

UNA PROPUESTA CONEXIONISTA PARA EL
RECONOCIMIENTO Y PREDICCIÓN DE PROMOTORES
EN SECUENCIAS DE ADN DE PROCARIOTAS

Tesista
Viviana Erica Cotik

Director
Dr. Jorge Sergio Igor Zwir

UNIVERSIDAD DE BUENOS AIRES
FACULTAD DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE COMPUTACION

Indice general

| | |
|--|-----------|
| Indice de tablas | IV |
| Indice de figuras | VI |
| Introducción | 1 |
| 1. Introducción biológica y conceptos de bioinformática | 6 |
| 1.1. Nociones biológicas básicas | 7 |
| 1.2. El proceso de transcripción | 14 |
| 1.2.1. La ARN polimerasa bacteriana | 18 |
| 1.2.2. El promotor | 18 |
| 1.2.3. Observaciones | 23 |
| 1.3. Introducción a conceptos de bioinformática | 24 |
| 1.3.1. Alineamiento de secuencias | 24 |
| 1.3.2. Modelos de alineamiento | 24 |
| 1.4. Compilaciones de datos de promotores | 28 |
| 2. Métodos computacionales para el reconocimiento de promotores | 32 |
| 2.1. Métodos fijos | 33 |
| 2.2. Métodos estadísticos | 34 |
| 2.3. Redes neuronales | 39 |
| 2.4. Comparación entre los distintos métodos | 46 |
| 2.5. Medidas de calidad de las predicciones | 47 |
| 2.6. Breve revisión de soluciones existentes basadas en redes neuronales | 55 |
| 3. CPR: un método conexionista para el reconocimiento de promotores | 59 |
| 3.1. Problema | 60 |
| 3.2. Método | 61 |
| 3.3. Tuning | 71 |

| | |
|--|------------|
| 4. Comparación del CPR con otros métodos computacionales para el reconocimiento de promotores | 79 |
| 4.1. Problema | 80 |
| 4.2. Métodos fijos | 85 |
| 4.2.1. Experimentación | 86 |
| 4.2.2. Resultados | 87 |
| 4.2.3. Análisis de resultados | 88 |
| 4.3. Método Consensus-Patser | 89 |
| 4.3.1. Experimentación | 91 |
| 4.3.2. Resultados | 93 |
| 4.3.3. Análisis de resultados | 103 |
| 4.4. Conexionist Promoter Recognition | 105 |
| 4.4.1. Experimentación | 105 |
| 4.4.2. Resultados | 106 |
| 4.4.3. Análisis de resultados | 110 |
| 4.5. Comparación de los distintos métodos | 110 |
| 4.5.1. Resultados de los distintos métodos | 110 |
| 4.5.2. Análisis de resultados de los distintos métodos | 111 |
| 5. CPR-MOSS: una metodología de dos niveles para la identificación de múltiples promotores | 113 |
| 5.1. Problema | 114 |
| 5.2. Metodología CPR-MOSS | 115 |
| 5.2.1. Introducción a conceptos de algoritmos evolutivos y optimización multiobjetivo | 117 |
| 5.2.2. MOSS | 120 |
| 5.3. Experimentación | 124 |
| 5.4. Resultados | 125 |
| 5.5. Aplicación Desarrollada | 126 |
| 6. Conclusiones y trabajo futuro | 128 |
| A. Comparación de medidas de calidad de las predicciones | 132 |
| B. Datos | 134 |
| C. Siglas y Abreviaturas | 142 |
| Glosario de términos biológicos | 144 |
| Referencias | 149 |

Indice de tablas

| | | |
|-------|--|-----|
| 1.1. | Diferencias entre organismos eucariotas y procariotas. | 12 |
| 1.2. | Grado de conservación de las regiones consenso según las compilaciones de Hawley y McClure, Harley y Reynolds, y Lisser y Margalit. | 30 |
| 2.1. | Tabla de contingencia de 2x2. | 49 |
| 2.2. | Medidas de calidad de predicciones. | 54 |
| 3.1. | SCCs de los conjuntos de entrenamiento y test para distintas relaciones positivos:negativos utilizadas en el conjunto de entrenamiento. | 72 |
| 3.2. | Resultados de distintas inicializaciones de la red neuronal | 75 |
| 4.1. | Resultados de las seis subsecuencias de longitud 46 de RegulonDB en que el promedio de la red neuronal dio más alto. | 82 |
| 4.2. | Métodos fijos - Porcentajes de FP, TP y SCC en las pruebas realizadas con el conjunto de test. | 87 |
| 4.3. | Resumen de las ejecuciones realizadas con Consensus. | 92 |
| 4.4. | Matriz de alineamiento resultante de las ejecuciones de Consensus con secuencias de longitud 46 | 95 |
| 4.5. | Frecuencias esperadas de las matrices de alineamiento resultantes de las ejecuciones de Consensus con secuencias de longitud 46 | 96 |
| 4.6. | Resultados de la ejecución número 4 de Consensus | 97 |
| 4.7. | Resultados de la ejecución número 5 de Consensus | 97 |
| 4.8. | Resultados de la ejecución número 6 de Consensus | 97 |
| 4.9. | Resumen de matrices elegidas de cada ejecución de Consensus | 98 |
| 4.10. | Resultados de las ejecuciones de Patser. | 98 |
| 4.11. | Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de test. Secuencias de longitud 46 | 101 |
| 4.12. | Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de test. Secuencias de longitud 56. | 102 |
| 4.13. | Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de entrenamiento | 104 |

| | |
|--|-----|
| 4.14. Resultados de la aplicación del método Consensus-Patser al conjunto de test para 0 %, 1 %, 5 % de FP y 80 % y 100 % de TP. | 105 |
| 4.15. Parámetros utilizados en el entrenamiento de la red neuronal. | 106 |
| 4.16. Red neuronal - Porcentajes de TP, FP y SCCs para distintos umbrales en el conjunto de test | 107 |
| 4.17. Red neuronal - Porcentajes de TP, FP y SCCs para distintos umbrales en el conjunto de entrenamiento | 108 |
| 4.18. Resumen de los resultados de la red neuronal con los conjuntos de entrenamiento y test | 109 |
| 4.19. Resultados de la aplicación de los métodos CPR y Consensus-Patser (C-P) al conjunto de test para 0 %, 1 %, 5 % de FP y 80 % y 100 % de TP. | 110 |
| 4.20. Máximos SCCs alcanzados por los métodos CPR, Consensus-Patser y métodos fijos | 111 |
| 5.1. Parámetros para la ejecución del MOSS. | 125 |
| 5.2. Resultados de la aplicación de la metodología CPR-MOSS a las secuencias de la compilación de Harley y Reynolds | 126 |
| B.1. Códigos ambiguos de IUPAC-IUB. | 134 |
| B.2. Conjunto de entrenamiento de la red neuronal. Provenientes de la compilación de Harley y Reynolds. | 137 |
| B.3. Conjunto de test de la red neuronal. Provenientes de la compilación de Harley y Reynolds. | 139 |
| B.4. Distribución de datos de la base RegulonDB en los conjuntos de entrenamiento y test. Utilizados para el reentrenamiento de la red neuronal. | 141 |

Índice de figuras

| | |
|--|----|
| 1.1. Composición química de bases púricas y pirimídicas | 9 |
| 1.2. Composición química de los azúcares presentes en los ácidos nucleicos | 9 |
| 1.3. Composición química de los nucleótidos que componen el ADN. | 10 |
| 1.4. Uniones puente hidrógeno entre las bases A-T y C-G. | 11 |
| 1.5. Doble hélice de ADN. | 11 |
| 1.6. Esquema del dogma central de la biología molecular | 15 |
| 1.7. Síntesis de una molécula de ARN | 16 |
| 1.8. La orientación de la ARN polimerasa determina cuál de las dos cadenas de ADN actuará como molde | 17 |
| 1.9. Direcciones de transcripción en una porción de un cromosoma bacteriano. | 17 |
| 1.10. El operón <i>lac</i> | 17 |
| 1.11. La ARN polimerasa de <i>E. coli</i> | 19 |
| 1.12. Esquema del promotor del gen <i>rrnB</i> | 20 |
| 1.13. Elementos en promotores reconocidos por la holoenzima con σ^{70} | 21 |
| 1.14. Consensos de promotores reconocidos por distintos factores σ | 22 |
| 1.15. Promotor fuerte <i>recA</i> y promotor débil <i>araBAD</i> | 23 |
| 1.16. Alineamiento de dos secuencias. | 24 |
| 1.17. Secuencias consenso definidas para las regiones -10 y -35 en un alineamiento | 25 |
| 1.18. Matrices de alineamiento y de pesos | 27 |
| | |
| 2.1. Pseudocódigo de los métodos fijos. | 34 |
| 2.2. Algoritmo Consensus. | 36 |
| 2.3. Ejemplo de ejecución del algoritmo Consensus | 37 |
| 2.4. Algoritmo Patser. | 38 |
| 2.5. Esquema de una neurona | 39 |
| 2.6. Esquema del modelo matemático de una neurona | 40 |
| 2.7. Función de activación logística. | 41 |
| 2.8. Arquitecturas de redes neuronales | 42 |
| 2.9. Diagrama explicativo del paradigma de aprendizaje supervisado | 43 |
| 2.10. Mínimos locales y globales en una función de error | 45 |

| | |
|--|-----|
| 2.11. Ejemplos de memorización y generalización en una red | 46 |
| 3.1. Detección de <i>features</i> independientemente de su posición en la entrada. | 61 |
| 3.2. TDNN de 2 capas | 62 |
| 3.3. Ejemplo de representación de los datos de entrada de la red | 64 |
| 3.4. Arquitectura de la red neuronal construida | 65 |
| 3.5. Algoritmo de aprendizaje <i>time delayed back-propagation</i> | 68 |
| 3.6. Gráfico del error cuadrático medio con los conjuntos de entrenamiento y validación. | 69 |
| 3.7. TDNN - Error cuadrático medio para datos positivos y negativos en distintas relaciones positivos:negativos | 73 |
| 3.8. Error cuadrático medio en función de las épocas con el conjunto de entrenamiento para distintos valores de η | 74 |
| 3.9. Diagrama de los pesos resultantes del mejor entrenamiento del perceptrón simple para el reconocimiento de la región -10 | 74 |
| 3.10. SCCs para distintos umbrales, resultantes de entrenar redes inicializadas con distintos valores en las distintas capas. Resultados de los conjuntos de entrenamiento y test. | 75 |
| 3.11. Pesos resultantes del entrenamiento de la red neuronal - Reconocimiento de subsecuencias conservadas | 76 |
| 3.12. Pesos resultantes del entrenamiento de la red neuronal - Reconocimiento de distancias | 77 |
| 4.1. Subsecuencia de RegulonDB utilizada | 81 |
| 4.2. Resultados para distintas subsecuencias | 82 |
| 4.3. Esquema del método Consensus-Patser | 90 |
| 4.4. Alineamiento. | 91 |
| 4.5. SCCs obtenidos en las distintas ejecuciones de Patser | 99 |
| 4.6. Proporciones de TP, SCCs y FP para distintos umbrales en el conjunto de test. Secuencias de Longitud 46. | 100 |
| 4.7. Proporciones de TP, SCCs y FP para distintos umbrales en el conjunto de test. Secuencias de Longitud 56. | 100 |
| 4.8. Red neuronal - SCCs y proporciones de TP y FP para los distintos umbrales con el conjunto de test. | 109 |
| 4.9. Red neuronal - SCCs y proporciones de TP y FP para los distintos umbrales con el conjunto de entrenamiento. | 109 |
| 5.1. Promotor <i>colE1-B</i> y las regiones TATAAT y TTGACA originales y alternativas marcadas por Harley y Reynolds | 114 |
| 5.2. Esquema de la metodología CPR-MOSS | 116 |

| | | |
|------|---|-----|
| 5.3. | Diagrama de flujo de un algoritmo evolutivo básico. | 118 |
| 5.4. | Población con cinco soluciones. | 120 |
| 5.5. | Gráfico de la función objetivo f_3 | 122 |
| 5.6. | Ejemplo de la representación de un individuo | 122 |
| 5.7. | Búsqueda local. | 123 |
| 5.8. | Algoritmo MOSS. | 124 |
| 5.9. | Distintas soluciones para el promotor ampC/C16 | 126 |
| A.1. | Comparación de distintos indicadores | 133 |

Agradecimientos

Deseo expresar mi agradecimiento a todas aquellas personas que colaboraron en la realización de este trabajo. Entre ellas se encuentran: Laura Ación, por sus valiosos comentarios y por haberme facilitado el acceso a la bibliografía. Diego Bendersky, por sus valiosos aportes y su paciencia. Natalia Bettini, por permitirme el acceso a su biblioteca. Diego Laplagne y Diana Posadas, por su generosidad al responder mis consultas del área de la biología molecular. Irene Loiseau, por el apoyo brindado para la concreción de este trabajo. Mi compañero de cuarto, Cristian Rocha, por su amabilidad. Rocío Romero Saliz, por su buena disposición. Mis padres, por la colaboración prestada, y mi director de tesis, Igor Zwir, por ayudarme a concretarlo.

