Detección de Estructuras ARN tipo SLA mediante RNA-FM y Visión por Computadora

Proyecto Final Integrador
Tecnicatura Universitaria en Inteligencia Artificial

Universidad Nacional de Hurlingham

Alumno: Fernando Lopez Orts
Tutor: Lic. Emiliano Churruca
Dr. Gabriel Iglesias

8 de agosto de 2025

Resumen

Este trabajo presenta y valida un sistema computacional híbrido para la detección de estructuras de ARN tipo Stem-Loop A (SLA), un elemento funcional clave en el genoma de flavivirus como el del Dengue. El enfoque propuesto evita la predicción explícita de la estructura secundaria, un proceso computacionalmente costoso. En su lugar, las secuencias de ARN se transforman en representaciones numéricas (*embeddings*) de longitud variable utilizando el modelo fundacional RNA-FM. Estos embeddings se reconceptualizan como imágenes en escala de grises, cuyo contraste se mejora mediante ecualización de histograma. Posteriormente, se aplica el modelo Bag of Visual Words (BoVW), utilizando el detector FAST y el descriptor BRIEF, para convertir el conjunto variable de características locales de cada imagen en un vector de características de longitud fija. Finalmente, un clasificador de Máquinas de Vectores de Soporte (SVM) con un kernel RBF predice la presencia o ausencia de la estructura SLA. El modelo final, validado mediante validación cruzada, alcanzó una precisión promedio del 98.99 %, demostrando que la fusión de modelos de lenguaje biológico y técnicas de visión por computadora es una estrategia eficaz y computacionalmente eficiente para problemas de clasificación bioinformática.

1. Introducción

Contexto del problema. El ácido ribonucleico (ARN) desempeña roles fundamentales en la biología celular. En virus de ARN de cadena positiva, como el virus del dengue (DENV) —miembro del género *Flavivirus*, un grupo de importantes patógenos humanos transmitidos por mosquitos y garrapatas—, el ARN no solo constituye el material genético, sino que además contiene elementos estructurales conservados que son indispensables para su ciclo de vida. Uno de estos elementos es una estructura de aproximadamente 70-90 nucleótidos conocida como **Stem-Loop A (SLA)**, localizada en la región 5' no traducida (5'UTR) del genoma viral (Figura 1).

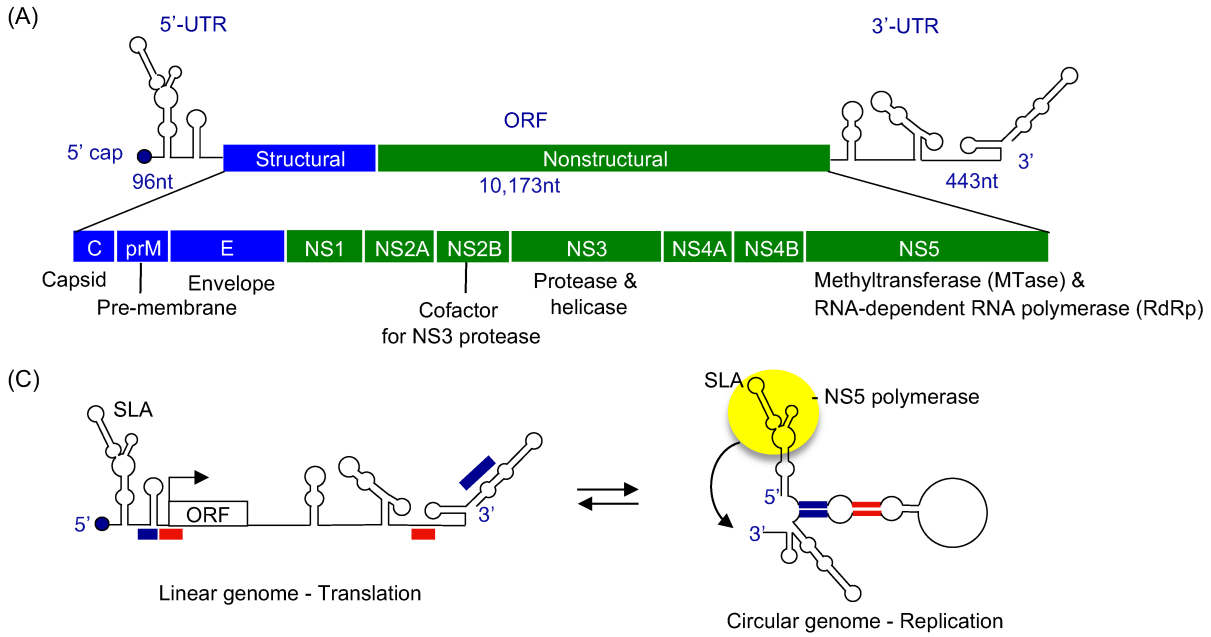

Figura 1: Organización del genoma de un Flavivirus. (A) El genoma contiene una región 5' no traducida (5'-UTR), un marco de lectura abierto (ORF) que codifica las proteínas estructurales y no estructurales, y una región 3' no traducida (3'-UTR). La estructura SLA se encuentra al inicio del 5'-UTR. (C) Modelo del cambio conformacional del genoma: lineal para la traducción de proteínas y circularizado para la replicación, donde la polimerasa NS5 reconoce la SLA en el extremo 5' para iniciar la síntesis en el extremo 3'. Imagen tomada de Choi [7].

La estructura SLA funciona como un **promotor** que es reconocido específicamente por la polimerasa viral ARN-dependiente de ARN (RdRp), la proteína no estructural 5 (NS5). Como demuestran los trabajos de Obi et al. [6] y Bujalowski et al. [3], esta interacción es un paso crítico e indispensable para iniciar la síntesis de la hebra de ARN de polaridad negativa, dando comienzo a la replicación del genoma viral. La funcionalidad de SLA no depende de una secuencia de nucleótidos específica, sino de su conformación tridimensional particular. Como se muestra en la Figura 2B, esta estructura adopta una topología en forma de 'Y', compuesta por un tallo inferior (*bottom stem*), un bucle lateral (*side loop*) y un tallo-bucle superior (*top stem-loop*). Esta arquitectura, que incluye características conservadas como un 'U-bulge' (Figura 2A), se mantiene entre los diferentes serotipos del DENV e incluso en otros flavivirus, como describen Lodeiro et al. [1], Filomatori et al. [2] y Lee et al. [5].

