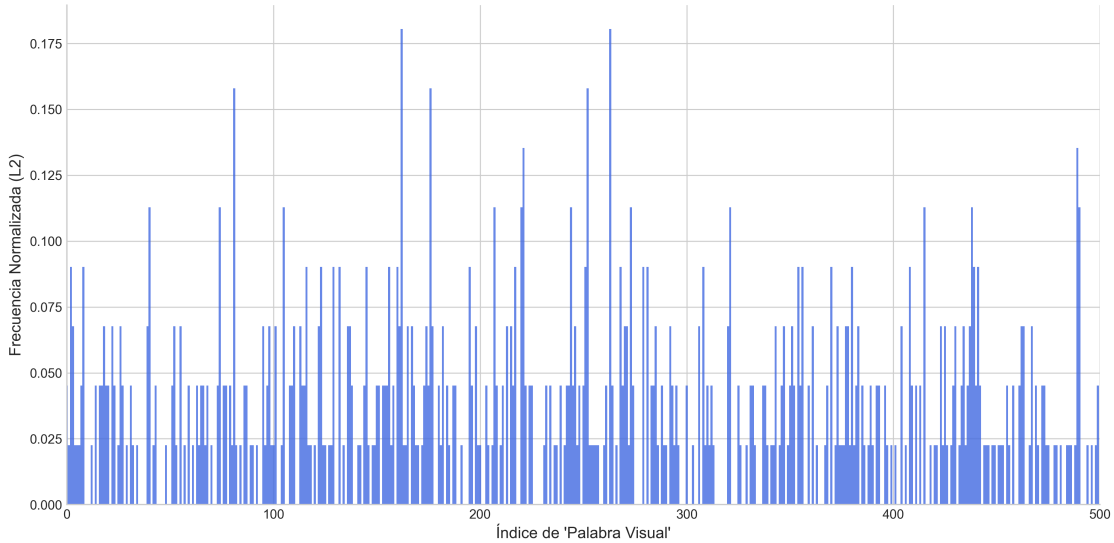


BoVW - Fase 2: Codificación de una Nueva Imagen (Inferencia)



BoVW - El Resultado Final: Un Vector de Características

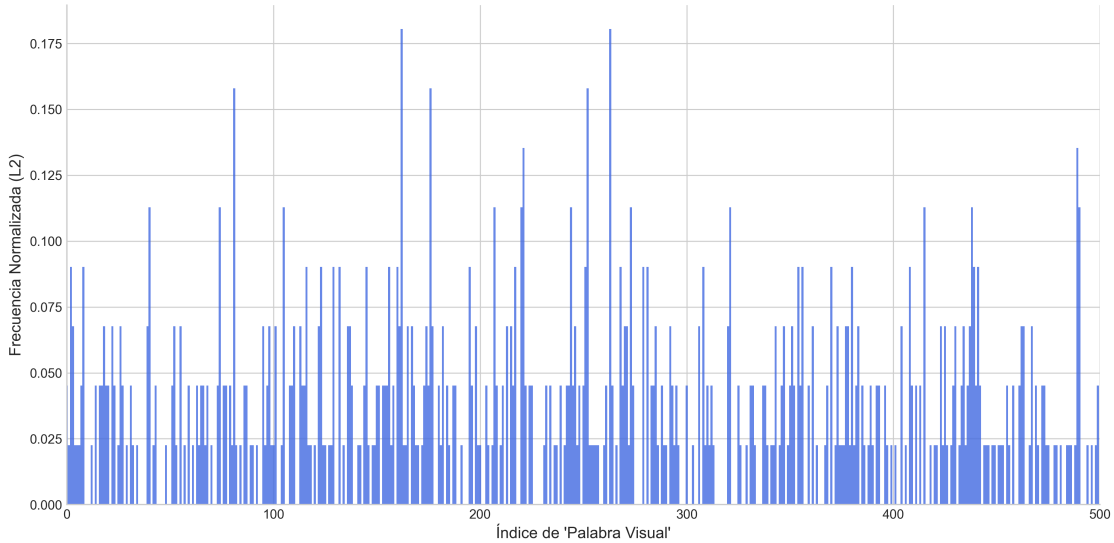
El histograma es la representación final de la imagen



- **Longitud Fija:** Este vector siempre tiene la misma longitud (k=500), sin importar el tamaño de la imagen original ni la cantidad de descriptores que surjan de la misma.
- **Informativo:** La distribución de las barras captura la composición de la imagen.
- **Listo para Clasificar:** Este es el vector que se le entrega al clasificador SVM.

BoVW - El Resultado Final: Un Vector de Características

El histograma es la representación final de la imagen



- **Longitud Fija:** Este vector siempre tiene la misma longitud (k=500), sin importar el tamaño de la imagen original ni la cantidad de descriptores que surjan de la misma.
- **Informativo:** La distribución de las barras captura la composición de la imagen.
- **Listo para Clasificar:** Este es el vector que se le entrega al clasificador SVM.

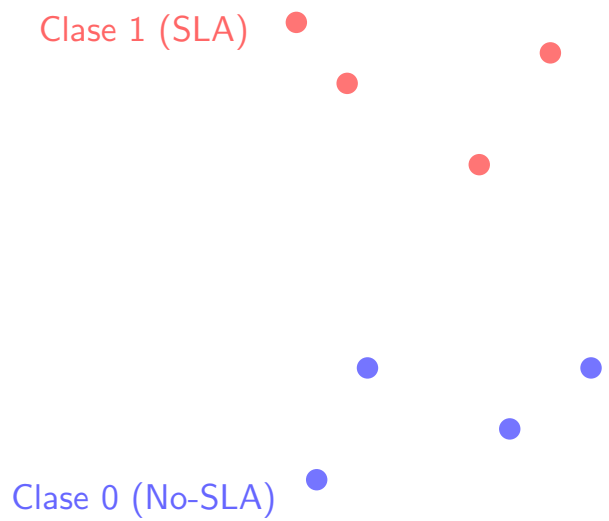
Conclución

BoVW ha transformado con éxito un problema de **datos no estructurados de longitud variable** en un problema de **clasificación de vectores de longitud fija**.

Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

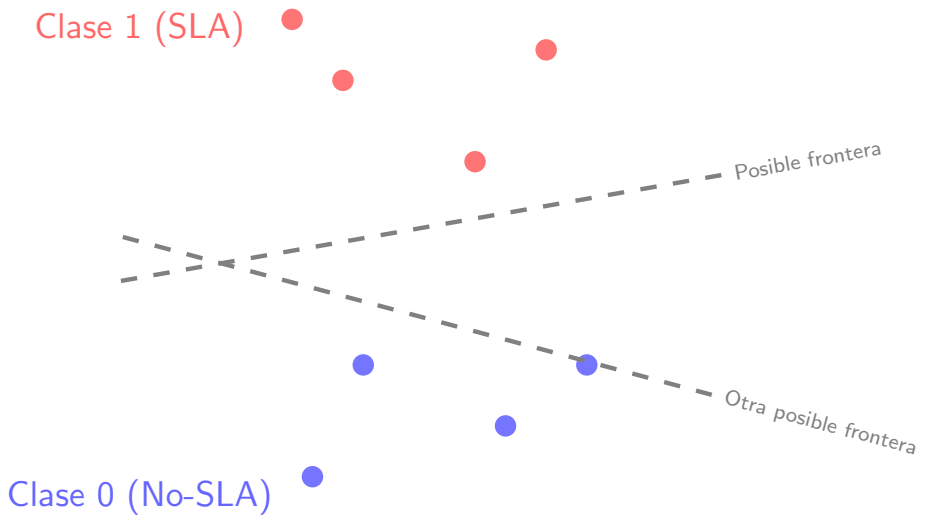
Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

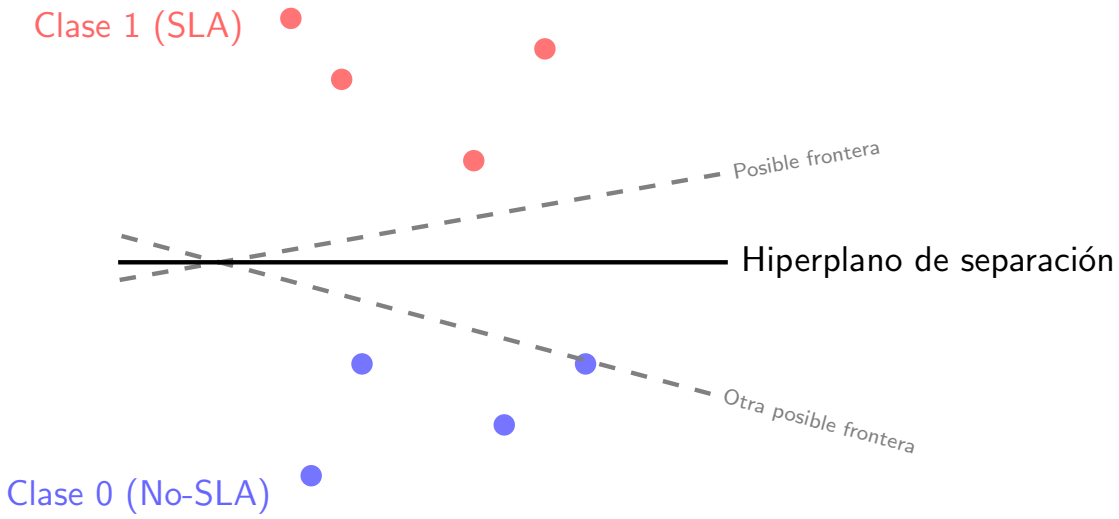
Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

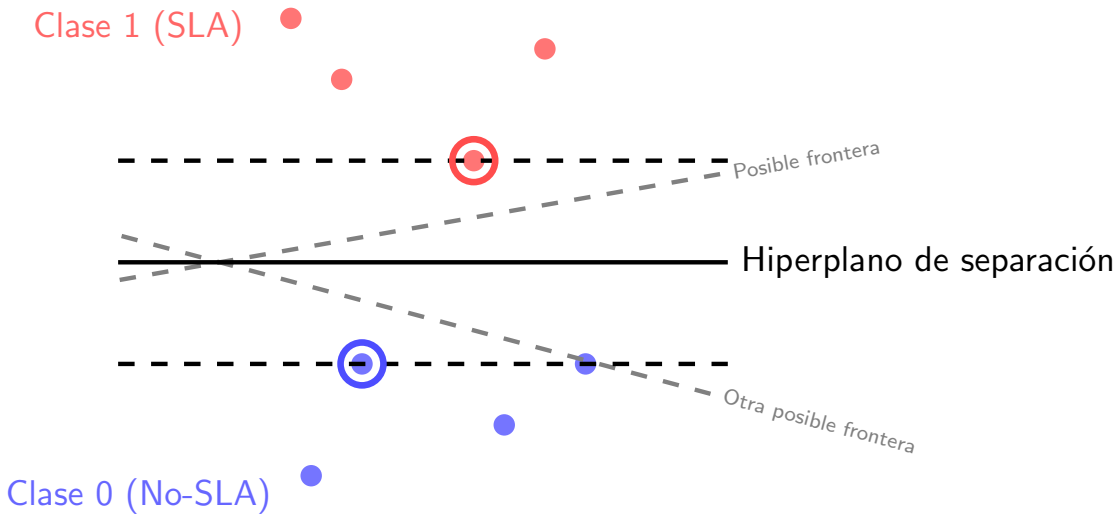
Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