Este trabajo ha sido realizado en el marco del proyecto estratégico UBASoft, de la Universidad de Buenos Aires.

Resumen

Uno de los grandes desafíos de la era postgenómica es la identificación de factores relacionados con la regulación de la expresión genética. La iniciación de la transcripción es el primer paso en la expresión genética y su regulación constituye un significativo punto de control de este fenómeno. El promotor, la secuencia reconocida por la ARN polimerasa para iniciar la transcripción, determina la ubicación del sitio de inicio de la transcripción y es un elemento importante para establecer su frecuencia de iniciación. Por lo tanto, la identificación de los promotores es crucial para detectar comportamientos regulatorios o *pathways* genéticos. Los promotores de organismos procariotas presentan subsecuencias conservadas, pero la definición de los nucleótidos que las componen es vaga y la distancia entre las mismas es variable, lo que hace que su reconocimiento mediante técnicas computacionales en cadenas de ADN no sea sencillo.

En este trabajo proponemos un método para resolver el problema de reconocimiento de promotores de organismos procariotas mediante el uso de una red neuronal *feedforward* con retardo temporal o *Time Delay Neural Network* (TDNN). La ventaja de esta red es que, mediante la técnica de *weight sharing* y la utilización de un algoritmo de *back-propagation* modificado, logra realizar el reconocimiento de las subsecuencias conservadas de los promotores separadas por distancias variables. Utilizamos los datos de la compilación más reciente de promotores de *E. coli*, almacenados en la base de datos RegulonDB¹ y fundamentados a partir de artículos referenciados en PubMed². Para la implementación del método propuesto, al que denominamos *Conexionist Promoter Recognition* (CPR), se realizó un análisis de los parámetros que influyen en la performance de la red, entre otros el armado de los conjuntos de entrenamiento y test, con el objetivo de optimizarlos. Estudiamos distintas medidas de calidad de las predicciones y proponemos una alternativa. Analizamos y evaluamos otros métodos existentes para la predicción de promotores y nuestro método obtuvo los mejores resultados. Por último, proponemos un modelo jerárquico de dos niveles, CPR-MOSS, compuesto por el método CPR y por un algoritmo evolutivo multiobjetivo, denominado *Multiobjective Scatter Search* (MOSS), para permitir identificar las regiones conservadas en los promotores predichos por la red. La modularidad de la red y el ajuste de los parámetros realizado permiten extender la solución encontrada a otros problemas similares, como el reconocimiento de promotores en eucariotas. La implementación del modelo CPR-MOSS está accesible en <http://soar-tools.wustl.edu>.

¹http://www.cifn.unam.mx/Computational_Genomics/regulondb/.

²<http://www.ncbi.nlm.nih.gov/PubMed/>.

Abstract

One of the big challenges of the post genomic era is the identification of factors related with the regulation of gene expression. Initiation of transcription is the first step in gene expression and its regulation constitutes an important point of control of this phenomenon. The sequence recognized by RNA polymerase to initiate the transcription, termed promoter, determines the location of the transcription start site and is an important element in determining the frequency of transcription initiation. Therefore, promoter recognition is crucial for detecting regulatory mechanisms or genetic pathways. Prokaryotic promoters present conserved subsequences. However, the vague definition of their nucleotide composition and the variability of the distance between them, make the promoter recognition task by computational methods a difficult one.

In this work, we propose a method based on a Time Delay Neural Network (TDNN) to solve the prokaryotic promoter recognition problem. This feedforward neural network achieves the recognition of the promoter conserved subsequences, through the use of weight sharing and a modified back-propagation algorithm. Data of the most recent *E. coli* promoter compilation available, stored in RegulonDB³, was used. This database contains information gathered from the literature and referenced by PubMed⁴. The proposed method, called Conexionist Promoter Recognition (CPR) involves an analysis of the parameters that influence the neural network performance, between others, the construction of training and test sets, in order to optimize them. Different measures of prediction performance were studied, and we propose an alternative. We analyzed and evaluated other existent methods for the promoter recognition task, and the best results were obtained with our method. Finally, we propose CPR-MOSS, a two step hierarchical model, based on CPR and a multiobjective evolutionary algorithm, termed *Multiobjective Scatter Search* (MOSS). This method identifies conserved sequences within the promoter regions predicted by the TDNN. The network modularity and the tuning performed enable the extension of our solution to similar problems, such as eukaryotic promoter recognition. CPR-MOSS is available for public use in <http://soar-tools.wustl.edu>.

³http://www.cifn.unam.mx/Computational_Genomics/regulondb/.

⁴<http://www.ncbi.nlm.nih.gov/PubMed/>.

Introducción

I can't be as confident about computer science as I can about biology.
Biology easily has 500 years of exciting problems to work on.

Donald E. Knuth

Desde 1953, año en que James Watson y Francis Crick propusieron el modelo de doble hélice del ADN, la biología molecular fue testigo de importantes avances. Particularmente, en los últimos años con proyectos como el del Genoma Humano⁵ se ha generado una gran cantidad de datos, cuyo procesamiento ha dado origen a un nuevo campo de estudio comúnmente denominado *biología molecular computacional* o *bioinformática* [71]. La bioinformática consiste en el uso de la matemática y la computación para resolver problemas de biología molecular.

Casi todas las células de un organismo multicelular contienen el mismo ADN. Sin embargo, la misma información genética produce una gran cantidad de células diferentes. La diferencia fundamental entre una neurona y una célula de la piel, por ejemplo, es que los genes que se expresan no son los mismos. Por lo tanto, el entendimiento de la regulación genética es un paso ineludible en el entendimiento del desarrollo [22]. Además, el conocimiento de los genes que normalmente se expresan en las células puede dar pistas acerca de enfermedades causadas por fallas en el circuito de control celular [21].

La iniciación de la transcripción es el primer paso en la expresión genética y su regulación es un medio significativo para el control de este fenómeno [64, 41]. El evento de iniciación comienza cuando la ARN polimerasa, la enzima que cataliza la producción de ARN a partir de un molde de ADN, reconoce una porción de la secuencia de ADN denominada promotor y se pega a la misma. Luego, la polimerasa inicia la síntesis del ARN a partir del apareamiento de bases complementarias. La secuencia del promotor es la que determina la ubicación y la orientación del extremo 5' del ARN y es un elemento importante para determinar la frecuencia de iniciación de

⁵Los proyectos de secuenciación, entre los que se encuentra el del Genoma Humano, lograron determinar experimentalmente las secuencias de ADN de muchos organismos.

la transcripción [64]. Por lo tanto, la identificación de los promotores es crucial para contribuir al entendimiento de la expresión genética, aportando algunas respuestas a las siguientes preguntas: ¿qué genes se expresan en una determinada célula en un momento determinado? [21], ¿cómo encuentra la ARN polimerasa a los promotores en el ADN? [41].

La conformación de la ARN polimerasa responsable del inicio de la transcripción en *Escherichia coli* (*E. coli*) se denomina *holoenzima* y está compuesta por dos elementos principales: el núcleo, formado por las subunidades $\alpha_2\beta\beta'$ y el factor sigma (σ). El factor σ interviene específicamente en el reconocimiento del promotor. Existen diferentes factores sigma que son utilizados para responder a cambios generados en el ambiente. En *E. coli* la mayor parte de los promotores son reconocidos por la holoenzima con factor σ^{70} [64, 41, 61, 3].

La comparación de promotores de *E. coli* llevó a la identificación de tres secuencias conservadas principales: la región -10 , la región -35 y la señal CAP [26, 25, 42]. Las regiones -10 y -35 están compuestas por secuencias de seis nucleótidos, nombradas de acuerdo a la posición aproximada de sus nucleótidos centrales relativos al sitio de inicio de la transcripción (TSS). La región -10 tiene consenso TATAAT y la región -35 tiene consenso TTGACA. Las regiones -10 y -35 se encuentran separadas por un espacio de entre 15 y 21 nucleótidos, aunque lo más común es un espacio de entre 16 y 18 nucleótidos [26, 25, 42]. Las regiones -10 y -35 definen al *core promoter* de *E. coli* [50].

El objetivo de este trabajo es desarrollar un método computacional para el reconocimiento y localización de promotores aún no detectados experimentalmente en cadenas de ADN de *E. coli* K12 reconocidas por la holoenzima σ^{70} . Para la realización del trabajo se utilizará la compilación de datos de promotores verificados experimentalmente⁶ de aparición más reciente, almacenados en RegulonDB [68]. Se pretende que la solución desarrollada sea fácilmente extensible para la obtención de soluciones que contemplen otro tipo de factores σ y organismos eucariotas y que tenga altas tasas de reconocimiento. Se pretende también poder reconocer las regiones conservadas del promotor. Los resultados de este trabajo deberían servirle al investigador del campo de la biología para tener una herramienta automática que le permita conocer los sitios del ADN en los cuales puede realizar experimentos con probabilidad alta de encontrar promotores.

El problema de encontrar un promotor en una cadena de ADN no es trivial. Si bien existen modelos de la estructura de los promotores, gran parte de los mismos no

⁶Los datos de promotores provienen de estudios experimentales, cuyas publicaciones están referenciadas en PubMed, <http://www.ncbi.nlm.nih.gov/PubMed/>.

resulta satisfactoria debido a que no puede tratar con la incertidumbre y ambigüedad de este problema biológico [64]. Se conoce la existencia de los hexámeros conservados, pero la secuencia de nucleótidos de estos motivos es variable y en general obedece a una distribución probabilística. No todos los promotores cuentan con ambos motivos [61], tampoco tienen exactamente estos motivos sino también extensiones de los mismos [61] y no todos los nucleótidos que los componen son igualmente importantes [25]. Además, tal como mencionamos anteriormente, las subsecuencias conservadas se encuentran a una distancia variable entre sí. Las características arriba mencionadas hacen que sea complejo modelar este problema.

Desde una perspectiva computacional, el descubrimiento y búsqueda de promotores puede ser abordado mediante distintos métodos. Los mismos incluyen:

Métodos fijos. Estos métodos buscan un patrón, donde cada uno de los nucleótidos que forma parte del mismo tiene igual peso o significación [80]. Debido a que la fuerza con la que la polimerasa se pega al ADN no es uniforme y varía en especificidad de acuerdo a la posición, estos métodos no parecen ser *a priori* la mejor solución.

Métodos estadísticos. Estos métodos generan matrices de frecuencia de aparición de los nucleótidos en cada una de las posiciones de un alineamiento de secuencias de ADN [30], considerando aspectos de teoría de la información y de estadística. Con estas técnicas se mejora la variabilidad en cuanto a los componentes de los motivos, sin embargo la variación de la distancia continúa siendo un problema [34].

Redes neuronales artificiales. Las redes neuronales artificiales⁷ incorporan el conocimiento mediante un proceso de aprendizaje basado en un conjunto de entrenamiento sin la necesidad de contar con un modelo preciso del problema a resolver. Existen arquitecturas de redes neuronales artificiales que permiten tratar el problema de variación de las distancias entre las secuencias conservadas.