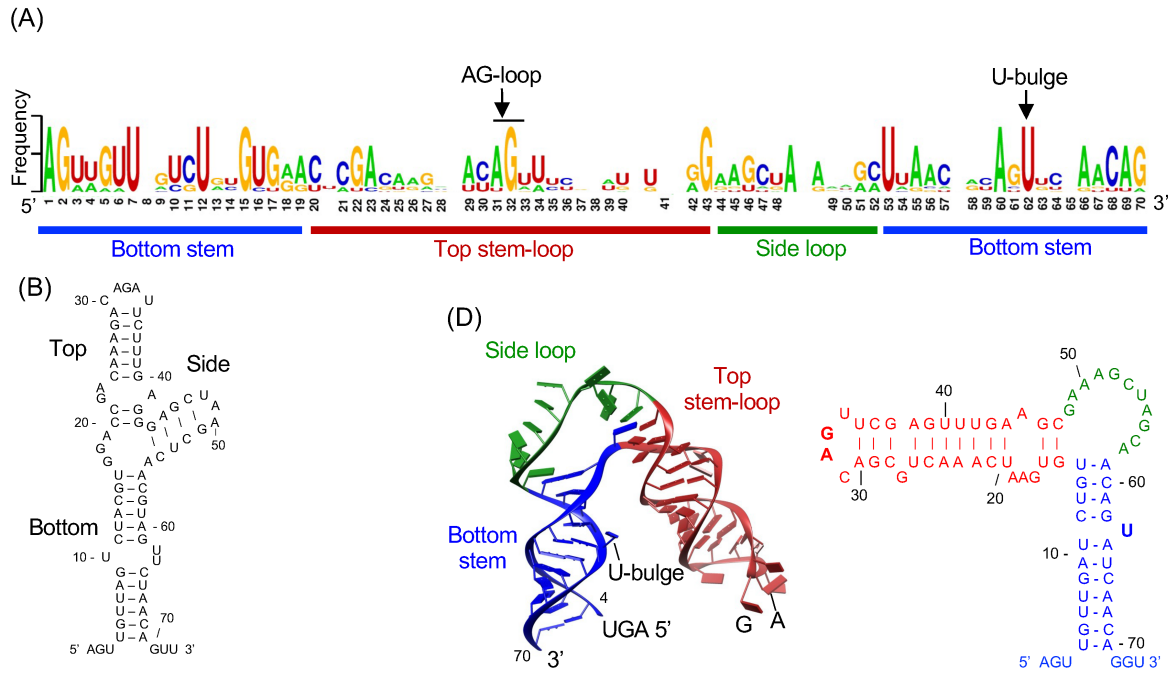

Figura 2: Estructura del promotor SLA de Flavivirus. (A) Muestra la conservación de la secuencia en diferentes flavivirus mediante un logo de secuencia. La altura total de las letras en cada posición indica el grado de conservación, mientras que la altura de cada letra individual representa su frecuencia relativa. Se marcan regiones estructuralmente importantes y motivos conservados como el 'AG-loop' y el 'U-bulge'. (B) Representa la estructura secundaria *predicha* para el SLA de DENV2, mostrando un plegamiento en forma de 'Y'. (D) Muestra la estructura del SLA del virus del Zika (ZIKV), que, a pesar de las diferencias en la secuencia, conserva la topología general en 'L' observada en DENV. Imagen tomada de Choi [7].

Dada su función esencial y su conservación estructural, detallada por Sun y Varani [4], la estructura SLA se ha mostrado prometedora para el desarrollo de terapias antivirales. Por lo tanto, la capacidad de predecir si una secuencia de ARN puede adoptar la conformación característica de SLA es un objetivo de gran interés. El enfoque tradicional para esta tarea consiste en la predicción explícita de la estructura secundaria, un proceso que puede ser computacionalmente costoso.

Motivación e hipótesis del proyecto. La inferencia directa de la estructura es computacionalmente compleja. La hipótesis de este proyecto es que es posible eludir esta reconstrucción explícita. La propuesta es utilizar un modelo de lenguaje de gran tamaño para generar una representación numérica (embedding) que ya contenga la información estructural de forma implícita. Sin embargo, esto introduce un desafío técnico clave: las secuencias de ARN tienen longitudes variables ('L'), resultando en embeddings de dimensiones variables de 'L x 640'. Los algoritmos de clasificación estándar requieren vectores de entrada de longitud fija. La motivación de este proyecto es resolver este problema mediante una **re-conceptualización**: tratar el embedding como una imagen y aplicar una secuencia de métodos de visión por computadora para extraer un vector de características de longitud fija.

Objetivo general y específicos. El objetivo general de este trabajo es desarrollar y validar un sistema computacional que transforme secuencias de ARN en representaciones visuales y las clasifique según la presencia de la estructura SLA. Los objetivos específicos, basados en el plan de trabajo original, son:

1. Implementar un método para obtener las representaciones numéricas de las secuencias de ARN y transformarlas en representaciones visuales (imágenes).
2. Obtener descriptores de características locales de las imágenes mediante algoritmos de visión por computadora y resolver el problema de la dimensionalidad variable.
3. Utilizar y evaluar clasificadores de aprendizaje automático para predecir la presencia de la estructura SLA a partir de los vectores de características generados.
4. Analizar y visualizar los resultados para validar la eficacia del sistema y seleccionar la combinación de algoritmos con mejor desempeño.

Breve descripción de la solución propuesta. La solución desarrollada sigue la hipótesis inicial. Las secuencias de ARN se convierten en matrices de reales de ('L' $\times$ 640) mediante **RNA-FM**, desarrollado por Chen et al. [18]. A los valores de estas matrices se les normaliza (local o globalmente) con valores entre 0 y 255 y se genera la imagen. Luego se extraen características locales (descriptores) utilizando una variedad de algoritmos del campo de visión por computadora. Mediante el modelo Bag of Visual Words (BoVW), propuesto por Csurka et al. [23], el conjunto variable de características de cada imagen se convierte en un único vector-histograma de longitud fija. Finalmente, un clasificador entrenado con estos vectores, realiza la predicción final.

Representamos esta secuencia de pasos en la Figura 3.

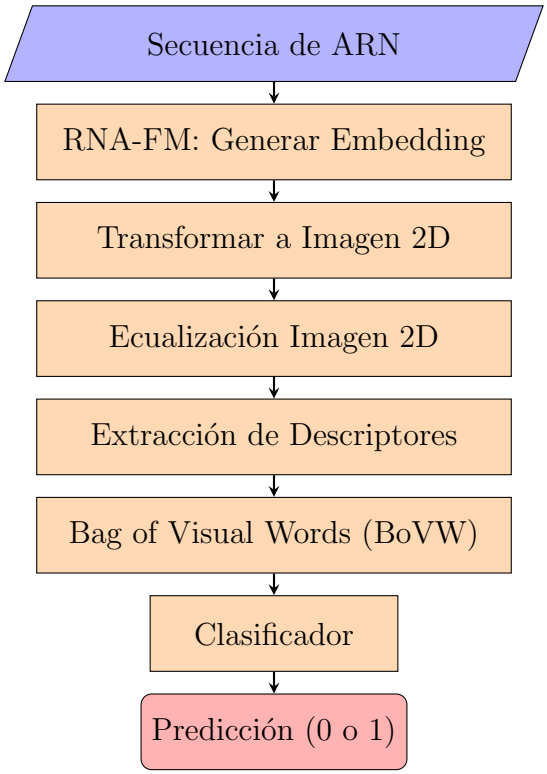

Figura 3: Diagrama de la arquitectura de la solución propuesta.