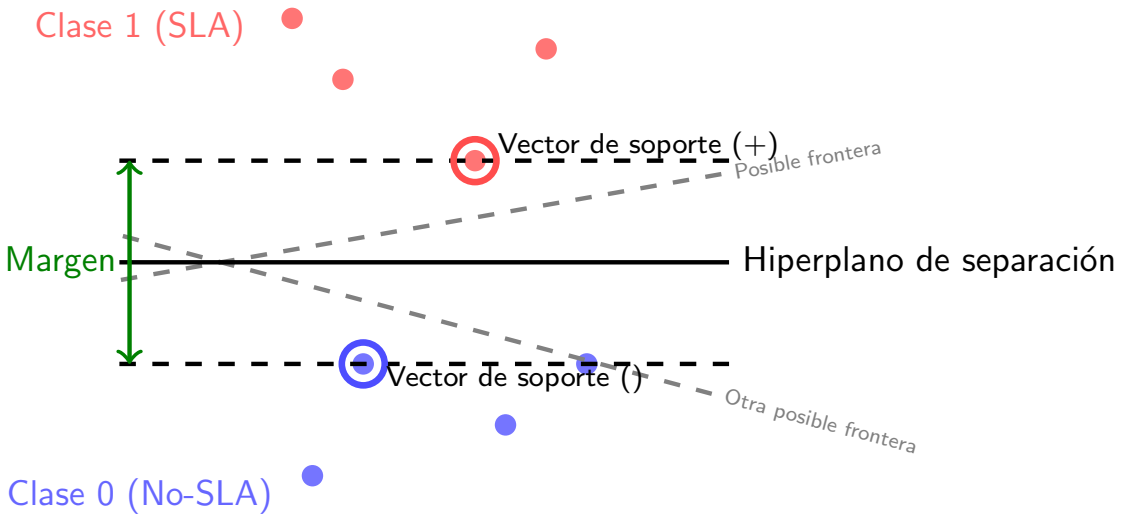
Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

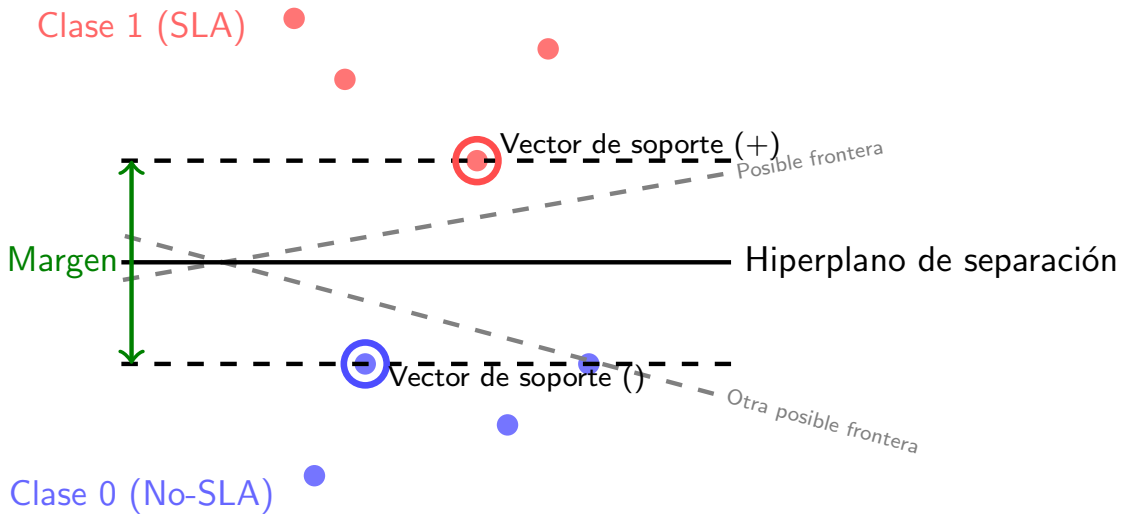
Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



Paso 6: Clasificación con Support Vector Machine (SVM)

Objetivo: Encontrar el Hiperplano Óptimo

Una vez que tenemos nuestro vector de características de longitud fija, necesitamos un clasificador que decida si pertenece a la clase *SLA* (1) o *No-SLA* (0). SVM busca el **hiperplano de decisión** que maximiza el margen entre las dos clases.



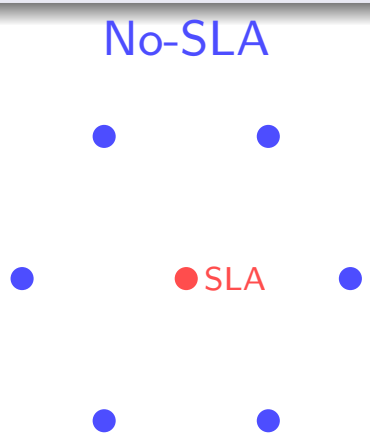
Idea clave de SVM

El **hiperplano óptimo** es el que maximiza la distancia (margen) entre las rectas paralelas que pasan por los vectores de soporte; esos vectores son exactamente los puntos más cercanos al hiperplano en el caso lineal separable.

SVM: El 'Truco del Kernel' para Datos no Lineales

El Problema

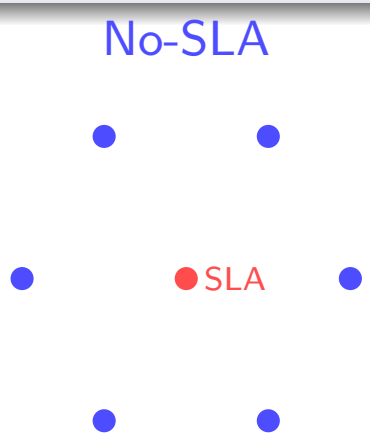
¿Qué pasa si los datos no se pueden separar con una línea recta?