A partir de los estudios existentes acerca de la estructura de los promotores en las cadenas de ADN de procariotas, proponemos un método basado en el diseño y entrenamiento de una red neuronal con retardo temporal o *Time Delay Neural Network* (TDNN) [81, 39] para la tarea de reconocimiento de promotores aún no detectados experimentalmente. Denominamos a este método *Reconocimiento Conexionista de Promotores* o *Conexionist Promoter Recognition* (CPR). La arquitectura construida tiene la propiedad de poder reconocer las subsecuencias conservadas de los promotores ubicadas a distancias variables entre sí. Se estudiaron varias características de la red y de los datos utilizados para el entrenamiento y test con el objetivo de mejorar su capacidad de aprendizaje y generalización. Inicialmente utilizamos los datos de la

⁷Las redes neuronales también son llamadas redes conexionistas. A partir de ahora las denominaremos indistintamente redes neuronales artificiales o redes neuronales.

compilación de Harley y Reynolds [25], que tienen la ventaja de tener marcadas las secuencias conservadas de las regiones -10 y -35 de los promotores. Con el objetivo de mejorar la calidad de la herramienta de predicción reentrenamos la red con los datos disponibles en la base de datos RegulonDB [68]. Esta es la compilación de datos de promotores verificados experimentalmente más actualizada disponible y está basada en compilaciones previas, en que se hizo una revisión exhaustiva de la corrección de los datos [42]. Las predicciones realizadas por la red neuronal resultaron mejores que las obtenidas mediante los métodos fijos y estadísticos. Para la comparación de los resultados de distintas configuraciones de la red neuronal, y entre los resultados de la red y de otras técnicas se propone una nueva métrica basada en el coeficiente de correlación [47], que resultó más adecuada que otras métricas [47, 12, 8].

Con el fin de poder realizar un mejor estudio de los promotores predichos por la red neuronal, al investigador del campo de la biología puede interesarle la posibilidad de identificar con exactitud las secuencias correspondientes a las regiones -10 y -35 . De esta forma se pueden hacer pruebas mediante mutaciones dirigidas. Además, más de un promotor puede intervenir en diferentes procesos de regulación en una misma región. Si bien se podrían realizar inferencias sobre la ubicación de las regiones -10 y -35 a partir de los pesos resultantes del entrenamiento de la red neuronal, se encuentran muchas posibles combinaciones *región -35 - distancia - región -10* y la elección de una de ellas no es sencilla. Con el objetivo de reducir la cantidad de posibles combinaciones, proponemos el uso de una metodología de dos niveles: CPR-MOSS [14]. El primero está formado por el método CPR y el segundo está basado en un algoritmo evolutivo, denominado *Multiobjective Scatter Search* (MOSS) [69], desarrollado a partir de la información deducida de las compilaciones de datos existentes [26, 25, 42].

La red implementada está disponible para uso público en <http://soar-tools.wustl.edu>. Este trabajo contó con la colaboración de M. Reese, del Departamento de Biología Molecular y Celular de la *University of California*, cuyo trabajo [63] y colaboración fueron tomados como punto de partida de esta investigación; del equipo de trabajo de RegulonDB, de la *Universidad Nacional Autónoma de México* (UNAM), que suministró los datos contenidos en RegulonDB [68] y del *Groisman Lab* de la *Washington University in St. Louis*, donde surgió la necesidad de estudio de este problema.

El trabajo está organizado de la siguiente manera: el Capítulo 1 introduce al lector en los conceptos utilizados en el desarrollo del mismo. Presenta nociones básicas de biología molecular y conceptos de bioinformática. El Capítulo 2 introduce métodos computacionales para la resolución del problema de reconocimiento de promotores. Se introducen conceptos de redes neuronales, distintas medidas de calidad de predicciones y se hace una breve revisión de las soluciones existentes basadas en redes

neuronales. El Capítulo 3 describe el método CPR, que incluye la aplicación de un método conexionista para la solución del problema de reconocimiento de promotores y la descripción de la arquitectura y las pruebas realizadas con el objeto de optimizar la performance de la red. En el Capítulo 4 se muestra el reentrenamiento realizado con la red obtenida en el Capítulo 3 para hacer uso de las nuevas compilaciones de datos existentes y se comparan los resultados de la red con los resultados de los otros métodos de predicción descritos en el Capítulo 2. En el Capítulo 5 se presenta la metodología CPR-MOSS para el reconocimiento de las secuencias conservadas en los promotores predichos por la red neuronal. Finalmente, se presentan en el Capítulo 6 las conclusiones de este trabajo y las posibles líneas de investigación que permitirían continuarlo en el futuro.

En el Apéndice A se muestran y analizan brevemente los resultados de la aplicación de las distintas medidas de calidad de las predicciones a los resultados de uno de los métodos predictivos. El Apéndice B muestra datos referenciados a lo largo del trabajo y el Apéndice C lista las siglas y abreviaturas utilizadas. El Glosario incluye los términos biológicos que creímos pertinentes.

Capítulo 1

Introducción biológica y conceptos de bioinformática

Desde 1953, año en que James Watson y Francis Crick propusieron el modelo de doble hélice del ADN, la biología molecular fue testigo de importantes avances. En los últimos años con proyectos como el del Genoma Humano se ha generado una gran cantidad de datos. El procesamiento y almacenamiento de los mismos con el fin de utilizarlos en avances científicos ha dado origen a un nuevo campo de estudio comúnmente denominado *biología molecular computacional* o *bioinformática* [71].

La bioinformática consiste en el uso de la matemática y la computación para resolver problemas de biología molecular. La necesidad de existencia de motores de bases de datos que permitan el almacenamiento y el acceso a la información biológica, de técnicas sofisticadas de reconocimiento de patrones, de realización de comparaciones de secuencias de nucleótidos o aminoácidos y de obtención de algoritmos de predicción de estructura tridimensional de proteínas, ejemplifican algunos aportes que estas disciplinas pueden brindarle a la biología.

Uno de los grandes desafíos de la era postgenómica¹ es la identificación de factores relacionados con la regulación de la expresión genética. La iniciación de la transcripción es el primer paso en la expresión genética y constituye un importante punto de control de este proceso. El promotor, la secuencia reconocida por la ARN polimerasa para iniciar la transcripción, determina la ubicación del sitio de inicio de la transcripción (TSS) y es un elemento importante para establecer la frecuencia de iniciación de la transcripción. Por lo tanto la identificación de los promotores es crucial para detectar comportamientos regulatorios o *pathways* genéticos.

En este trabajo trataremos de reconocer una región específica del promotor, denominada *core promoter*. Trabajamos con promotores de *Escherichia coli* (*E. coli*) K12 reconocibles por la subunidad σ^{70} de la ARN polimerasa, dado que son los que

¹Denominamos *era postgenómica* a la que sucede a los resultados de los proyectos genomas.

predominan y transcriben la mayor parte de los genes de esta bacteria [41, 61, 3].

En este capítulo presentamos una introducción biológica para poder explicar la utilidad y dificultad del descubrimiento de promotores. Se divide en cuatro secciones, en la Sección 1.1 se presentan nociones biológicas básicas. Primero se las describe funcionalmente y luego se hace una descripción más detallada. En la Sección 1.2 se explica el proceso de transcripción en procariotas. En la Sección 1.3 se presentan algunos conceptos de bioinformática que se utilizarán a lo largo del trabajo y, finalmente, en la Sección 1.4 se presenta una breve historia de las compilaciones y análisis de datos, que permitieron llegar al conocimiento que se tiene actualmente de los promotores de *E. coli*.

A lo largo del capítulo definiremos los términos que consideramos más importantes, otros pueden encontrarse en el glosario.

1.1. Nociones biológicas básicas

Las biomoléculas son las moléculas constituyentes de los seres vivos. Están formadas por cuatro elementos: hidrógeno, oxígeno, carbono y nitrógeno. Según su grado de complejidad estructural las biomoléculas pueden ser, entre otros, *unidades estructurales*, también llamadas *unidades constitutivas de macromoléculas*, como los aminoácidos (unidades constitutivas de las proteínas) y los nucleótidos (unidades constitutivas de los ácidos nucleicos), y *macromoléculas*, de peso molecular alto, como las proteínas y los ácidos nucleicos. A estas macromoléculas se las denomina polímeros y su formación mediante la unión de las unidades constitutivas (monómeros) se realiza a través de un proceso denominado *polimerización*.

La célula es la unidad mínima de un organismo capaz de actuar de manera autónoma. Todos los organismos vivos están formados por células. Algunos organismos microscópicos están compuestos por una sola célula, mientras que otros están formados por muchos millones de células organizadas en tejidos y órganos. Existen dos tipos de organización celular: la procariota y la eucariota. La primera, que corresponde a la estructura de las células de bacterias y cianobacterias es más primitiva que la segunda, que corresponde a la estructura de células de protistas, hongos, plantas y animales. La diferencia fundamental entre ambos tipos es que las células eucariotas tienen núcleo, mientras que las procariotas no tienen un núcleo separado del resto del medio celular por una membrana. También hay diferencias en cuanto a tamaño y organización interna.

Las proteínas son la “maquinaria” de la célula. Representan más de la mitad del peso seco de la mayor parte de las células y realizan casi todas las funciones bioquímicas del organismo [13]. Algunas proteínas tienen una función estructural, dando forma

a las células. Otras cumplen un papel funcional, como las proteínas contráctiles del músculo o las enzimas, que hacen posibles las reacciones químicas que tienen lugar en el organismo. Todas las proteínas están compuestas por las mismas unidades básicas: los aminoácidos.

Las proteínas se construyen a partir de secuencias de ADN y es por esto que la información que contiene el ADN es esencial para el organismo. El ADN no solamente codifica las proteínas, sino que también influye en la regulación de las proteínas a sintetizar. La porción contigua de ADN que contiene la información necesaria para formar una proteína se denomina *gen*. El pasaje de ADN a proteínas se lleva a cabo mediante los procesos de *transcripción* (en que se pasa de ADN a ARN) y *traducción* (en que se traduce el ARN en aminoácidos). Cuando una célula se divide el ADN se replica mediante un proceso denominado *replicación* de forma tal de que cada célula hija mantenga la misma información genética que la célula madre.

Casi todas las células de un organismo multicelular contienen el mismo ADN. Sin embargo la misma información genética, produce una gran cantidad de células diferentes. La diferencia fundamental entre una neurona y una célula de la piel, por ejemplo, es que los genes que se expresan no son los mismos. Por lo tanto, el entendimiento de la regulación genética es un paso importante en el entendimiento del desarrollo [22].

Nucleótidos

Los nucleótidos están compuestos por una base nitrogenada, una pentosa y un radical fosfato. Las **bases nitrogenadas** son compuestos con Carbono, Nitrógeno e Hidrógeno (C, N, H). Hay dos tipos: 1) bases **púricas**, derivadas de un compuesto denominado purina: éstas son la adenina y la guanina (A y G respectivamente) y 2) bases **pirimídicas**, derivadas de la pirimidina: citosina, timina y uracilo (C, T y U respectivamente). En la Figura 1.1 se puede ver la composición química de todas las bases. Las **pentosas** son azúcares. En los nucleótidos se encuentran la ribosa y la desoxirribosa. La composición química de los azúcares puede verse en la Figura 1.2. La unión covalente entre una base nitrogenada y una pentosa forma un **nucleósido**. El producto de la unión de un fosfato al nucleósido es un **nucleótido**. En la Figura 1.3 se puede ver la composición química de los nucleótidos que componen el ADN.

Ácidos nucleicos

Los ácidos nucleicos son polímeros, cuyos monómeros son los nucleótidos. Dependiendo de las pentosas que los constituyen pueden ser **ADN (ácido desoxirribonucleico)** o **ARN (ácido ribonucleico)**.