2. Marco Teórico y Estado del Arte

Evolución en la Predicción de Estructura Secundaria de ARN

La predicción de la estructura secundaria del ARN, es decir, el patrón de apareamiento de bases intramoleculares, es un problema fundamental en la biología computacional. Durante décadas, el campo estuvo dominado por **métodos termodinámicos** que utilizan la minimización de la energía libre para encontrar la conformación más estable. Herramientas como `Mfold`, desarrollado por Zuker [9], y el paquete `ViennaRNA`, introducido por Hofacker et al. [8], se convirtieron en estándares al implementar algoritmos basados en parámetros energéticos derivados experimentalmente.

Sin embargo, estos métodos clásicos enfrentan limitaciones significativas, como señalan Singh et al. [10]. Su rendimiento se ha estancado, en parte, debido a su dificultad para modelar dos tipos de complejidades estructurales:

- **Interacciones no canónicas:** Se refieren a apareamientos de bases que no siguen las reglas estándar de Watson-Crick (A-U, G-C) o la regla del 'tambaleo' (G-U). Estas interacciones 'fuera de regla' son biológicamente importantes y difíciles de modelar termodinámicamente con la misma precisión que las canónicas.
- **Estructuras complejas como los pseudonudos:** Un pseudonudo es una configuración en la que las bases de un bucle se aparean con bases fuera de ese bucle. Desde una perspectiva computacional, esto rompe la jerarquía de anidamiento simple que asumen los algoritmos de programación dinámica clásicos. Si imaginamos la secuencia de ARN como un arco, los apareamientos en una estructura simple nunca se cruzan. En un pseudonudo, sí lo hacen, lo que incrementa exponencialmente la complejidad del espacio de búsqueda y requiere algoritmos mucho más costosos.

La llegada del **aprendizaje profundo** ha supuesto un cambio de paradigma para abordar estos desafíos. Modelos como `SPOT-RNA` de Singh et al. [10] y, más recientemente, `RNAformer` de Franke et al. [11], han demostrado mejoras significativas en la precisión al tratar la predicción como un problema de aprendizaje contextual profundo sobre la secuencia completa, siendo capaces de inferir estos patrones complejos directamente de los datos. A diferencia de estos enfoques, que buscan una reconstrucción estructural explícita, este proyecto propone un método de clasificación que infiere la presencia de un motivo funcional (SLA) directamente desde una representación latente de la secuencia, eludiendo la predicción estructural directa.

Modelos Fundacionales de Lenguaje para Secuencias Biológicas

Inspirados por su éxito en el procesamiento del lenguaje natural, los Modelos de Lenguaje de Gran Tamaño (LLMs) se han adaptado para descifrar el 'lenguaje de la vida' codificado en secuencias biológicas, como revisan Liu et al. [12]. Estos modelos, denominados **modelos fundacionales**, se pre-entrenan en corpus masivos de secuencias de ADN, ARN o proteínas de manera auto-supervisada para aprender representaciones numéricas (embeddings) que capturan información sintáctica y semántica fundamental, según Wang et al. [13].

Para el dominio del ARN, ha surgido un conjunto de métodos basados en LLMs especializados. Modelos como `RiNALMo`, presentado por Penić et al. [14], y `ERNIE-RNA`, propuesto

por Yin et al. [15], han demostrado capacidades avanzadas de generalización en tareas de predicción estructural. Estos desarrollos son revisados y comparados en estudios exhaustivos como los de Chaturvedi et al. [17] y Zablocki et al. [16]. Para este proyecto, se seleccionó el modelo **RNA-FM** debido a su robustez, disponibilidad pública y su validación como un modelo fundacional de alto rendimiento.

RNA-FM: Un Modelo para la Representación de ARN. RNA-FM, desarrollado por Chen et al. [18], es un modelo de lenguaje basado en la arquitectura Transformer-Encoder (BERT). Fue pre-entrenado utilizando la tarea de modelado de lenguaje enmascarado (MLM) sobre un corpus masivo de 23.7 millones de secuencias de ARN no codificante de la base de datos RNAcentral. En este esquema de entrenamiento, ciertas posiciones de la secuencia se reemplazan por un token especial de máscara, y el modelo debe predecir el contenido original en esas posiciones utilizando el contexto circundante. Este enfoque permite al modelo aprender representaciones ricas y contextualizadas que capturan patrones tanto locales como globales dentro de la secuencia. La arquitectura de RNA-FM, compuesta por 12 capas, genera un vector de dimensión 640 por nucleótido, lo que facilita una representación informativa y efectiva para el aprendizaje por transferencia en diversas tareas posteriores. Estos nucleótidos son 'adenina'(A), 'guanina' (G), 'citosina' (C) y 'uracilo' (U), este último reemplaza a la timina (T) del ADN.

La elección de RNA-FM se fundamenta en su desempeño documentado en la evaluación comparativa rigurosa de modelos ARN realizada por Zablocki et al. [16], donde RNA-FM demostró ser uno de los modelos con mejor rendimiento. Su capacidad para capturar patrones estructurales y funcionales directamente desde la secuencia primaria, sin necesidad de alineamientos múltiples ni información evolutiva explícita, lo establece como una elección sólida y fiable para el propósito de este proyecto: generar representaciones estructuralmente informadas de secuencias de ARN.

Métodos de Visión por Computadora para Clasificación

Para transformar los embeddings de ARN en un formato compatible con algoritmos de clasificación, este proyecto adopta una secuencia de métodos de visión por computadora. Este enfoque se aleja de las arquitecturas de aprendizaje profundo de extremo a extremo y se basa en un flujo modular de detección, descripción y codificación de características.

Detección y Descripción de Características Locales

SIFT (Scale-Invariant Feature Transform). Es uno de los algoritmos más robustos e influyentes en la visión por computadora. Su proceso se divide en cuatro etapas principales:

1. **Detección de extremos en el espacio de escala:** Para encontrar puntos que sean robustos a cambios de escala, SIFT utiliza una pirámide de Diferencia de Gaussianas (DoG). La imagen se desenfoca progresivamente con filtros Gaussianos de diferente intensidad, y se calculan las diferencias entre imágenes consecutivas. Los puntos de interés se identifican como los máximos o mínimos locales en esta pirámide tridimensional (dos dimensiones espaciales y una de escala).
2. **Localización de puntos clave:** Los puntos candidatos se refinan para eliminar aquellos con bajo contraste (sensibles al ruido) y los que se encuentran en bordes

(mal localizados). Esto se logra mediante un análisis de la matriz de Hessian.