SVM: El 'Truco del Kernel' para Datos no Lineales

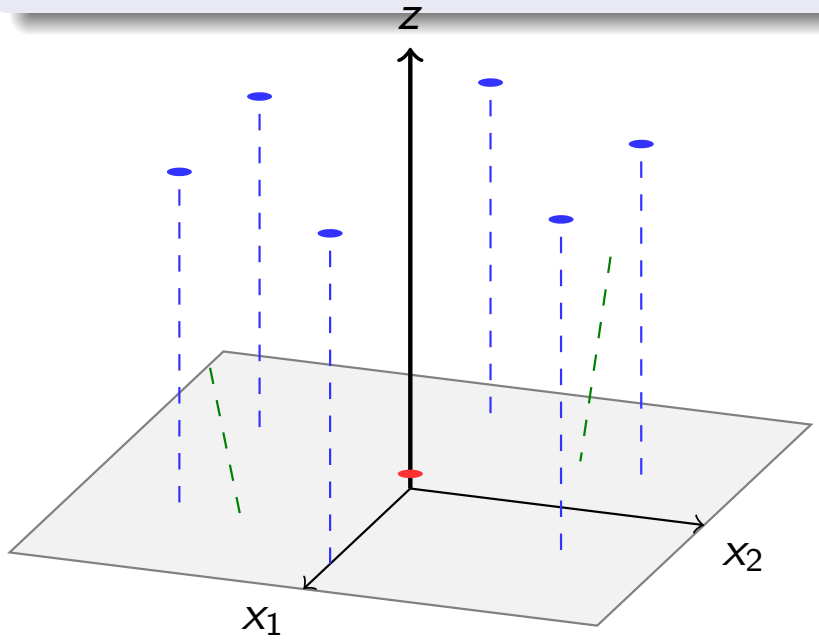
El Problema

¿Qué pasa si los datos no se pueden separar con una línea recta?



En 3D

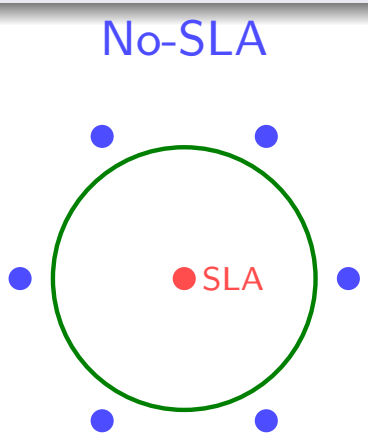
Se proyecta los datos a R^3 aplicando $\varphi(x_1, x_2) = (x_1, x_2, z)$ donde $z = x_1^2 + x_2^2$



SVM: El 'Truco del Kernel' para Datos no Lineales

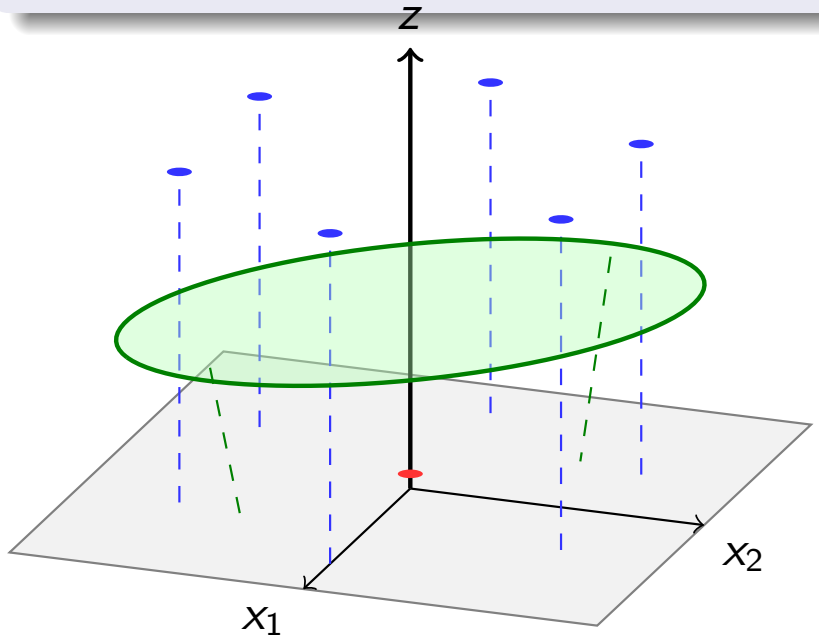
El Problema

¿Qué pasa si los datos no se pueden separar con una línea recta?



En 3D

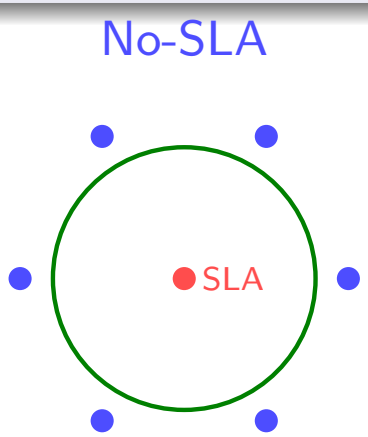
Se proyecta los datos a R^3 aplicando $\varphi(x_1, x_2) = (x_1, x_2, z)$ donde $z = x_1^2 + x_2^2$



SVM: El 'Truco del Kernel' para Datos no Lineales

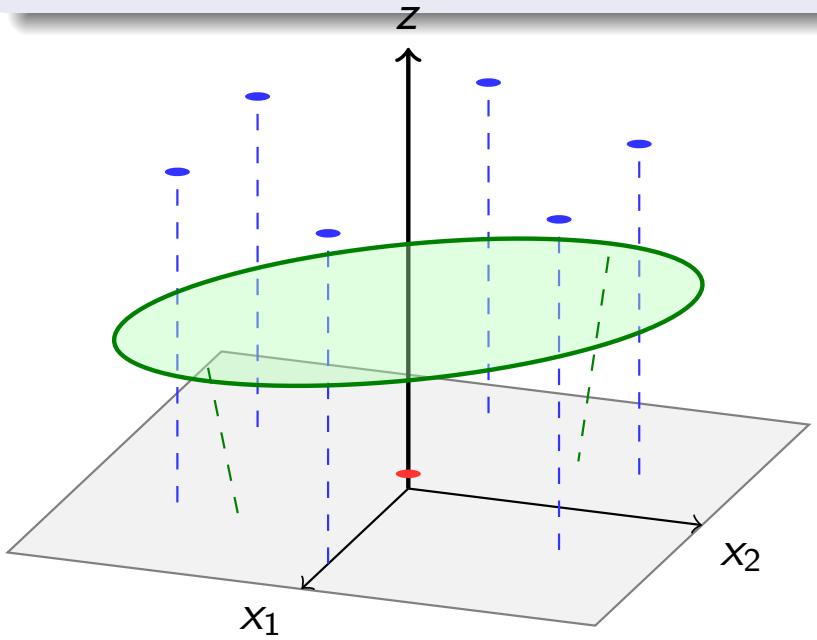
El Problema

¿Qué pasa si los datos no se pueden separar con una línea recta?