Como se puede ver en la Figura 1.2, la molécula de azúcar contiene cinco átomos de carbono, que se numeran de 1' a 5'. Dado que la unión que forma el ADN se

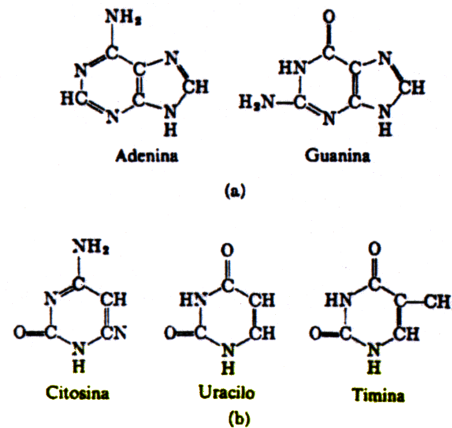


Figura 1.1: Composición química de bases púricas (a) y bases pirimídicas (b).

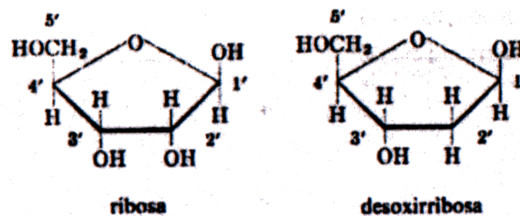


Figura 1.2: Composición química de los azúcares presentes en los ácidos nucleicos. La única diferencia entre ambos azúcares es el oxígeno en el carbono 2'.

hace entre el carbono 3' de un nucleótido y el residuo fosfato unido al carbono 5' del próximo nucleótido, las moléculas de ADN tienen una orientación que, por convención, comienza en el extremo 5' y termina en el 3'. Siempre que no se aclare lo contrario escribiremos las secuencias en la dirección 5' a 3'.

El ARN está constituido por una cadena simple de los nucleótidos A, C, G y U. Hay tres tipos básicos: el mensajero, el ribosomal y el de transferencia. El ADN está constituido por dos cadenas de polinucleótidos enfrentadas por sus bases nitrogenadas y unidas por enlaces de puente hidrógeno. Sólo pueden aparearse una base púrica con una pirimídica. A las bases que se aparean se las denomina bases complementarias. A se aparea con T y C con G. Entre las bases complementarias A y T se forman dos enlaces puente hidrógeno y entre C y G se forman tres² (ver Figura 1.4). Las cadenas son antiparalelas, lo que quiere decir que una está en dirección 5'-3' y la otra en

²Los enlaces puente hidrógeno mantienen unidas las bases complementarias. Cuantos más enlaces hay, más energía es necesaria para separarlas.

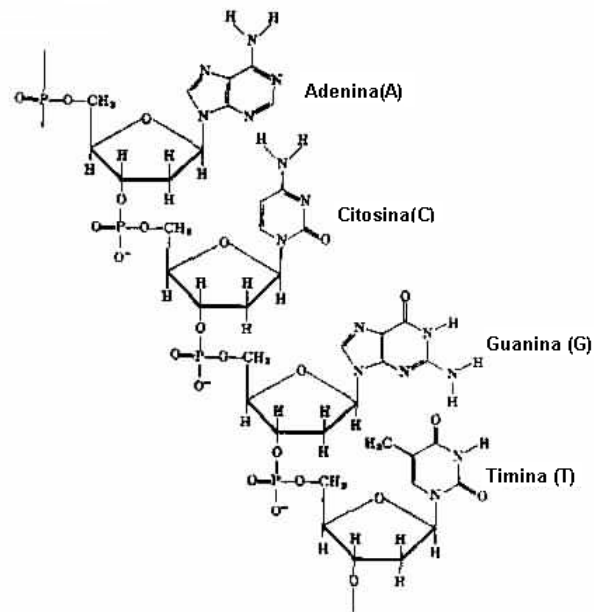


Figura 1.3: Composición química de los nucleótidos que componen el ADN.

dirección 3'-5', y adoptan una estructura helicoidal, denominada **doble hélice** (ver Figura 1.5).

Dado que cada nucleótido contiene una base, es habitual referirse a la longitud de las cadenas de ADN y ARN en cantidad de bases o de nucleótidos, y debido a que el ADN es una doble hélice, estas unidades son habitualmente denominadas *pares de bases* (pb). La longitud de las cadenas de ADN suelen medirse en miles o millones de pares de bases, abreviados kb y Mb respectivamente.

Proteínas

Las **proteínas** son polímeros. Los monómeros que las componen son los aminoácidos. Los **aminoácidos** son moléculas orgánicas compuestas por un ácido y una base (amina). Cada aminoácido está compuesto por tres nucleótidos. Existen 20 aminoácidos en la naturaleza. El código genético es un mapeo entre el nombre de un aminoácido y la cadena de nucleótidos que lo codifica.

Genoma

Los genes consisten en una región codificante y una región regulatoria. La región codificante es la parte del gen que codifica una proteína. La región regulatoria es la porción del ADN que contribuye al control de expresión del gen en respuesta a señales

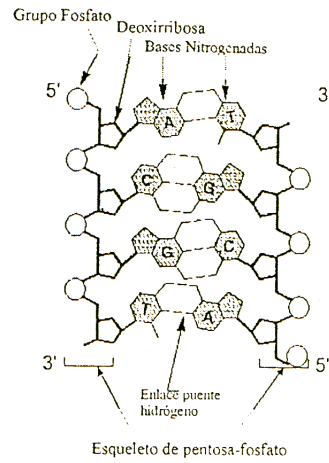


Figura 1.4: Uniones puente hidrógeno entre las bases A-T y C-G.

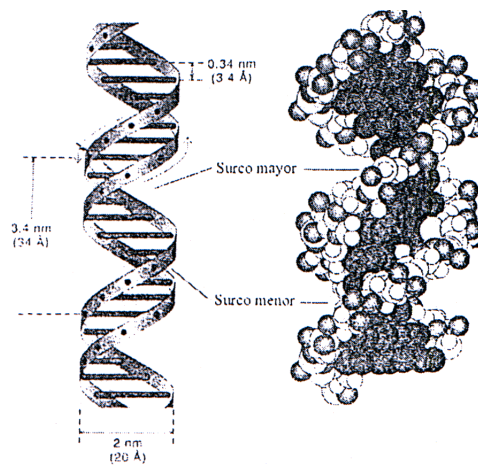


Figura 1.5: Doble hélice de ADN.

del medio celular. En procariotas la región codificante es contigua, pero en eucariotas la región codificante suele estar partida en porciones codificantes denominadas *exones* y en secuencias no codificantes denominadas *intrones*.

La secuencia completa de ADN de un organismo constituye el **genoma** del mismo. Los **cromosomas** son estructuras compuestas por una molécula larga de ADN y contienen numerosos genes.

Organismos procariotas y eucariotas

Los organismos vivos pueden dividirse en dos grandes grupos de acuerdo a la estructura de sus células: los procariotas y los eucariotas. La diferencia fundamental entre procariotas y eucariotas es que las células de los primeros no tienen un núcleo separado del resto del medio celular por una membrana. La complejidad en la transcripción es mayor en los eucariotas [21]. En la Tabla 1.1 se presentan las diferencias entre los organismos eucariotas y procariotas.

| | Procariotas | Eucariotas |
|-----------------------------|--|--|
| Organismos | bacterias y cianobacterias | protistas, hongos, plantas y animales |
| Tamaño celular | generalmente 1 a 10 μm en dimensión lineal (1 micrómetro (μm) equivale a 10^{-6} metros) | generalmente de 5 a 100 μm en dimensión lineal |
| Metabolismo | anaeróbico o aeróbico | aeróbico |
| Organelas | pocas o ninguna | núcleo, mitocondrias, retículo endoplasmático, cloroplastos, etc. |
| ADN | circular en citoplasma, también puede ser lineal | moléculas lineales, muy largas, con muchas regiones no codificantes, organizado en cromosomas y rodeado por la envoltura nuclear |
| ARN y proteína | ARN y proteína sintetizados en el mismo compartimiento | ARN sintetizado y procesado en el núcleo, proteínas en el citoplasma |
| Organización celular | principalmente unicelular | principalmente pluricelular, con diferenciación de muchos tipos celulares |

Tabla 1.1: Diferencias entre organismos eucariotas y procariotas.

Desde el punto de vista evolutivo se considera que los procariotas son antecesores de los eucariotas. Las células eucariotas suelen tener un volumen mucho mayor que las procariotas y contienen una cantidad proporcionalmente superior de la mayoría

de materiales nucleares; una célula humana, por ej., contiene 1000 veces más ADN que una bacteria típica [3].

Las **bacterias** son organismos principalmente unicelulares. Se multiplican por replicación de ADN y lo hacen en lapsos de tiempo muy cortos. *E. coli* es una bacteria que normalmente se halla en el intestino de los humanos y de otros animales. Es un organismo modelo muy utilizado para la investigación (en [36] se describen varios motivos de su uso como organismo de referencia para una gran cantidad de estudios en organismos procariontes). Su velocidad de reproducción permite que millones de bacterias puedan generarse y tratarse en el laboratorio [71]. El genoma de *E. coli* tiene 4.6 Mb y contiene 4288 genes de un tamaño medio de ~ 950 pb y una separación media entre los genes de 118 pb [41].

Bacteriófagos y plásmidos

Los virus son parásitos a nivel molecular. Se reproducen cuando infectan a una célula susceptible denominada huésped. No tienen metabolismo propio y utilizan el del huésped para replicarse.

Los **bacteriófagos** o **fagos** son virus que infectan a las bacterias inyectando su propio ADN al huésped bacteriano. Las formas en las que se propagan los fagos están determinadas por la regulación de la transcripción. El primer estadio de la expresión genética de los fagos depende del aparato de transcripción de la célula huésped. Generalmente sólo unos pocos genes se expresan en este estadio [41].

Los **plásmidos** son pequeñas moléculas de ADN circular que se encuentran en las bacterias en forma separada del cromosoma bacteriano. Todos los plásmidos son capaces de autoreplicarse. Se introducen fácilmente en la célula bacteriana. Esto ocurre en la naturaleza, pero también pueden ser introducidos *in vitro*, con el fin de realizar transformaciones en la célula.

Los plásmidos y bacteriófagos son muy utilizados como vectores en clonaje de ADN. Un **vector de clonaje** es un elemento genético usado para transportar un fragmento de ADN a una célula receptora con el propósito de clonar el gen introducido a medida que el ADN del organismo alterado se replica.

Cepas

Una cepa es una población de bacterias descendientes todas de una sola bacteria. *E. coli* tiene muchas cepas, entre las que se encuentran la K12, que es inofensiva y la O157:H7, que es virulenta.

Mutaciones

Las **mutaciones** son cambios en la secuencia de nucleótidos de un genoma. Cuando un cambio en la secuencia de un ADN provoca una alteración en la secuencia de una proteína se puede llegar a la conclusión de que ese ADN codifica esa proteína. Además, la modificación que aparezca en el fenotipo del organismo, permitirá identificar la función de dicha proteína [41]. Todos los organismos sufren cierto número de mutaciones como consecuencia de sus procesos celulares normales o de interacciones al azar con el entorno que los rodea. Estas mutaciones reciben el nombre de *espontáneas*. La frecuencia con la que se producen dependen de cada organismo. Cualquier par de bases de ADN puede sufrir mutaciones. Una **mutación puntual** es la que recae sobre un sólo nucleótido del ADN [3].

Las mutaciones también pueden ser *dirigidas*, i.e. generadas específicamente *in vitro* para construir genes nuevos con propiedades diseñadas a voluntad. Existen mutaciones de inserción, deleción y sustitución. Denominamos *inserción* al agregado de un nucleótido en una secuencia, *deleción* a la eliminación de un nucleótido en una secuencia y *sustitución* al cambio de un nucleótido por otro.

1.2. El proceso de transcripción

El dogma central de la biología molecular, representado en la Figura 1.6, dice que la información genética está almacenada en el ADN, es transcripta a ARN mensajero y luego traducida a proteínas. Si se considera que algunas proteínas regulan la transcripción, se ve que los factores de transcripción proveen una retroalimentación, en la que algunos genes regulan la expresión de otros genes [21].