3. **Asignación de orientación:** Para lograr la invarianza a la rotación, se calcula la orientación dominante de los gradientes de la imagen en una ventana alrededor de cada punto clave. Esta orientación se asigna al punto, y todas las operaciones posteriores se realizan de forma relativa a ella.

4. **Generación del descriptor:** Se toma una región de 16x16 píxeles alrededor del punto clave, se rota según la orientación asignada y se divide en una cuadrícula de 4x4 subregiones. Para cada subregión, se calcula un histograma de 8 orientaciones de gradiente. La concatenación de estos 16 histogramas da como resultado un vector descriptor de 128 dimensiones (16 subregiones $\times$ 8 orientaciones), que es altamente distintivo y robusto.

Detector FAST (Features from Accelerated Segment Test). Desarrollado por Rosten y Drummond [19], es un detector de esquinas de alta velocidad. Su criterio se basa en el 'test de segmento': para un píxel candidato p, se examina un círculo de 16 píxeles a su alrededor (Figura 4). Se clasifica como esquina si existe un arco contiguo de n píxeles (usualmente 9) que son todos significativamente más brillantes o más oscuros que el centro por un umbral t. Para acelerar el proceso, primero se comprueban los píxeles 1, 9, 5 y 13 del círculo; si al menos tres de ellos no cumplen la condición, el punto se descarta. Finalmente, se aplica la supresión de no máximos para eliminar puntos detectados en regiones adyacentes.

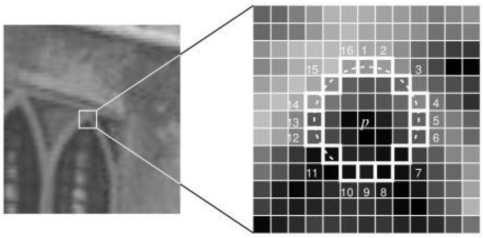

Figura 4: Máscara circular de 16 píxeles utilizada por el detector FAST. El píxel central (p) se compara con los píxeles del círculo para determinar si es una esquina.

Descriptor BRIEF (Binary Robust Independent Elementary Features). Propuesto por Calonder et al. [20], es un descriptor binario extremadamente eficiente. En lugar de calcular gradientes, genera una cadena de bits (típicamente de 128, 256 o 512 bits) a partir de simples comparaciones de intensidad. Primero, se define un conjunto de pares de puntos dentro de un parche alrededor del punto clave. Para cada par, se compara la intensidad de sus píxeles: si el primer píxel es más brillante que el segundo, el bit correspondiente es 1; de lo contrario, es 0. La comparación entre dos descriptores BRIEF se realiza mediante la distancia de Hamming (contando el número de bits que difieren), que se calcula con una operación XOR a nivel de bits, siendo órdenes de magnitud más rápida que la distancia euclidiana usada para SIFT.

ORB (Oriented FAST and Rotated BRIEF). Propuesto por Rublee et al. [21], es una alternativa eficiente y libre de patentes a SIFT que fusiona las mejores ideas de FAST y BRIEF.

- **Detector:** Utiliza FAST para detectar esquinas, pero añade un componente de escala (construyendo una pirámide de imágenes) y las ordena usando una medida de Harris para quedarse con las mejores.
- **Orientación:** Para ser invariante a la rotación, ORB calcula la orientación del parche usando el método del 'centroide de intensidad'. Calcula los momentos de una región y utiliza su centroide para determinar una orientación canónica.
- **Descriptor:** Emplea una versión modificada de BRIEF llamada rBRIEF (rotated BRIEF). Una vez calculada la orientación del punto, el patrón de pares de puntos de BRIEF se rota de acuerdo a esa orientación antes de generar el descriptor binario. Esto asegura que el descriptor sea el mismo independientemente de la rotación de la imagen.

BRISK (Binary Robust Invariant Scalable Keypoints). Propuesto por Leutenegger et al. [22], es otro detector y descriptor binario que busca un equilibrio entre velocidad y robustez.

- **Detector:** Al igual que SIFT y ORB, detecta puntos clave en un espacio de escala para ser invariante a la escala.
- **Descriptor:** Utiliza un patrón de muestreo fijo de puntos distribuidos en círculos concéntricos alrededor del punto clave. La orientación se determina comparando las intensidades de pares de puntos de larga distancia en este patrón. Una vez determinada la orientación, el descriptor binario se construye a partir de comparaciones de intensidad de pares de corta distancia, lo que le da robustez ante el ruido.

El primer paso consiste en identificar puntos de interés dentro de la 'imagen' generada por el embedding normalizado (entre 0 y 255) y describir su vecindario local. La combinación **FAST+BRIEF** fue seleccionada por su eficiencia y alto rendimiento, como se detalla en la sección de resultados. Sin embargo, para realizar una evaluación exhaustiva, también se probaron otros algoritmos estándar en el campo.

Codificación con el Modelo Bag of Visual Words (BoVW)

El modelo Bag of Visual Words (BoVW) es una técnica adaptada del procesamiento de lenguaje natural que permite crear una representación de longitud fija a partir de un conjunto de características locales de tamaño variable. El proceso consta de dos fases principales:

1. **Creación del Vocabulario Visual (Fase de Entrenamiento):**
   - Se extraen los descriptores de características locales (e.g., BRIEF) de todas las imágenes del conjunto de entrenamiento. Esto genera un gran conjunto de miles o millones de vectores descriptores.
   - Se aplica un algoritmo de clustering, como Mini-Batch K-Means (una variante eficiente de K-Means para grandes volúmenes de datos), para agrupar estos descriptores en k clústeres. El hiperparámetro k define el tamaño del vocabulario.
   - Los centroides de estos k clústeres se convierten en las 'palabras visuales' de nuestro vocabulario. Cada palabra visual representa una característica local

común en los datos.

2. **Generación de Histogramas (Fase de Inferencia):**

   - Para cada imagen (tanto de entrenamiento como de prueba), se extraen sus descriptores locales.
   - Cada descriptor se asigna a un centroide del clúster (*palabra visual*) más cercano según una métrica de distancia (e.g., distancia de Hamming para descriptores binarios). La **distancia de Hamming** entre dos vectores binarios de igual longitud se calcula como el número de posiciones en las que difieren. Formalmente, dada una pareja de vectores binarios x e y, ambos de dimensión n, la distancia de Hamming se define como:

$$d_H(x, y) = \sum_{i=1}^{n} 1(x_i \neq y_i)$$

     donde $1(x_i \neq y_i)$ es una función indicadora que vale 1 si los bits en la posición i son distintos y 0 en caso contrario. Este proceso de asignación se conoce como *cuantización vectorial*.
   - Se construye un histograma de frecuencia de tamaño k. Cada bin del histograma cuenta cuántos descriptores de la imagen fueron asignados al centroide correspondiente.
   - Finalmente, el histograma se normaliza para que sea invariante al número total de características en la imagen. El vector resultante, de longitud fija k, es la representación final de la imagen.

$$H' = \frac{H}{\|H\|_2} = \frac{H}{\sqrt{\sum_{j=1}^{k} H[j]^2}}$$

     donde H es el histograma de frecuencias absolutas y H' es el histograma normalizado (Figura 5).