En 3D

Se proyecta los datos a R^3 aplicando $\varphi(x_1, x_2) = (x_1, x_2, z)$ donde $z = x_1^2 + x_2^2$



El 'Truco'

El **Kernel RBF** de SVM realiza esta separación proyectando a R^2 la intersección entre el hiperplano y el paraboloide formado por la función $z = x_1^2 + x_2^2$ **sin tener que calcular explícitamente** las coordenadas en la nueva dimensión, bajando el costo computacional.

¿Por Qué SVM es una Elección Ideal para Este Proyecto?

1. Efectividad en Alta Dimensión

- Nuestros vectores BoVW tienen k=500 dimensiones.
- SVM está diseñado para funcionar en estos espacios de muchas características sin caer en la 'problematica de la dimensionalidad'.

¿Por Qué SVM es una Elección Ideal para Este Proyecto?

1. Efectividad en Alta Dimensión

- Nuestros vectores BoVW tienen k=500 dimensiones.
- SVM está diseñado para funcionar en estos espacios de muchas características sin caer en la 'problematica de la dimensionalidad'.

2. Poder de los Kernels para Relaciones Complejas

- Usamos el **Kernel RBF (Función de Base Radial)**.
- Este kernel es extremadamente flexible y puede crear fronteras de decisión muy complejas y no lineales.
- Es perfecto para capturar los sutiles patrones en los histogramas que diferencian una estructura SLA de una que no lo es.

¿Por Qué SVM es una Elección Ideal para Este Proyecto?

1. Efectividad en Alta Dimensión

- Nuestros vectores BoVW tienen k=500 dimensiones.
- SVM está diseñado para funcionar en estos espacios de muchas características sin caer en la 'problematica de la dimensionalidad'.

2. Poder de los Kernels para Relaciones Complejas

- Usamos el **Kernel RBF (Función de Base Radial)**.
- Este kernel es extremadamente flexible y puede crear fronteras de decisión muy complejas y no lineales.
- Es perfecto para capturar los sutiles patrones en los histogramas que diferencian una estructura SLA de una que no lo es.

3. Excelente Generalización

- El principio de maximizar el margen hace que el modelo sea inherentemente robusto contra el sobreajuste (overfitting).
- Esto significa que el modelo no solo aprende bien los datos de entrenamiento, sino que también predice correctamente datos nuevos, como demuestran nuestras curvas de aprendizaje.

Resultados Comparativos

		SVM	RandomForest
Detector+Descriptor	**Normalización**	**Accuracy**	**Accuracy**
BRISK+BRISK	Global	0.9076	0.8739
	Local	**0.9916**	0.8908
FAST+BRIEF	Global	**0.9916**	0.9748
	Local	**0.9916**	0.9496
FAST+BRISK	Global	0.9580	0.8992
	Local	0.9664	0.9328
FAST+SIFT	Global	**0.9916**	0.9832
	Local	0.9832	0.9832
ORB+ORB	Global	0.9153	0.8559
	Local	0.8898	0.7966
SIFT+SIFT	Global	0.9412	0.8739
	Local	0.9580	0.8739

Resultados Comparativos

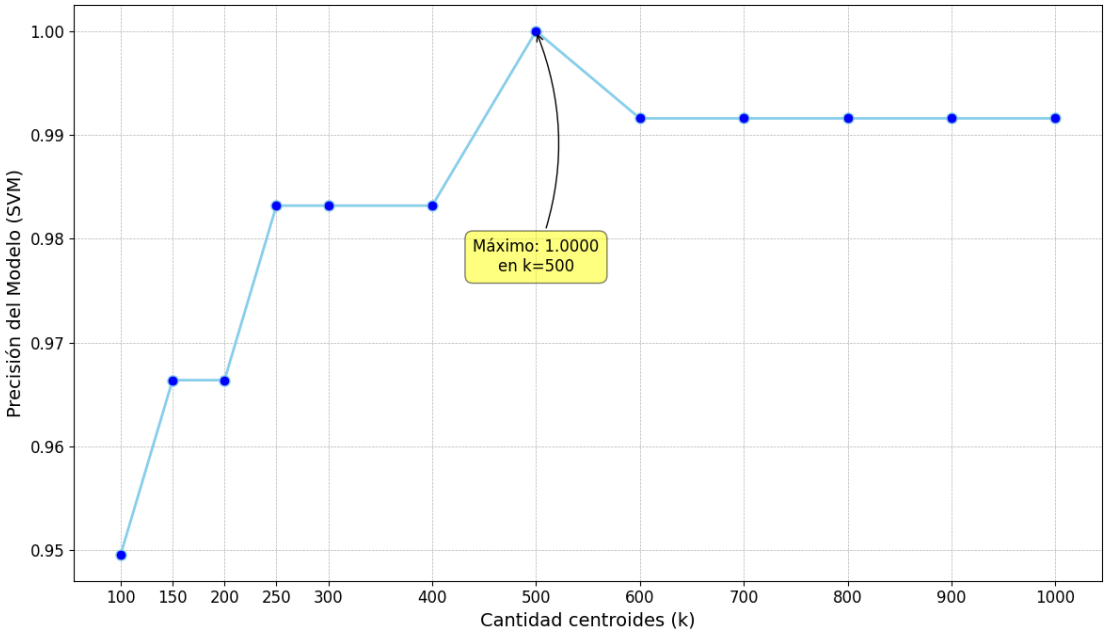
		SVM	RandomForest
Detector+Descriptor	**Normalización**	**Accuracy**	**Accuracy**
BRISK+BRISK	Global	0.9076	0.8739
	Local	**0.9916**	0.8908
FAST+BRIEF	Global	**0.9916**	0.9748
	Local	**0.9916**	0.9496
FAST+BRISK	Global	0.9580	0.8992
	Local	0.9664	0.9328
FAST+SIFT	Global	**0.9916**	0.9832
	Local	0.9832	0.9832
ORB+ORB	Global	0.9153	0.8559
	Local	0.8898	0.7966
SIFT+SIFT	Global	0.9412	0.8739
	Local	0.9580	0.8739

Conclusión

La combinación **FAST+BRIEF con SVM** demostró ser una de las más precisas y la más eficiente computacionalmente.