La síntesis de proteínas depende de la colaboración entre distintos tipos de moléculas de ARN. Primero se copia una molécula de ARN mensajero (ARNm) a partir de la cadena de ADN que codifica la proteína que se va a sintetizar. Mientras tanto, en el citoplasma los aminoácidos a partir de los cuales se construirá la proteína se unen a su molécula específica de ARN de transferencia (ARNt). Luego, a medida que la molécula de ARNm se desplaza a través de los ribosomas (compuestos por proteínas y ARNr), la secuencia de nucleótidos es traducida a la correspondiente secuencia de aminoácidos, produciendo una cadena proteica determinada, especificada por la secuencia de ADN de su gen [2].

El proceso mediante el cual un gen da origen a una proteína se conoce como **expresión genética**. La síntesis de los distintos tipos de moléculas de ARN³ sobre

³El ARNm es el utilizado para la codificación de proteínas a partir de los procesos de transcripción y de traducción. El ARNr y el ARNt están codificados por genes y se generan a partir del mecanismo de transcripción (sin traducción).

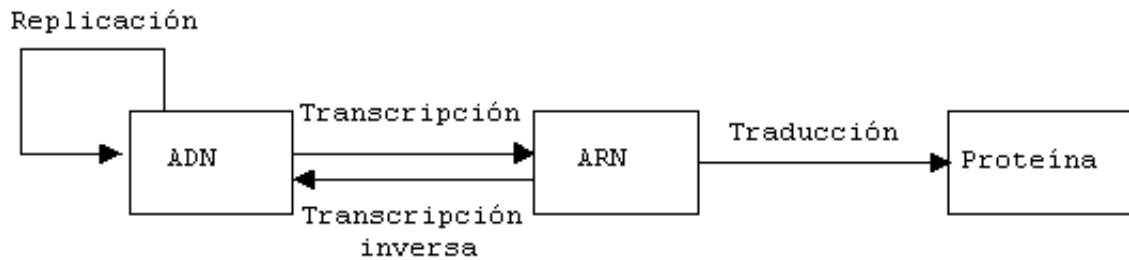


Figura 1.6: Esquema del dogma central de la biología molecular, que muestra el flujo de la información genética de las células. Los genes se perpetúan como secuencias de ácidos nucleicos, pero funcionan mediante su traducción a proteínas. La replicación es la responsable de la herencia de la información genética, mientras que la transcripción y la traducción hacen posible la conversión de ADN a ARN y de ARN a proteínas respectivamente.

el molde de ADN se denomina **transcripción del ADN** y es realizada por enzimas **ARN polimerasa**.

La ARN polimerasa bacteriana está formada por múltiples subunidades, asociadas con varias subunidades adicionales que se unen y se separan al complejo ADN-polimerasa en diferentes momentos de la transcripción. Las moléculas libres de ARN polimerasa colisionan aleatoriamente con el cromosoma bacteriano, uniéndose muy débilmente a la mayor parte del ADN. Sin embargo, la polimerasa se une fuertemente al ADN bacteriano cuando entra en contacto con una secuencia específica del ADN, denominada **promotor**, que señala la ubicación en donde debe iniciarse la síntesis del ARN [2].

Después de unirse al promotor, la ARN polimerasa abre una región localizada de la doble hélice, de forma que expone los nucleótidos de ambas cadenas de una pequeña zona de ADN. Una de las dos cadenas expuestas del ADN actúa como molde⁴ para el apareamiento de bases complementarias con monómeros de ribonucleósido trifosfato (ATP, GTP, CTP y UTP), dos de los cuales son unidos entre sí por la polimerasa y se inicia una cadena de ARN. La molécula de polimerasa avanza a lo largo del ADN, desenrollando la hélice lo necesario para exponer una nueva región de la cadena molde para el apareamiento de bases. De esta forma, la cadena de ARN va creciendo nucleótido a nucleótido en dirección 5' a 3'. El proceso de elongación de la cadena continúa hasta que la enzima encuentra una segunda secuencia especial del ADN, la

⁴La cadena que sirve de molde se denomina *cadena molde*, mientras que la complementaria se llama *cadena codificante*, ésta es idéntica en secuencia de bases con el ARN transcripto excepto que T es sustituida por U en el ARN. Por una cuestión de claridad, generalmente sólo se muestra la secuencia de nucleótidos de la cadena codificante, a pesar de que la ARN polimerasa reconoce al promotor como una doble cadena de ADN.

señal de terminación, en cuyo momento la polimerasa se detiene, liberándose tanto del ADN molde como de la cadena de ARN recién formada [2]. En la Figura 1.7 se puede ver esquemáticamente el proceso de síntesis de ARN.

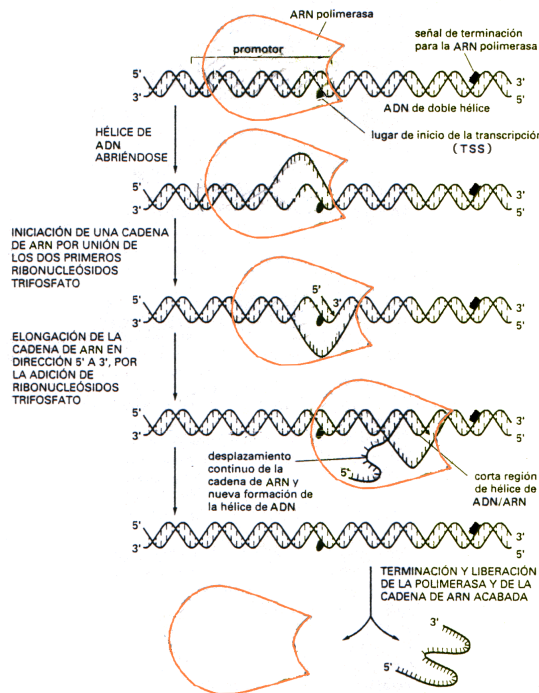


Figura 1.7: Síntesis de una molécula de ARN por la ARN polimerasa. La enzima se une al promotor de la secuencia de ADN e inicia la síntesis a partir de ésta. Continúa la síntesis hasta encontrar la señal de terminación. Tras ello, la polimerasa y la cadena completa de ARN se liberan. Gráfico adaptado de [2].

En cada región del ADN únicamente una de las dos cadenas se utiliza como molde. La orientación del promotor determina la dirección en que se mueve la polimerasa y ésta determina cuál de las dos cadenas de ADN será utilizada como molde para la síntesis de ARN (ver Figura 1.8). En genes vecinos la cadena de ADN copiada puede ser diferente o la misma [2] (como puede observarse en la Figura 1.9).

Los ARNm bacterianos pueden codificar una o muchas proteínas (se los denomina mono y policistrónicos respectivamente). En el último caso se transcribe un único ARNm que codifica para varios genes. Estos elementos se denominan **operones**. Un ejemplo de operón es el Lac (Figura 1.10) que codifica para los genes lacZ, lacY y lacA.

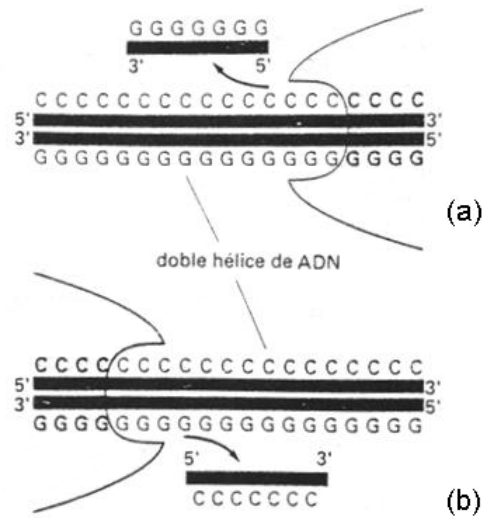


Figura 1.8: La orientación de la ARN polimerasa determina cuál de las dos cadenas de ADN actuará como molde. Si la ARN polimerasa se desplaza de derecha a izquierda, sintetiza un ARN utilizando como molde la cadena superior (en dirección 5'-3' del ADN). Si se desplaza de izquierda a derecha, la síntesis del ARN toma como molde la cadena inferior del ADN. Gráfico adaptado de [2].

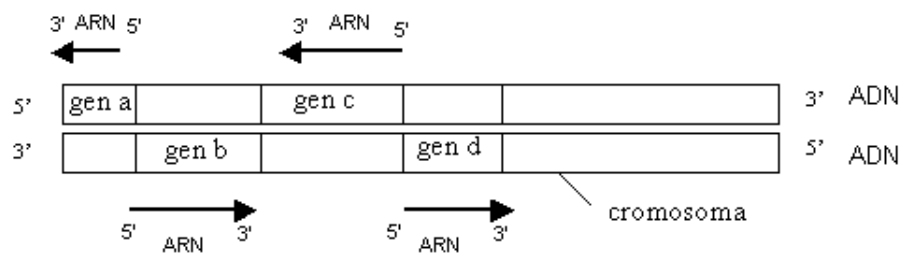


Figura 1.9: Direcciones de transcripción en una porción de un cromosoma bacteriano.

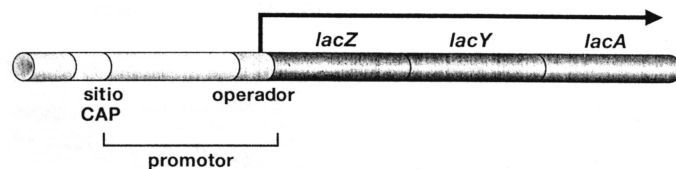


Figura 1.10: El operón *lac*. El gen *lacZ* es transcrito en una sola cadena de ARNm con los genes *lacY* y *lacA*. El gráfico fue adaptado de [61].

La transcripción como fue descrita es válida para los organismos procariotas. En los organismos eucariotas el mecanismo de transcripción es más complejo. En procariotas no hay separación física entre transcripción y traducción, mientras que en los eucariotas la transcripción tiene lugar en el núcleo donde está el ADN y la traducción en el citoplasma donde están los ribosomas. Por otro lado, una vez realizada la transcripción en eucariotas, los intrones se extraen del ARNm. De esta forma los intrones son parte del gen, pero no se utilizan en la síntesis de proteínas [71]. En general el ARNm de eucariotas es monocistrónico. En procariotas hay un sólo tipo de moléculas de ARN polimerasa, mientras que en eucariotas hay tres tipos de ARN polimerasa (polimerasa I, II y III) utilizados para la síntesis de distintos tipos de ARN.

1.2.1. La ARN polimerasa bacteriana

La ARN polimerasa cataliza y controla el proceso de transcripción. El núcleo de la ARN polimerasa está formado por cuatro subunidades ($\alpha_2\beta\beta'$). En las células bacterianas el núcleo está generalmente asociado a una subunidad llamada *subunidad* o *factor sigma* (σ). El complejo formado por la enzima y la subunidad σ , es denominado *holoenzima* (ver Figura 1.11). La subunidad σ interviene específicamente en el reconocimiento del promotor, asegurando que la ARN polimerasa bacteriana se una al ADN en forma estable solamente en los promotores, de esta forma impone un nivel de especificidad.

Algunas bacterias producen varios tipos de subunidades σ distintas, que son utilizadas para responder a cambios generados en el medio ambiente. Por ejemplo, existen subunidades cuya cantidad se incrementa ante un aumento de temperatura, para disminuir la síntesis de proteínas que se estaba produciendo y realizar la síntesis de una nueva serie de proteínas [41]. Estas subunidades son denominadas *subunidades de choque térmico*. Los promotores reconocidos por distintos tipos de subunidades σ difieren unos de otros. Esto significa que una enzima que contiene un factor σ determinado, sólo es específica para un tipo de promotor. En [64] se explican los roles biológicos de distintas subunidades σ .

1.2.2. El promotor

Como se mencionó anteriormente, la ARN polimerasa no inicia la síntesis de ARN en cualquier lugar de la célula, sino en lugares específicos denominados promotores.