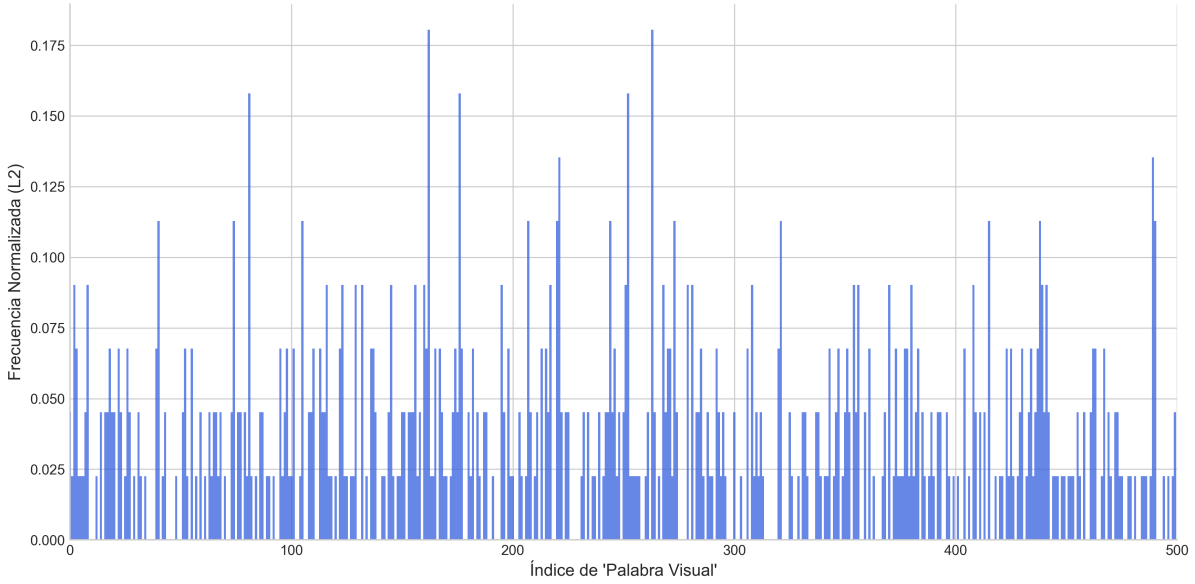

Figura 5: Representación del vector de salida luego de aplicar BoVW.

Métodos de Clasificación

Una vez obtenidas las representaciones vectoriales de longitud fija, se utilizaron dos algoritmos de clasificación para la predicción final.

Máquinas de Vectores de Soporte (SVM). Este algoritmo de aprendizaje supervisado busca encontrar el hiperplano óptimo en un espacio de N dimensiones que separe las clases de datos. El hiperplano 'óptimo' es aquel que tiene el mayor margen, es decir, la mayor distancia entre él y los puntos de datos más cercanos de cualquiera de las clases (los **vectores de soporte**).

Para problemas no linealmente separables, el método SVM utiliza el **'truco del kernel'**, una función que calcula el producto escalar entre puntos en un espacio de mayor dimensionalidad sin tener que transformar los datos explícitamente. En este proyecto se utilizó el **kernel de base radial (RBF)**:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$$

El hiperparámetro **gamma (γ)** controla la influencia de cada punto. En implementaciones como la de la librería `scikit-learn`, además de valores numéricos, `gamma` puede tomar valores predefinidos:

- **'scale':** Es el valor recomendado y por defecto en versiones modernas. Calcula un valor para gamma basado tanto en el número de características como en la varianza de los datos, usando la fórmula $1/(\text{n\_features} \times X.\text{var}())$. Esta adaptabilidad lo hace robusto ante diferentes escalas de datos.
- **'auto':** Utiliza una heurística más simple, calculando gamma únicamente a partir del número de características: 1/n_features.

La optimización se realiza mediante un problema de **margen blando**, que equilibra la maximización del margen y la minimización de errores de clasificación, controlado por el hiperparámetro **C**.

Bosques Aleatorios (Random Forest). Es un método de aprendizaje conjunto (ensemble) que construye múltiples árboles de decisión durante el entrenamiento y genera la clase que es la moda de las clases de los árboles individuales. Su robustez contra el sobreajuste proviene de dos fuentes de aleatoriedad:

- **Bagging (Bootstrap Aggregating):** Cada árbol se entrena en una muestra aleatoria del conjunto de datos, extraída con reemplazo.
- **Aleatoriedad de características:** En cada nodo de un árbol, solo se considera un subconjunto aleatorio de características para encontrar la mejor división.

Estos dos mecanismos aseguran que los árboles del bosque sean diversos y que sus errores individuales tiendan a anularse, resultando en un modelo final más preciso y estable.

3. Solución Propuesta

Diseño de la Solución

La solución se diseñó como un proceso computacional modular y secuencial, como se muestra en la Figura 3. Este diseño permitió la experimentación sistemática con diferentes componentes en cada etapa.

Implementación

El proceso completo se implementó a través de una serie de scripts de Python, modulando cada etapa del flujo de trabajo.

Preprocesamiento de Datos y Generación de Embeddings

El punto de partida fue un archivo CSV que contenía 643 secuencias. Se aplicó un proceso de limpieza que incluyó la estandarización de las secuencias a su forma canónica de ARN (reemplazando Timina 'T' por Uracilo 'U') y la eliminación de duplicados. Finalmente, las etiquetas categóricas ('+' para presencia de SLA y '-' para ausencia) se mapearon a valores numéricos (1 y 0). Este proceso resultó en un conjunto de datos final de 592 secuencias únicas, distribuidas en 273 secuencias positivas y 319 negativas. Utilizando el modelo RNA-FM, cada una de estas secuencias se convirtió en un tensor numérico de longitud variable $(1 \times L_i \times 640)$. Este enfoque es crucial, ya que preserva la totalidad de la información biológica de cada secuencia, evitando el uso de técnicas de homogeneización de longitud como el ***padding***, que consiste en añadir valores nulos o 'tokens' especiales a las secuencias más cortas para que todas alcancen una longitud máxima común, lo cual puede introducir ruido artificial, o el ***truncamiento***, que implica acortar las secuencias más largas para ajustarlas a un tamaño fijo, con el riesgo evidente de eliminar información estructural o funcional crítica.