Optimización de k y Validación

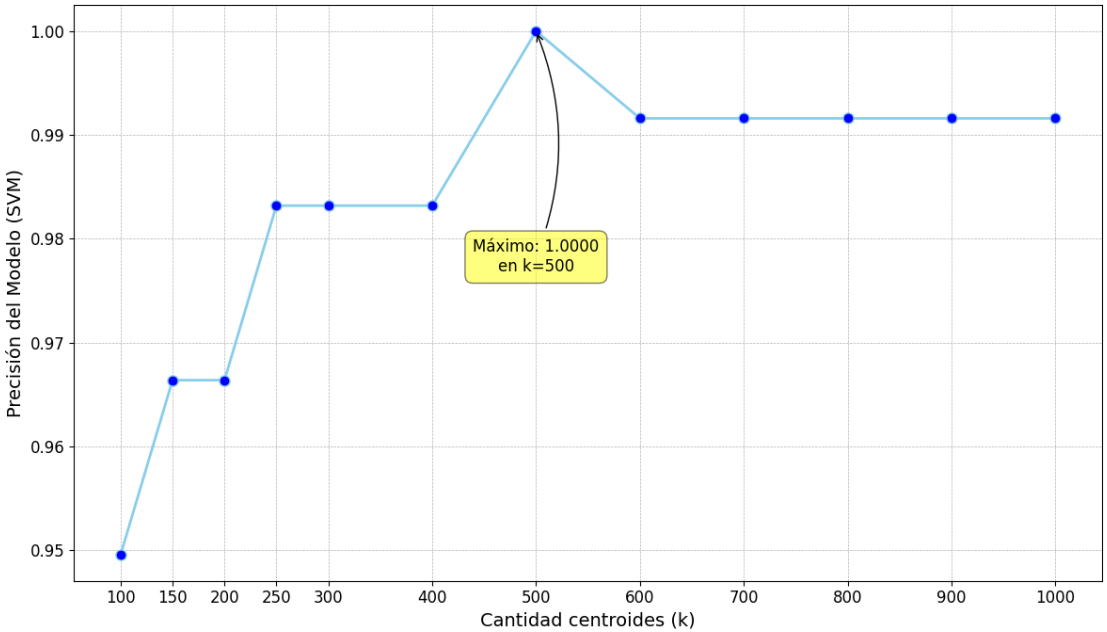
Optimización del tamaño del vocabulario (k)



- El máximo rendimiento se alcanza con **k=500**.
- Se usó **validación cruzada** estratificada de 5 particiones.
- Esto asegura que el resultado no es producto del azar.

Optimización de k y Validación

Optimización del tamaño del vocabulario (k)



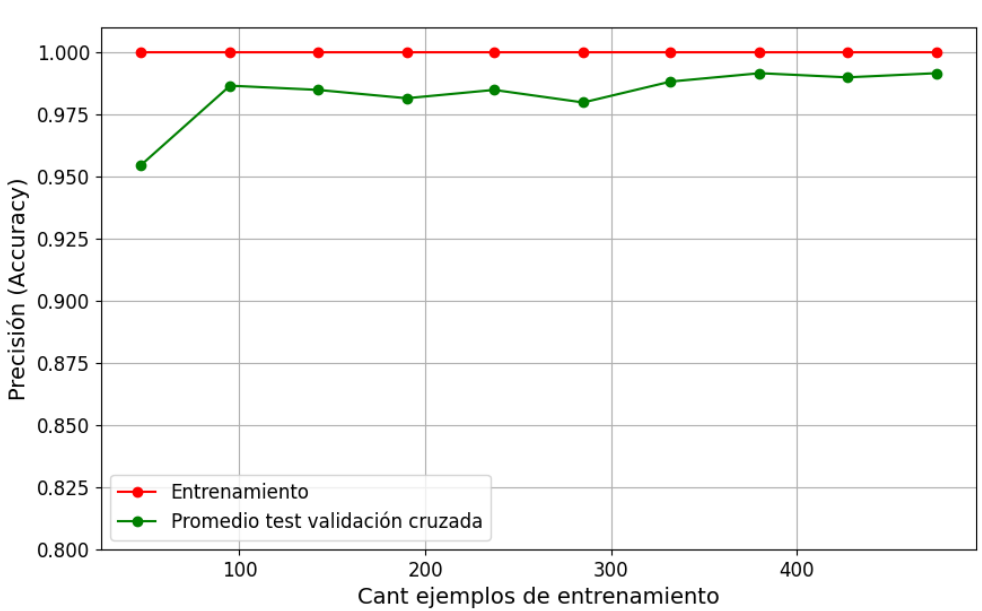
- El máximo rendimiento se alcanza con **k=500**.
- Se usó **validación cruzada** estratificada de 5 particiones.
- Esto asegura que el resultado no es producto del azar.

Rendimiento Final del Modelo (FAST+BRIEF)

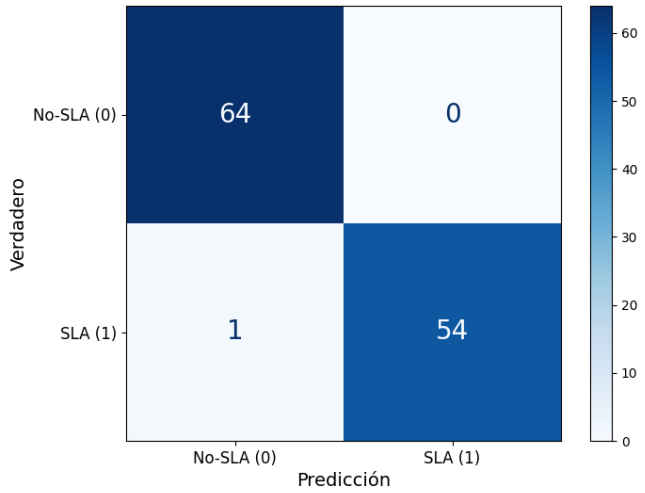
Precisión Promedio: **99.16 % ± 0.54 %**