Con el objetivo de identificar cuáles son las características con las que tiene que contar el ADN para la unión con la ARN polimerasa, se compararon secuencias de distintos promotores. Se busca saber si hay secuencias que son comunes a todos los

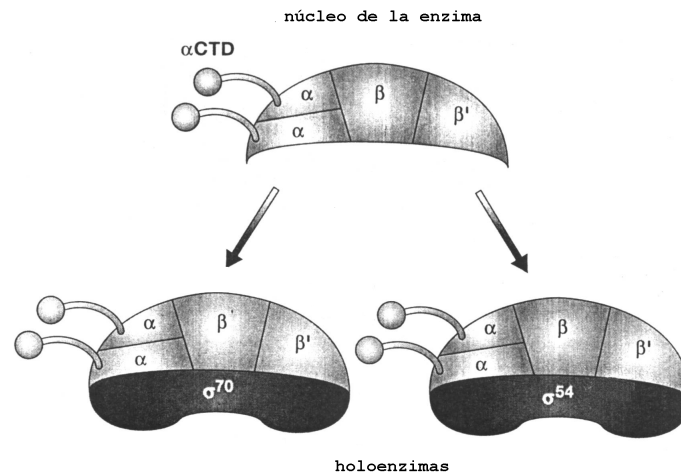


Figura 1.11: La ARN polimerasa de *E. coli*. Es la responsable del inicio de la transcripción. Los números asociados al nombre de un factor indican su masa, por ejemplo σ^{70} tiene 70 kD. La enzima se une con una subunidad σ . En este caso se muestran holoenzimas con subunidades σ^{70} y σ^{54} . Gráfico tomado de [61].

promotores (a éstas se las denomina *secuencias conservadas*). La conservación de estas secuencias se puede modelar de muchas maneras (mencionaremos algunas en la Sección 1.3). Una de éstas es mediante la *secuencia consenso*, que muestra la base con mayor aparición en cada posición de una secuencia conservada. En los promotores de *E. coli* no existe una conservación a lo largo de toda la longitud de la secuencia, sin embargo existen algunos fragmentos cortos en el promotor que sí están conservados. El que sólo se conserven secuencias muy cortas es una característica típica de los sitios reguladores, como los promotores, tanto en los genomas procariontes como en los eucariotes [41].

Los promotores más representativos de *E. coli* (i.e. aquellos reconocidos por la subunidad σ^{70} de la polimerasa) contienen hasta cuatro secuencias conservadas: una cadena de 6 nucleótidos centrada a aproximadamente 10 pb aguas arriba⁵ del TSS con consenso TATAAT. Por la posición en que se encuentra, a esta secuencia se la suele denominar **región -10**. Un hexámero centrado a aproximadamente 35 pb aguas arriba del TSS, con consenso TTGACA, denominado **región -35**. A la región de los promotores de *E. coli* compuesta por estas dos regiones se la denomina *core promoter*⁶ [50]. Existe también una señal mucho más débil en el TSS denominada

⁵Los términos *aguas arriba* y *aguas abajo* son utilizados para indicar las posiciones del ADN en referencia a la orientación de la cadena codificante, de manera tal que el promotor está *aguas arriba* de su gen.

⁶Los *core promoters* de *E. coli* rodean la región del inicio de la transcripción. Se extienden desde

CAP, que normalmente está compuesta por una pirimidina (*C* o *T*) seguida por una purina (*A* o *G*). Generalmente el TSS es la segunda base de la secuencia *CA*. El tamaño pequeño, la debilidad del consenso de la señal CAP y su distancia variable con respecto a la región -10 hacen que no se la considere como una señal obligatoria. Algunos promotores fuertes contienen un elemento adicional aguas arriba de la región -35 , llamado **elemento UP**. Este está localizado entre -60 y -40 pb aguas arriba del TSS y es rico en As y en Ts [65, 18, 61, 3]. En la Figura 1.12 se ven las regiones conservadas sobre un promotor particular.

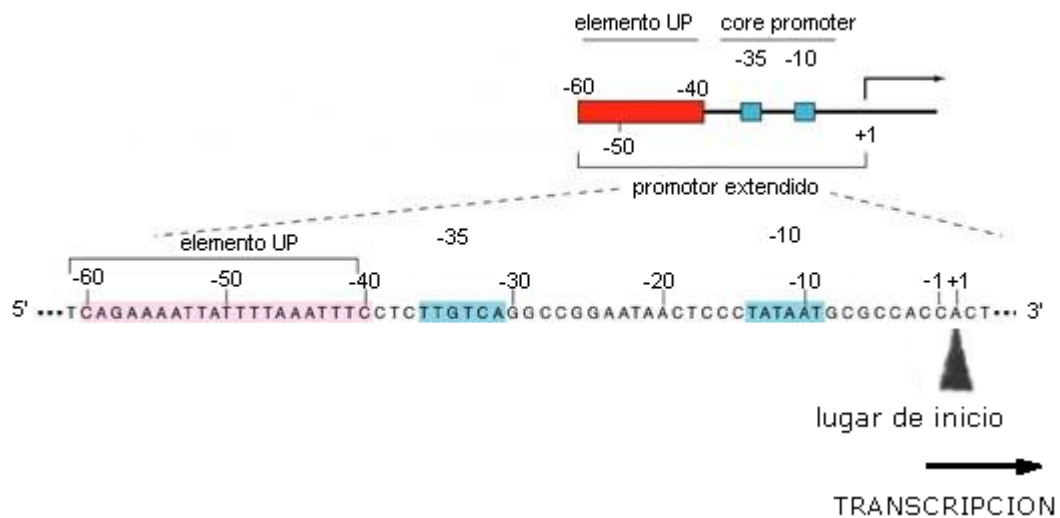


Figura 1.12: Esquema del promotor del gen *rrnB*, que codifica ARN ribosomal. La región *core* contiene las regiones -10 y -35 . Este promotor cuenta también con la región UP, que lo hace más fuerte. La polimerasa inicia la transcripción en el lugar de inicio (TSS) marcado como +1. Nótese que en la numeración no se utiliza el cero, el primer nucleótido del transcripto de ARN es el +1, y el inmediato aguas arriba es el -1 .

Las regiones -10 y -35 están separadas por distancias de entre 15 y 21 pb, pero en la mayor cantidad de los casos la distancia está entre 16 y 18 pb [41, 26, 25, 42]. Un espaciado entre regiones que esté afuera del rango 16-18 pb requiere conformaciones inusuales de ADN o polimerasa [25, 41]. La distancia entre la región -10 y el TSS (también denominado +1) está entre 4 y 12 pb, pero en la mayor parte de los casos el TSS está a 7 ± 2 pb de la región -10 [25]. En la Sección 1.4 daremos más detalle acerca de los consensos de las regiones -10 y -35 .

aproximadamente el nucleótido -40 hasta el +5 y contienen las regiones -10 y -35 . Constituyen un subconjunto de la región del promotor, que se expande entre -300 pb aguas arriba y 50 pb aguas abajo del TSS.

Se ha estudiado la interacción de la ARN polimerasa con el promotor y se encontró evidencia física de la importancia de la existencia de las secuencias conservadas en el promotor. Los hexámeros correspondientes a las regiones -10 y -35 interactúan con el factor σ^{70} de la ARN polimerasa. El dominio carboxilo-terminal de la subunidad α de la ARN polimerasa (αCTD) se pega al elemento UP, en los casos en que este existe, y estimula la transcripción [18] (ver Figura 1.13). La secuencia consenso del sitio -10 consiste exclusivamente en bases AT. Dado que el par AT está unido por dos enlaces puente hidrógeno y el par CG está unido por tres enlaces puente hidrógeno (ver Sección 1.1), la separación de las cadenas de ADN en un sitio compuesto exclusivamente por pares de bases AT demanda menos energía que la necesaria para la separación de bases GC. La región -10 sería el lugar en que se abre la doble hélice de ADN [41].

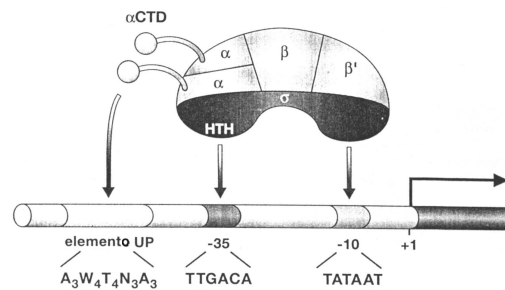


Figura 1.13: Elementos en promotores reconocidos por la holoenzima que contiene σ^{70} . La subunidad σ es una proteína elongada que puede contactar simultáneamente las regiones -10 y -35 del promotor. En los casos en que el elemento UP está presente es reconocido por las extensiones terminales carboxilo de las subunidades α . Abajo de cada elemento se muestra una secuencia consenso para ese elemento. W representa adenina o timina y N representa cualquier base (ver Tabla B.1). El subíndice indica la cantidad de reiteraciones de cada base. Gráfico adaptado de [61].

Como mencionamos anteriormente, los promotores reconocidos por distintos factores σ tienen otros consensos. En la Figura 1.14 se detallan los nucleótidos con mayor cantidad de apariciones en las regiones conservadas de los promotores reconocidos por algunos factores σ .

Mediante análisis estadísticos también se han encontrado secuencias conservadas en los promotores eucariotas. La cantidad de secuencias conservadas es mayor, y la distancia entre las mismas tiene mayor variabilidad. Los consensos conocidos son: el *TATA Box* y el *Inr* o *Initiator*, que contiene generalmente a la señal CAP que indica el TSS. La distancia entre el TATA Box y el Inr varía más que la distancia entre las regiones -35 y -10 en procariontas, pero en la mayor parte de los casos el TATA



Figura 1.14: Consensos de promotores reconocidos por distintos factores σ . El símbolo N representa cualquier base. Gráfico adaptado de [77].

Box se encuentra entre 25 y 30 pb aguas arriba del TSS. También se encuentran otras secuencias conservadas denominadas *CCAAT Box* y *GC Box*.

Las **mutaciones** en los promotores afectan los niveles de expresión de los genes que controlan, sin alterar los productos genéticos. Hay dos tipos de mutaciones: 1) mutaciones de disminución (*down mutations*), en que se pierde o se reduce la transcripción de los genes adyacentes y 2) mutaciones de incremento (*up mutations*), con las que se produce un incremento en la transcripción desde el promotor [41].

Fuerza de los promotores

Existen moléculas de ARNm que tienen que ser sintetizadas más frecuentemente que otras. La variación de las secuencias promotoras permite controlar el nivel de síntesis de ARNm. Cuanto más parecido sea el promotor a la secuencia consenso, mejor reconocimiento tendrá la ARN polimerasa por éste y más frecuentemente lo sintetizará.

En las bacterias se denomina **promotores fuertes** a aquellos que tienen secuencias que difieren poco de la secuencia consenso en las regiones conservadas y en que la distancia entre las mismas es cercana al ideal de 17 pb. Los **promotores débiles** tienen más diferencias. Por ejemplo, al promotor *recA*, cuya secuencia se muestra en la Figura 1.15, se lo considera fuerte. Difiere del promotor consenso de *E. coli* en sólo un nucleótido y en una base en el espaciado entre las dos regiones. Al promotor *araBAD* se lo considera débil, dado que difiere del consenso en 5 nucleótidos y en una base en el espacio entre las dos regiones [50]. La fuerza del promotor influye en la tasa de iniciación de la transcripción y por lo tanto es un factor de regulación de la transcripción.

Algunos de los promotores más fuertes conocidos en las bacterias son los de bacteriófagos [50, 24]. De esta forma pueden tener una replicación más rápida.

| región -35 | espacio | región -10 | secuencia |
|------------|---------|------------|-----------|
| TTGATA | - 16 - | TATAAT | recA |
| CTGACG | - 18 - | TACTGT | araBA |
| TTGACA | - 17 - | TATAAT | consenso |

Figura 1.15: Promotor fuerte recA y promotor débil araBAD. Se pueden observar las diferencias y similitudes con la secuencia consenso.