De los datos obtenidos se utilizó el 80 % (473 registros) para entrenamiento y el 20 % para testeo (119 registros), a esto refiere 'split 80/20'.

Creación y Mejora de Imágenes

Los embeddings se transformaron en imágenes en escala de grises. Este paso incluyó dos procesos críticos: la normalización de intensidades y la ecualización del histograma.

Estrategias de Normalización. Se exploraron dos estrategias para escalar los valores de los embeddings al rango de píxeles [0, 255]:

- **Normalización Global:** Se identifican los valores mínimo (min_{global}) y máximo (max_{global}) a lo largo de **todo el conjunto de datos**. Luego, cada valor v en cada embedding se escala utilizando la misma fórmula:

$$v' = 255 \times \frac{v - min_{global}}{max_{global} - min_{global}}$$

  Esta estrategia preserva las diferencias relativas de intensidad entre las distintas imágenes, pero puede ser sensible a valores atípicos que compriman el rango dinámico del resto de los datos.

- **Normalización Local:** Cada embedding (representado como una matriz) se normaliza de forma **independiente**. Para cada matriz, se encuentran su propio mínimo (min_{local}) y máximo (max_{local}), y sus valores se escalan en consecuencia:

$$v' = 255 \times \frac{v - min_{local}}{max_{local} - min_{local}}$$

  Esta técnica maximiza el contraste en cada imagen individualmente, pero elimina la información sobre las diferencias de intensidad relativas entre imágenes.

De este proceso se obtiene una matriz de $L_i \times 640$ con valores que van de 0 a 255, lo que se corresponde con los niveles de intensidad posibles para un píxel en escala de grises. Esto nos permite visualizarla como imagen en escala de grises como se puede observar en la Figura 6.

Ecualización del Histograma. Uno de los primeros impedimentos que nos encontramos se debió al bajo contraste de las imágenes normalizadas (Figura 6) que impedía extraer los puntos de interés. Se aplicó ecualización de histograma para redistribuir las intensidades y mejorar la visibilidad de las características, como se muestra en la Figura 7. El proceso se basa en la función de distribución acumulada (CDF) del histograma de la imagen, $h(i)$, para transformar cada nivel de intensidad i a un nuevo nivel i':

$$i' = T(i) = \text{round}\left((L-1) \cdot \sum_{j=0}^{i} \frac{h(j)}{N}\right)$$

donde L es el número de niveles de gris (256) y N es el total de píxeles. Este paso fue fundamental para que los detectores de características funcionaran.

Figura 6: Imagen de bajo contraste obtenida luego de normalizar.

Figura 7: Misma imagen anterior luego de la ecualización.

Extracción de Descriptores y Pruebas

Sobre las imágenes ecualizadas, se extrajeron características locales (Figura 8). Para realizar un análisis exhaustivo, se generaron 12 conjuntos de descriptores, producto de la combinación de 6 pares de detectores/descriptores (SIFT, BRISK, ORB, FAST+BRISK, FAST+BRIEF, FAST+SIFT) con las 2 estrategias de normalización. El candidato más prometedor, `FAST+BRIEF+Local+SVM`, se sometió a una validación robusta: se analizó la sensibilidad al tamaño del vocabulario k, se aplicó validación cruzada de 5 particiones y se realizó una búsqueda de hiperparámetros.

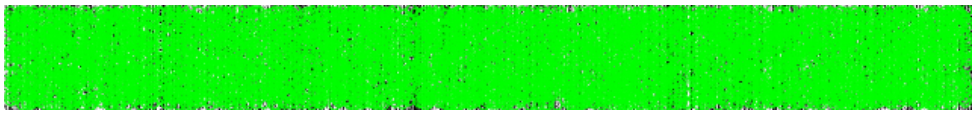

Figura 8: Ejemplo de los puntos de interés detectados por el algoritmo FAST (círculos verdes) sobre una imagen de ARN ecualizada. Cada círculo representa una región de la imagen con alto contenido informativo (esquinas, texturas) que será descrita por BRIEF.

4. Resultados

Resultados de la evaluación comparativa. La evaluación inicial reveló que la combinación de **FAST+BRIEF** con normalización local y un clasificador **SVM** alcanzaba un accuracy de **99.16 %**, superando a todas las demás combinaciones, como se detalla en la Tabla 1.

Cuadro 1: Resultados comparativos de clasificadores para diferentes combinaciones de algoritmos y normalización. Para este caso se utilizó un vocabulario de 500 centroides (k=500).

		SVM		**RandomForest**	
Detector+Descriptor	**Normalización**	**Accuracy**	**F1-Macro**	**Accuracy**	**F1-Macro**
BRISK+BRISK	Global	0.9076	0.91	0.8739	0.87
	Local	**0.9916**	**0.99**	0.8908	0.89
FAST+BRIEF	Global	**0.9916**	**0.99**	0.9748	0.97
	Local	**0.9916**	**0.99**	0.9496	0.95
FAST+BRISK	Global	0.9580	0.96	0.8992	0.90
	Local	0.9664	0.97	0.9328	0.93
FAST+SIFT	Global	**0.9916**	**0.99**	0.9832	0.98
	Local	0.9832	0.98	0.9832	0.98
ORB+ORB	Global	0.9153	0.91	0.8559	0.85
	Local	0.8898	0.89	0.7966	0.79
SIFT+SIFT	Global	0.9412	0.94	0.8739	0.87
	Local	0.9580	0.96	0.8739	0.87

Optimización y validación del modelo final. El análisis de sensibilidad de la cantidad de centroides (Figura 9) para la combinación FAST+BRIEF con normalización local y SVM mostró que el rendimiento máximo se logra con 500 centroides (**k=500**). La evaluación final del modelo optimizado se realizó mediante una validación cruzada estratificada de 5 particiones, que arrojó una **precisión promedio de 99.16 % ± 0.54 %**, con resultados individuales de [99.16 %, 99.16 %, 99.16 %, 100 %, 98.31 %]. Adicionalmente, una búsqueda de hiperparámetros (Tabla 2) identificó que los parámetros óptimos para el SVM son C=10, kernel='rbf', y gamma='scale', alcanzando un score de validación cruzada de 99.16 %.