Métricas finales

Curva de Aprendizaje

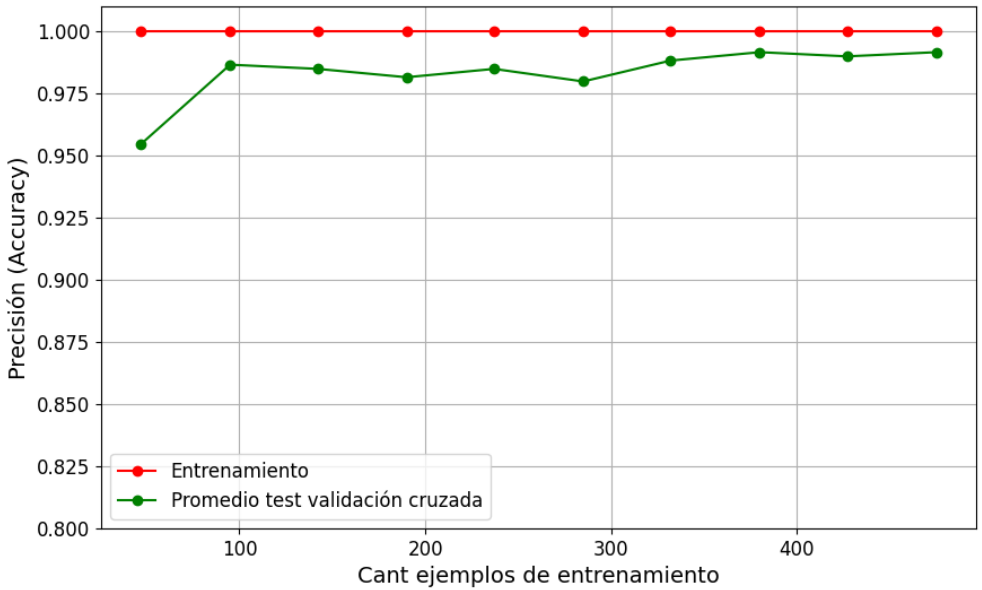


Matriz de Confusión

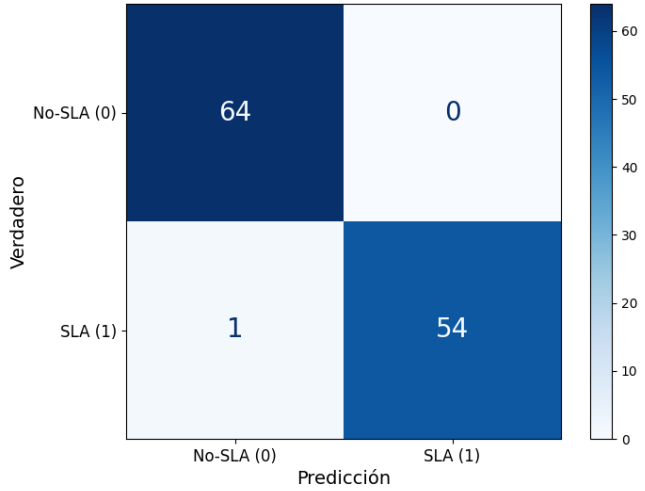


Métricas finales

Curva de Aprendizaje



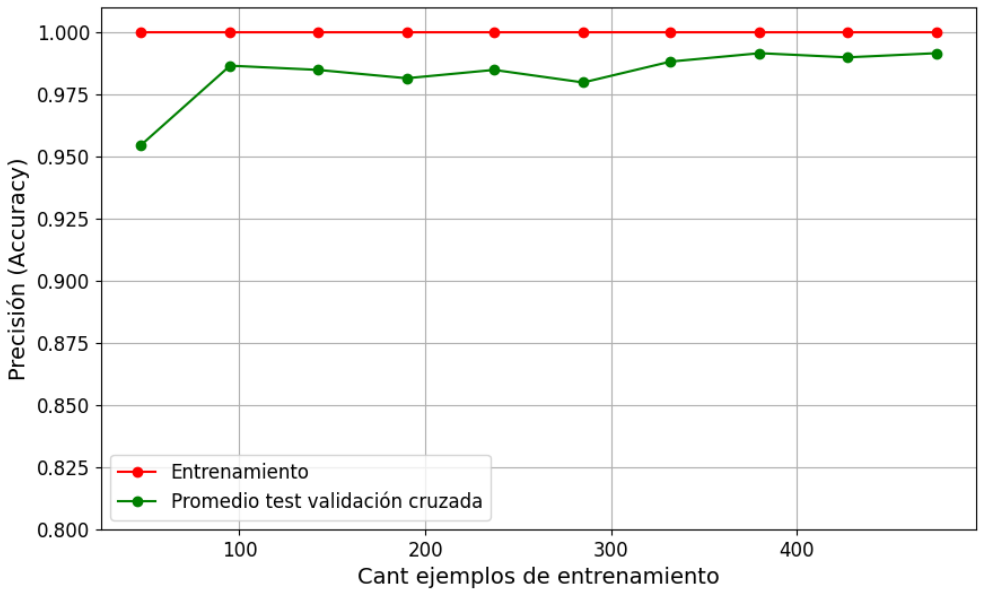
Matriz de Confusión



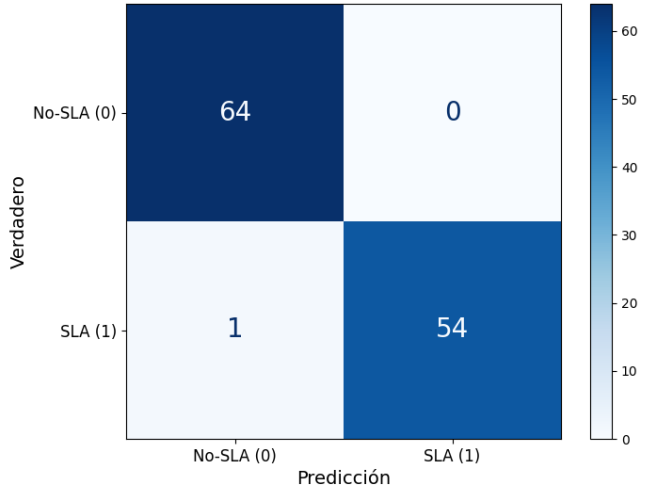
- Las curvas de entrenamiento y validación convergen a un valor alto y se mantienen juntas $\Longrightarrow$ **Buena generalización, sin sobreajuste**.