Regulación de la transcripción

Algunos promotores bacterianos son poco funcionales por sí mismos, ya sea porque son poco reconocidos por la ARN polimerasa o porque ésta tiene dificultades en abrir la hélice de ADN en el inicio de la transcripción. Estos promotores pueden recuperar la función normal mediante proteínas reguladoras que se unen en un lugar próximo del ADN, contactando con la ARN polimerasa de forma tal de incrementar la probabilidad de que se inicie la transcripción. Este modo de regulación se llama control positivo.

Las proteínas responsables de la regulación de la expresión de un gen en una célula determinada se denominan **factores de transcripción** (FT) y pueden actuar positiva o negativamente (aumentando o disminuyendo la expresión genética). Los FT se pegan al ADN en sitios de unión (*binding sites*) pertenecientes a la región regulatoria del gen y afectan el inicio de la transcripción.

1.2.3. Observaciones

A partir de lo expuesto en las subsecciones precedentes se puede notar que el proceso de transcripción, y en especial el proceso de iniciación de la transcripción, es muy complejo y está influenciado por muchos factores, cuyas funciones exactas aún no son totalmente conocidas.

El desarrollo de métodos computacionales para el reconocimiento de promotores, facilita las tareas del investigador del área de la biología. Con el uso de métodos automáticos de clasificación los experimentos biológicos son necesarios, pero pueden acotarse a los resultados positivos de los métodos predictivos.

La variabilidad de la distancia entre las dos secuencias conservadas y el que éstas sean difusas (*fuzzy*), no hacen posible la existencia de un modelo preciso de promotores de procariontes, lo que dificulta el hallazgo de una solución computacional.

1.3. Introducción a conceptos de bioinformática

Introduciremos algunos conceptos de bioinformática que utilizaremos a lo largo del trabajo.

1.3.1. Alineamiento de secuencias

El alineamiento de un conjunto de secuencias de ADN, ARN o proteínas suele ser utilizado para determinar relaciones evolutivas o funcionales. En nuestro caso estamos interesados en la identificación de relaciones funcionales. Normalmente las secuencias relacionadas funcionalmente comparten algún patrón, que se puede descubrir al alinearlas⁷ [30].

Se llega a un alineamiento de dos secuencias, disponiendo una secuencia sobre la otra y creando una correspondencia entre los caracteres⁸ de ambas. Un ejemplo de alineamiento puede verse en la Figura 1.16.

```
TCCATATTATAGGAT
ACCATATATTTGGAT
```

Figura 1.16: Alineamiento de dos secuencias.

Los alineamientos múltiples consisten en el alineamiento simultáneo de muchas secuencias. Se puede leer más acerca de alineamientos en [71]. Es necesario disponer de un mecanismo para medir cuán relacionadas están las secuencias alineadas para de esta forma poder determinar la calidad del alineamiento.

1.3.2. Modelos de alineamiento

Existen distintos métodos para modelar un alineamiento. El objetivo de los modelos es describir alineamientos de manera concisa.

Secuencias consenso

Las secuencias consenso se obtienen eligiendo la base con mayor aparición en cada posición de un alineamiento de secuencias y forman una especie de resumen del alineamiento. Es posible que ninguna de las secuencias que originó la secuencia consenso

⁷A partir de ahora nos referiremos a alineamientos de cadenas de ADN, que son las que nos interesan en este trabajo. Los alineamientos son iguales en el caso de ARN y proteínas, lo que difiere son las unidades que se alinean (nucleótidos en los dos primeros casos y aminoácidos en el tercer caso).

⁸Al referirnos a los alineamientos mencionaremos *nucleótidos*, *caracteres* o *letras* indistintamente.

tenga exactamente las mismas bases que ésta. La Figura 1.17 presenta las secuencias consenso definidas para las regiones -10 y -35 de las 12 primeras secuencias del alineamiento de promotores de *E. coli* realizado por Harley y Reynolds [25].

| | región -35 | | | región -10 | | | |
|-------------------|-----------------------------|--------|-----------|-------------------|---------|--------------------|----|
| 1 aceEF | ACGTAGACCTGT | CTTATT | GAGCTTTC | CGGCGAGAG | TTC AAT | GGGACAGGTCCAG | 17 |
| 2 ada | AAGATTGTTGGTTT | TTGCGT | GATGGTGA | CCGGGCAGC | CTAAAG | GCTATCCTTAACC | 17 |
| 3 alaS | AACGCATACGGTAT | TTTACC | TTCCCAGTC | AAGAAAAC | TATCTT | ATTCCCACCTTTTCAGT | 18 |
| 4 ampC | TGCTATCCTGACAG | TTGTCA | CGCTGATT | GGTGTCTG | TACAAT | CTAACGCATCGCCAATG | 16 |
| 5 ampC/cl6 | GCTATC | TTGACA | GTTGTCC | GCTGATTGG | TATCGT | TACAATCTAACGTATCG | 17 |
| 6 araBAD | TTAGCGGATCCTAC | CTGACG | CTTTTTAT | CGCAACTC | TCTACT | GTTTCTCCATACCCGTT | 16 |
| 7 araC | GCAAATAATCAATG | TGGACT | TTTCTGCC | GTGATTATA | GACACT | TTTGTACGCGTTTTTG | 17 |
| 8 araE | CTGTTTCCGAC | CTGACA | CCTGCGTGA | GTTGTTCAG | TATTTT | TTCACTATGTCTTACTC | 19 |
| 9 araI (c) | AGCGGATCCTAC | CTGGCG | CTTTTTAT | CGCAACTC | TCTACT | GTTTCTCCATACCCGTT | 16 |
| 10 araI (c) X (c) | AGCGGATCCTAC | CTGGCG | CTTTTTATC | GCAACTCTC | TACTAT | TTTCTCCATACCCGTTTT | 18 |
| 11 argCBH | TTTGTTTTTTCATTG | TTGACA | CACCTCTGG | TCATGATAG | TATCAA | TATTCATGCAGTATT | 18 |
| 12 argCBH-P1/6- | TTTGTTTTTTCATTG | TTGACA | CACCTCT | GGTCATAA | TATTAT | CAATATTCATGCAGTAT | 15 |
| | secuencias consenso: | TTGACA | | | TATAAT | | |

Figura 1.17: Secuencias consenso TATAAT y TTGACA definidas para las regiones -10 y -35 de 12 secuencias del alineamiento de promotores de *E. coli* realizado por Harley y Reynolds. En cada secuencia se resaltan los nucleótidos que coinciden con los consensos.

El concepto de consenso es útil y muy empleado, a pesar de que pierde la información relativa a la cantidad de apariciones de todos aquellos nucleótidos, que no son los más conservados para una posición determinada. Existen métodos alternativos para la representación de alineamientos de secuencias que no conllevan esa pérdida de información.

Matrices de alineamiento y de pesos

Los alineamientos de secuencias pueden ser modelados mediante matrices. Un modelo sencillo de matriz es la *matriz de alineamiento* [30]. Las entradas de una matriz de alineamiento están formadas por la cantidad de oportunidades en que un nucleótido dado es observado en la posición indicada de un alineamiento de secuencias. Otro tipo de matriz es la *matriz de pesos*, cuyos elementos pueden ser derivados de una matriz de alineamiento o determinados experimentalmente [30, 73, 28, 79]. Los elementos de las matrices de pesos determinan los pesos utilizados para determinar cuán parecida es una secuencia dada a los patrones descritos por la matriz. En [76] se explica el uso y construcción de distintas matrices de pesos. En particular, un elemento de una matriz de pesos puede ser derivado de una matriz de alineamiento, mediante la siguiente fórmula [30]:

$$w_{i,j} = \ln \frac{(n_{i,j} + p_i)/(N + 1)}{p_i}, \quad (1.1)$$

donde $n_{i,j}$ es la cantidad de oportunidades en que se observa la letra i en la posición j del alineamiento, N es la cantidad de secuencias consideradas en el alineamiento y p_i es la probabilidad *a priori* del nucleótido i . Se puede observar que $w_{ij} \approx \ln(\frac{f_{i,j}}{p_i})$, donde $f_{i,j} = n_{i,j}/N$, es la frecuencia de aparición del nucleótido i en la posición j .

La probabilidad *a priori* de un nucleótido es la probabilidad que tiene éste de aparecer en una secuencia. Para determinarla se puede considerar la frecuencia de aparición del nucleótido en el genoma con el que se trabaja o la frecuencia observada en el conjunto de secuencias a alinear. Como puede observarse en la Fórmula 1.1, el valor de un mismo elemento w_{ij} de la matriz de pesos difiere si se consideran distintas probabilidades *a priori*.

En la Figura 1.18 puede verse un ejemplo de una matriz de alineamiento y su correspondiente matriz de pesos.

La matriz de pesos mostrada no considera *gaps* (secuencias con inserciones o deleciones de nucleótidos) ni correlaciones entre distintas posiciones. Existen otras matrices que describen alineamientos que consideran estas características⁹ [29].

El uso de matrices que resumen el contenido del alineamiento evita el problema mencionado de las secuencias consenso, dado que resulta posible almacenar información relativa a la aparición de cada uno de los nucleótidos en cada posición de la matriz.

⁹No describiremos estas matrices, dado que los métodos que utilizaremos están basados en las matrices simples anteriormente descritas.

Alineamiento de secuencias

T A C A A T
 C T A A C G
 T A T C G T
 T A T A T T

 sec. consenso T A T A N T



Matriz de alineamiento

| | | | | | | |
|---|---|---|---|---|---|---|
| A | 0 | 3 | 1 | 3 | 1 | 0 |
| C | 1 | 0 | 1 | 1 | 1 | 0 |
| G | 0 | 0 | 0 | 0 | 1 | 1 |
| T | 3 | 1 | 2 | 0 | 1 | 3 |

$$\ln \frac{(n_{i,j} + p_i)/(N+1)}{p_i}$$



Matriz de pesos

| | | | | | | | |
|---|-------|-------|-------|-------|-------|------|-------|
| A | -1,6 | 0,96 | 0 | 0 | 0,96 | 0 | -1,61 |
| C | 0 | -1,61 | 0 | 0 | 0 | 0 | -1,61 |
| G | -1,61 | -1,61 | -1,61 | -1,61 | -1,61 | 0 | 0 |
| T | 0,96 | 0 | 0,59 | -1,61 | 0 | 0,96 | 0 |

secuencia de prueba T A T A T

Figura 1.18: Matrices de alineamiento y de pesos. El símbolo N representa cualquier base (ver Tabla B.1). Se muestra el alineamiento de cuatro secuencias de longitud 6 y la secuencia consenso del alineamiento. La posición ij de la matriz de alineamiento contiene la cantidad de apariciones del nucleótido i en la posición j del alineamiento. Se muestra la fórmula con la que se pasa de la matriz de alineamiento a la matriz de pesos. En la fórmula, N es la cantidad total de secuencias alineadas (en este caso 4), p_i es la probabilidad *a priori* del nucleótido i (en este ejemplo consideramos 0.25) y $f_{ij} = n_{ij}/N$ es la frecuencia de la letra i en la posición j . Los elementos recuadrados en la matriz de pesos corresponden a los pesos asignados a cada posición de la secuencia de prueba. El puntaje total de la secuencia de prueba, que proviene de sumar cada uno de los elementos recuadrados es 4.43.

Medidas de calidad de un alineamiento

Existen varias métricas para determinar qué es un buen alineamiento de secuencias [76, 30]. Nos centramos en la que utilizaremos a lo largo del trabajo, en la que se asume que la aparición de nucleótidos en una secuencia es independiente e idénticamente distribuida.