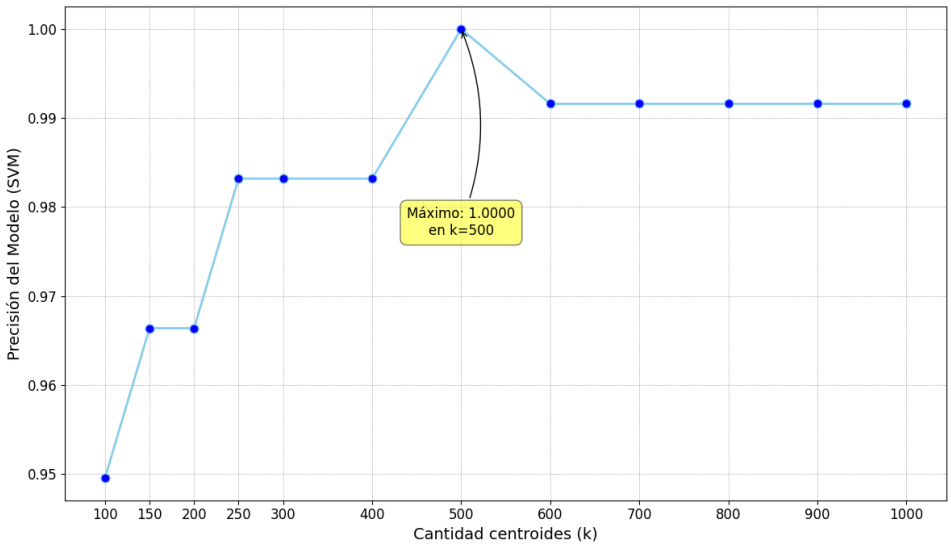

Figura 9: Precisión del modelo SVM en función de la cantidad de centroides (k).

La curva de aprendizaje del modelo final, mostrada en la Figura 10, es un indicador clave de su comportamiento. La *precisión en el entrenamiento* (curva roja) alcanza rápidamente valores cercanos al 100 %, lo que indica que el modelo tiene la capacidad suficiente para memorizar el conjunto de entrenamiento. Más importante aún, los valores obtenidos al aplicar *validación cruzada* (curva verde) a los datos de test también convergen a un valor muy alto y se mantienen muy cerca de los valores de entrenamiento. La pequeña brecha entre ambas curvas sugiere que el modelo no sufre de sobreajuste y generaliza bien a datos no vistos. La convergencia de ambas curvas con un número relativamente bajo de ejemplos (aproximadamente 100-200) indica que el modelo es eficiente en el aprendizaje de los patrones. Finalmente, la matriz de confusión (Figura 11) confirma el rendimiento casi ideal, con una única clasificación incorrecta en 119 secuencias.

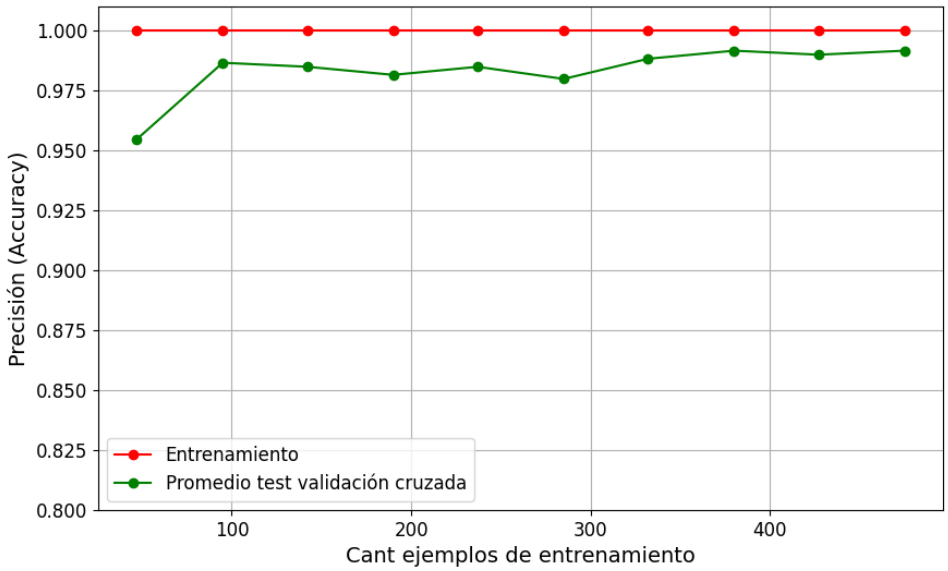

Figura 10: Curva de aprendizaje para el modelo final.

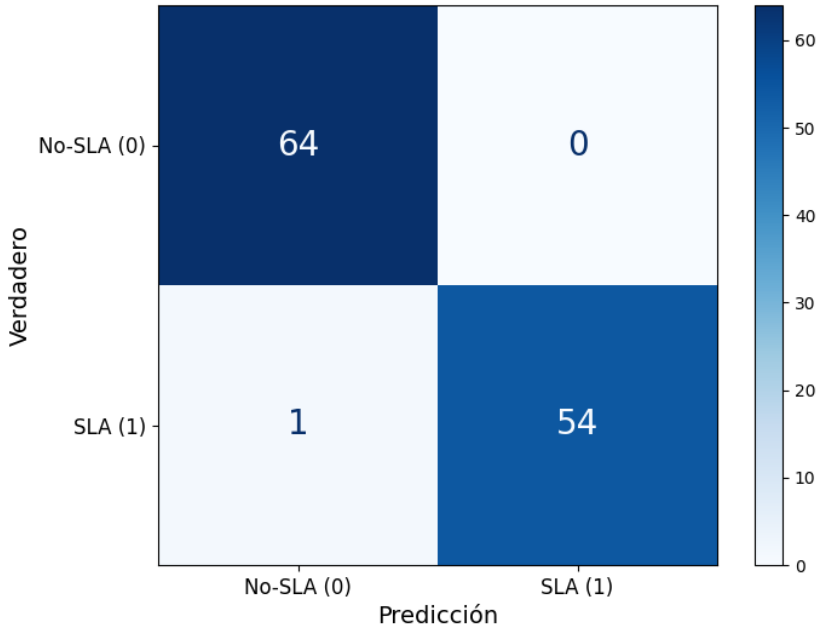

Figura 11: Matriz de confusión representativa para el modelo final.

Cuadro 2: Resultados de la búsqueda de hiperparámetros para el clasificador SVM.

Parámetro C	**Parámetro Gamma**	**Kernel**	**Puntuación Media (CV)**	**Desvío Std**	**Ranking**
100.0	scale	rbf	0.991568	0.005360	1
10.0	scale	rbf	0.991568	0.005360	1
1.0	scale	rbf	0.989887	0.006319	3
10.0	0.1	rbf	0.989887	0.009820	3
100.0	0.01	rbf	0.988207	0.008587	5
100.0	0.1	rbf	0.984845	0.011153	6
100.0	auto	rbf	0.984831	0.009844	7
1.0	0.1	rbf	0.981470	0.012374	8
10.0	0.01	rbf	0.981470	0.012374	8
0.1	scale	rbf	0.959593	0.017147	10
10.0	auto	rbf	0.693648	0.029914	11
1.0	0.01	rbf	0.538727	0.001823	12
0.1	auto	rbf	0.538727	0.001823	12
1.0	auto	rbf	0.538727	0.001823	12
0.1	0.01	rbf	0.538727	0.001823	12
0.1	0.1	rbf	0.538727	0.001823	12

5. Conclusiones

Reflexión general. Este proyecto ha validado con éxito un enfoque híbrido que fusiona el aprendizaje de representaciones de modelos fundacionales de ARN con la eficiencia de la visión por computadora clásica para resolver un problema de clasificación bioinformática. Se ha demostrado que re-conceptualizar los embeddings de secuencias como imágenes es una estrategia viable.