Métricas finales

Curva de Aprendizaje



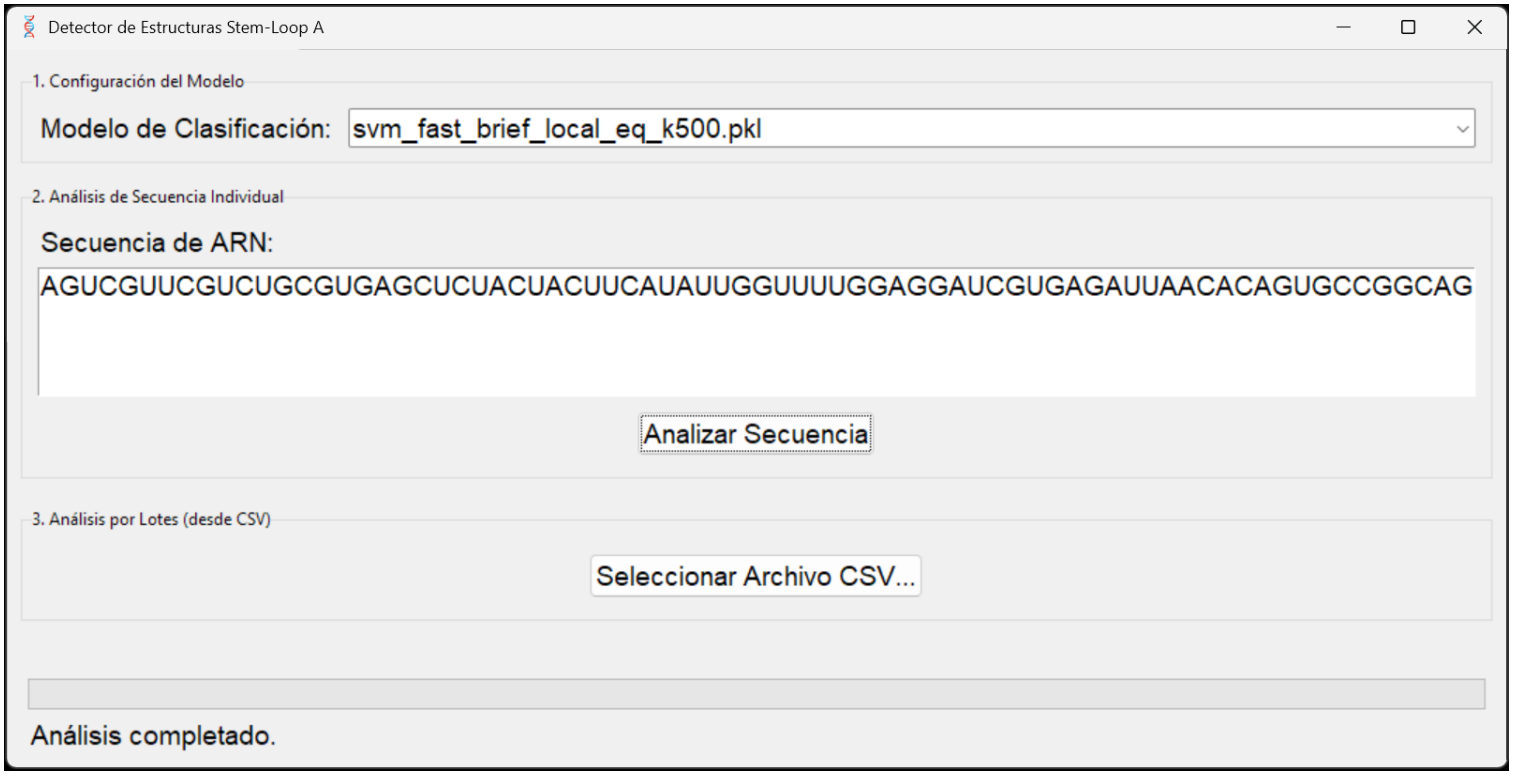
Matriz de Confusión



- Las curvas de entrenamiento y validación convergen a un valor alto y se mantienen juntas $\Longrightarrow$ **Buena generalización, sin sobreajuste**.
- La matriz de confusión ilustra el rendimiento casi perfecto, con un único error.

GUI desarrollada

Para encapsular la solución, se desarrolló una aplicación de escritorio con Tkinter que permite usar el modelo entrenado.



Funcionalidades:

- Seleccionar el modelo a utilizar.
- Analizar una única secuencia de ARN.
- Procesar un lote de secuencias desde un archivo.

Conclusiones y Trabajo Futuro

Conclusiones Clave

- Se construyó un sistema híbrido y reproducible con una precisión superior al **99 %**.
- La **re-conceptualización de datos** (secuencias $\rightarrow$ imágenes) fue una estrategia exitosa.
- El **preprocesamiento** (ecualización) y la selección modular de herramientas fueron fundamentales.

Conclusiones y Trabajo Futuro

Conclusiones Clave

- Se construyó un sistema híbrido y reproducible con una precisión superior al **99 %**.
- La **re-conceptualización de datos** (secuencias $\rightarrow$ imágenes) fue una estrategia exitosa.
- El **preprocesamiento** (ecualización) y la selección modular de herramientas fueron fundamentales.

Trabajo Futuro

- Extender el enfoque para detectar otros motivos funcionales en ARN.
- Entrenar el modelo con datos sintéticos para reducir posibles sesgos.
- Desarrollar una herramienta que analice genomas completos en busca de estructuras SLA.

¡Muchas Gracias!

Preguntas