Logros. El logro principal es la construcción de un sistema completo, reproducible y preciso (98.99 % de precisión promedio en validación cruzada) para la detección de estructuras SLA. Se ha realizado un análisis comparativo y una validación robusta que demuestra que la combinación de RNA-FM, ecualización, FAST+BRIEF, BoVW y un clasificador SVM con kernel RBF constituye una solución de alto rendimiento y computacionalmente eficiente.

Aprendizajes. Una lección importante es que la innovación en el aprendizaje automático no siempre reside en la creación de arquitecturas de red más complejas, sino a menudo en la representación creativa de los datos. La transformación de un problema de secuencias a uno de imágenes permitió aplicar un conjunto de herramientas bien establecidas y computacionalmente eficientes para lograr un alto rendimiento. Además, se destaca la importancia del preprocesamiento, como la ecualización de histograma, para la implementación de los algoritmos de visión por computadora.

6. Bibliografía

Referencias

[1] Lodeiro, M.F., Filomatori, C.V. y Gamarnik, A.V. (2009). «Structural and functional studies of the promoter element for dengue virus RNA replication». En: *Journal of Virology*, 83(2), 993-1008. DOI: 10.1128/JVI.01647-08.

[2] Filomatori, C.V. et al. (2011). «RNA sequences and structures required for the recruitment and activity of the dengue virus polymerase». En: *Journal of Biological Chemistry*, 286(9), 6929-6939. DOI: 10.1074/jbc.M110.162289.

[3] Bujalowski, P.J., Bujalowski, W. y Choi, K.H. (2017). «Interactions between the dengue virus polymerase NS5 and stem-loop A». En: *Journal of Virology*, 91(11), e00047-17. DOI: 10.1128/JVI.00047-17.

[4] Sun, Y.T. y Varani, G. (2022). «Structure of the dengue virus RNA promoter». En: *RNA*, 28(9), 1210-1223. DOI: 10.1261/rna.079197.122.

[5] Lee, E. et al. (2021). «Structures of flavivirus RNA promoters suggest two binding modes with NS5 polymerase». En: *Nature Communications*, 12, 2530. DOI: 10.1038/s41467-021-22846-1.

[6] Obi, J.O. et al. (2024). «Structural dynamics of the dengue virus non-structural 5 (NS5) interactions with promoter stem loop A (SLA)». En: *bioRxiv preprint*. DOI: 10.1101/2024.12.03.626708.

[7] Choi, K.H. (2021). «The Role of the Stem-Loop A RNA Promoter in Flavivirus Replication». En: *Viruses*, 13(6), 1107. DOI: 10.3390/v13061107.

[8] Lorenz, R. et al. (2011). «ViennaRNA Package 2.0». En: *Algorithms for Molecular Biology*, 6(1), 26. DOI: 10.1186/1748-7188-6-26.

[9] Zuker, M. (2003). «Mfold web server for nucleic acid folding and hybridization prediction». En: *Nucleic Acids Research*, 31(13), 3406-3415. DOI: 10.1093/nar/gkg595.

[10] Singh, J. et al. (2019). «RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning». En: *Nature Communications*, 10(1), 5407. DOI: 10.1038/s41467-019-13395-9.

[11] Franke, J.K.H. et al. (2024). «RNAformer: A simple yet effective model for homology-aware RNA secondary structure prediction». En: *bioRxiv preprint*. DOI: 10.1101/2024.02.12.579881.

[12] Liu, J. et al. (2024). «Advancing bioinformatics with large language models: components, applications and perspectives». En: *arXiv preprint arXiv:2401.04155*. DOI: 10.48550/arXiv.2401.04155.

[13] Wang, Z. et al. (2025). «Large Language Models in Bioinformatics: A Survey». En: *arXiv preprint arXiv:2503.04490*. DOI: 10.48550/arXiv.2503.04490.

[14] Penić, R.J. et al. (2024). «RiNALMo: General-Purpose RNA Language Models Can Generalize Well on Structure Prediction Tasks». En: *arXiv preprint arXiv:2403.00043*. DOI: 10.48550/arXiv.2403.00043.

[15] Yin, W. et al. (2024). «ERNIE-RNA: An RNA Language Model with Structure-enhanced Representations». En: *bioRxiv preprint*. DOI: 10.1101/2024.03.17.585376.

[16] Zablocki, L.I. et al. (2025). «Comprehensive benchmarking of large language models for RNA secondary structure prediction». En: *arXiv preprint arXiv:2410.16212*. DOI: 10.48550/arXiv.2410.16212.

[17] Chaturvedi, M., Rashid, M.A. y Paliwal, K.K. (2025). «Transformers in RNA structure prediction: A review». En: *Computational and Structural Biotechnology Journal*, 27, 1187-1203. DOI: 10.1016/j.csbj.2025.03.021.

[18] Chen, J. et al. (2022). «Interpretable RNA Foundation Model from Unannotated Data for Highly Accurate RNA Structure and Function Predictions». En: *arXiv preprint arXiv:2204.00300*. DOI: 10.48550/arXiv.2204.00300.

[19] Rosten, E. y Drummond, T. (2006). «Machine learning for high-speed corner detection». En: *European conference on computer vision*, 430-443. DOI: 10.1007/11744023_34.

[20] Calonder, M., Lepetit, V., Strecha, C. y Fua, P. (2010). «BRIEF: Binary Robust Independent Elementary Features». En: *European Conference on Computer Vision*, 778-792. DOI: 10.1007/978-3-642-15561-1_56.

[21] Rublee, E., Rabaud, V., Konolige, K. y Bradski, G. (2011). «ORB: An efficient alternative to SIFT or SURF». En: *2011 International conference on computer vision*, 2564-2571. DOI: 10.1109/ICCV.2011.6126544.

[22] Leutenegger, S., Chli, M. y Siegwart, R.Y. (2011). «BRISK: Binary robust invariant scalable keypoints». En: *2011 International conference on computer vision*, 2548-2555. DOI: 10.1109/ICCV.2011.6126542.

[23] Csurka, G., Dance, C.R., Fan, L., Willamowski, J. y Bray, C. (2004). «Visual categorization with bags of keypoints». En: *Workshop on Statistical Learning in Computer Vision, ECCV*. url: https://www.researchgate.net/.../Visual_categorization_with_bags_of_keypoints.