Se puede decir que los alineamientos más interesantes son aquellos en los que las frecuencias de los nucleótidos que aparecen difieren mucho de las probabilidades *a priori* de los nucleótidos. Una estadística estándar para medir esto es la medida *log-likelihood* [30]. Una variante de esta estadística es el *information content*, que proviene de dividir el *log-likelihood* por $-N$ [30, 29], y cuya fórmula es:

$$I_{seq} = \sum_{j=1}^L \sum_{i=1}^A f_{i,j} \cdot \ln \frac{f_{i,j}}{p_i}, \quad (1.2)$$

donde A es el tamaño del alfabeto (4 en este caso) y L la cantidad de columnas de la matriz. Esta fórmula es también denominada *fórmula Kullback-Leibler* o de *entropía relativa* en teoría de la información [38].

A partir de la probabilidad de obtener un *information content* mayor o igual al observado –dadas la cantidad de secuencias del alineamiento y el ancho del mismo¹⁰– y de la cantidad de alineamientos posibles, es posible calcular la frecuencia esperada de un *information content*, lo que da el significado estadístico del alineamiento [30]. La frecuencia esperada puede utilizarse para comparar alineamientos con distintos anchos y con distinta cantidad de secuencias [30].

1.4. Compilaciones de datos de promotores

Existen muchos estudios que intentan determinar exactamente las secuencias de nucleótidos que componen un promotor, con el objetivo de conocer qué secuencias son reconocidas y a qué secuencias se pega la polimerasa. Se pretende definir la estructura de los promotores a partir de distintos alineamientos.

A diferencia de lo que sucede con los promotores eucariotas de ARN polimerasa II, para los cuales existe una base de datos con información de promotores, *Eukaryotic Promoter Database* (EPD)¹¹, no hay un repositorio común para los datos de promotores de *E. coli*.

¹⁰Denominamos *ancho de un alineamiento* a la cantidad de nucleótidos considerados de cada secuencia a alinear.

¹¹Se puede acceder a la Eukaryotic Promoter Database en la dirección: <http://www.epd.isb-sib.ch/>.

Secuencias conservadas

En 1975 Pribnow [60] examinó y comparó las secuencias de cinco fragmentos de ADN a los que se pegaba la ARN polimerasa. Cuatro de éstos correspondían a promotores de bacteriófagos y uno a *E. coli*. A partir de la ubicación de las secuencias de forma tal que los sitios de inicio de la transcripción estuviesen alineados, se identificó una secuencia conservada centrada a 10 pb aguas arriba del lugar de inicio de la transcripción, a la que originalmente se la llamó Pribnow Box y actualmente se la denomina región -10 , con secuencia consenso TATAAT.

Posteriormente se descubrió que había una segunda secuencia conservada centrada a 35 pb aguas arriba del TSS, a la que se la denominó -35 , con secuencia consenso TTGACA.

Estas son las regiones a las que nos referiremos desde ahora cuando mencionemos las secuencias conservadas de los promotores.

Compilaciones y análisis de secuencias de promotores de procariontas

A partir del descubrimiento de estas dos secuencias consenso, se hicieron varios estudios estadísticos que analizan la conservación de las mismas y la distancia que las separa. Las secuencias de promotores de *E. coli* están entre los datos de secuencias más estudiados para encontrar una función común [42].

Hawley y McClure [26] realizaron en 1983 la primer compilación extensa. Los autores ubicaron las secuencias de manera tal que se alineasen mejor con los dos consensos determinados previamente con separación de entre 15 y 21 nucleótidos. Se prefirieron los apareamientos con distancia 17 entre los consensos. Algunos de los nucleótidos de secuencias se identifican como más conservados.

A partir del análisis de 112 promotores (de bacterias *E. coli* K12, fagos, plásmidos y transposones) llegaron –entre otras– a las siguientes conclusiones: 1) todos los promotores reconocidos por la subunidad σ^{70} tienen al menos dos de las tres bases más conservadas en la región -10 (TA _ _ T), 2) todos los promotores tienen al menos uno de los nucleótidos de más conservación de la región -35 (TTG _ _).

En 1987 **Harley y Reynolds** [25] presentaron la compilación de 272 y análisis de 263 promotores con TSS conocido para genes de *E. coli*, sus plásmidos y fagos. Entre estas secuencias incluyeron también promotores mutados. La compilación incluye los promotores compilados por Hawley y McClure.

Este estudio, que respalda en gran parte los resultados a los que llegaron Hawley y McClure en su trabajo anterior, llegó a las siguientes conclusiones: todas las bases de los hexámeros -35 y -10 están altamente conservadas, el 92 % de los promotores tienen un espaciado entre las regiones consenso de 17 ± 1 pb y 75 % de los lugares de

inicio de la transcripción se encuentra a 7 ± 2 bases aguas abajo de la región -10 . Las secuencias consenso de los promotores con espaciado de 16, 17 y 18 pb entre las regiones no difieren.

Se critica que la compilación de Harley y Reynolds es sesgada, ya que incluye promotores de bacteriófagos que son promotores fuertes. Estos no son típicos representantes de promotores de *E. coli*, que presenta también promotores débiles.

Por esta razón, **Lisser y Margalit** [42] publicaron en 1993 una compilación con promotores exclusivamente de *E. coli* (300 promotores). El aporte de este trabajo consiste principalmente en haber hecho una revisión y corrección exhaustiva de los datos publicados, eliminándose la información duplicada y corrigiéndose los datos mal transcritos de la literatura. Para cada secuencia se buscó la publicación más reciente y relevante, verificándose que el TSS de los repositorios de datos haya sido validado experimentalmente, actualizando datos incorrectos y privilegiando la información que se encuentra en la literatura sobre la que está en los repositorios de datos.

Los resultados del estudio mostraron cambios en la conservación de algunas bases con respecto a los estudios anteriores (8 de las 12 bases están menos conservadas), pero en general los resultados fueron parecidos, como muestra la Tabla 1.2.

| | -35 | | | | | | -10 | | | | | |
|--------------------------|-----|----|----|----|----|----|-----|----|----|----|----|----|
| | T | T | G | A | C | A | T | A | T | A | A | T |
| Hawley y McClure | 82 | 84 | 79 | 64 | 54 | 45 | 79 | 95 | 44 | 59 | 51 | 96 |
| Harley y Reynolds | 78 | 82 | 68 | 58 | 52 | 54 | 82 | 89 | 52 | 59 | 49 | 89 |
| Lisser y Margalit | 69 | 79 | 61 | 56 | 54 | 54 | 77 | 76 | 60 | 61 | 56 | 82 |

Tabla 1.2: Grado de conservación de las regiones consenso según las compilaciones de Hawley y McClure, Harley y Reynolds, y Lisser y Margalit. Se muestra el porcentaje de aparición de cada nucleótido del consenso.

Todos los estudios mencionados asumen, en base a los estudios anteriores, la existencia de las dos regiones consenso, una distancia óptima de 17 pb entre éstas y una distancia de entre 7 y 9 pb entre el TSS y la región -10 . Estas suposiciones pueden haber sesgado los resultados de los análisis. Por otro lado, la diferencia de criterios para la generación del alineamiento y la inclusión o exclusión de fagos pueden ser las causas de que los resultados tuviesen algunas diferencias. Hawley y McClure consideran distancias entre 15 y 21, Lisser y Margalit sólo estudian entre 15 y 19. Todos coinciden en que la mejor distancia es de 17 pb.

Por otro lado, para estudios computacionales basados solamente en los datos de las secuencias es muy importante contar con datos correctos y no sesgados. Por esta razón es importante el trabajo de compilación de secuencias realizado por Lisser y

Margalit.

Las compilaciones mencionadas no estaban integradas en una base de datos. **RegulonDB** [68] es una base de datos relacional, que contiene información de mecanismos de regulación de la transcripción y organización de operones en el cromosoma K12 de *E. coli*. Entre otros, cuenta con una tabla con información de promotores de *E. coli*. La última versión disponible a la fecha de realización de este trabajo, versión 3.2 del año 2001, contiene información de 580 promotores de *E. coli* K12 con factor σ^{70} . En [35, 66, 67] se explica el diseño relacional original y las modificaciones realizadas. La base de datos incluye información tomada de la literatura e información predicha por distintos métodos computacionales, ambas con una distinción clara. Los datos de *E. coli* fueron poblados inicialmente con las secuencias de la compilación de Lisser y Margalit.

Capítulo 2

Métodos computacionales para el reconocimiento de promotores

Uno de los temas de investigación en biología molecular es el descubrimiento de la función de secuencias de ADN o aminoácidos a partir de su estructura. Para esto se pueden utilizar distintos métodos basados en modelos matemáticos, físicoquímicos e intrínsecamente computacionales. Con el objetivo de inferir información funcional a partir de las secuencias se puede pensar en dos tipos de análisis: 1) reconocimiento de patrones determinados con secuencia conocida, por ejemplo búsqueda de un gen o un promotor conocido en una secuencia de ADN, y 2) búsqueda de subsecuencias con algún significado. Por ejemplo se quiere saber si una subsecuencia transcribe o no transcribe, si es un intrón o exón, si es o no es un promotor. Este trabajo está relacionado con el segundo tipo de búsqueda.

El problema de descubrimiento de promotores en cadenas de ADN no es trivial. Si bien existen modelos de la estructura de los promotores, no resultan satisfactorios debido a que no se puede tratar con la incertidumbre y ambigüedad de este problema biológico. La variabilidad en la secuencia de nucleótidos de los hexámeros identificados, la distinta importancia de cada uno de ellos, el que no todos los promotores cuentan con ambos motivos y la distancia variable entre los mismos, hacen que sea complejo modelar este problema. Más aún si se tiene en cuenta que desde el punto de vista específicamente biológico, los patrones regulatorios no necesariamente se adecuan a los modelos construidos.

Desde una perspectiva computacional, el descubrimiento y búsqueda de promotores puede ser abordado mediante distintos métodos. Los mismos incluyen: métodos fijos, estadísticos y redes neuronales. Los métodos fijos buscan una secuencia específica (patrón), donde cada uno de los nucleótidos que forma parte del mismo tiene igual peso o significación. Los métodos estadísticos generan matrices de alineamiento y les asignan puntajes a las secuencias de prueba a partir de los valores contenidos en los

elementos de las mejores matrices obtenidas. Existen arquitecturas de redes neuronales artificiales que proponen un método automático para calcular la incertidumbre relacionada con el *matching* y con la distancia entre patrones, a partir de un conjunto de entrenamiento. Las redes neuronales artificiales son muy utilizadas para el análisis de secuencias y reconocimiento de promotores.

En las secciones 2.1 y 2.2 describimos los métodos fijos y estadísticos, que compararemos en el Capítulo 4 con nuestro método. En la Sección 2.3 hacemos una introducción a conceptos básicos de redes neuronales necesarios para la comprensión de la red neuronal que desarrollamos (explicada en el Capítulo 3). En la Sección 2.4 comparamos los distintos métodos y en la Sección 2.5 presentamos distintas medidas de calidad de las predicciones necesarias para comparar los resultados de la aplicación de distintos métodos, o de variaciones de un mismo método. Por último, hacemos en la Sección 2.6 una breve revisión de las soluciones existentes basadas en redes neuronales.

2.1. Métodos fijos

Estos métodos toman como entrada un *patrón de búsqueda*¹ y un conjunto de secuencias y devuelven las secuencias en las que se encuentra el patrón. Los denominamos métodos fijos, dado que a cada uno de los nucleótidos que forma parte del patrón buscado se le da igual peso o significación y, por otro lado, se permiten pocas sustituciones de nucleótidos con respecto al patrón buscado. Se permite la búsqueda de patrones que contengan espaciados de longitud variable, lo que constituye una característica interesante para el tratamiento del problema de reconocimiento de promotores.

Una desventaja de estos métodos reside en que su implementación depende del conocimiento que se tiene del problema, que en muchos casos no es completo. Los resultados de la aplicación de los métodos fijos surgen a partir del uso de la secuencia consenso como patrón de búsqueda. Recordemos que en dicha secuencia cada posición se arma a partir del nucleótido que aparece más frecuentemente en cada posición del alineamiento.

En la Figura 2.1 puede verse el pseudocódigo de los métodos fijos.

¹Denominamos *patrón de búsqueda* a una subsecuencia específica a buscar.

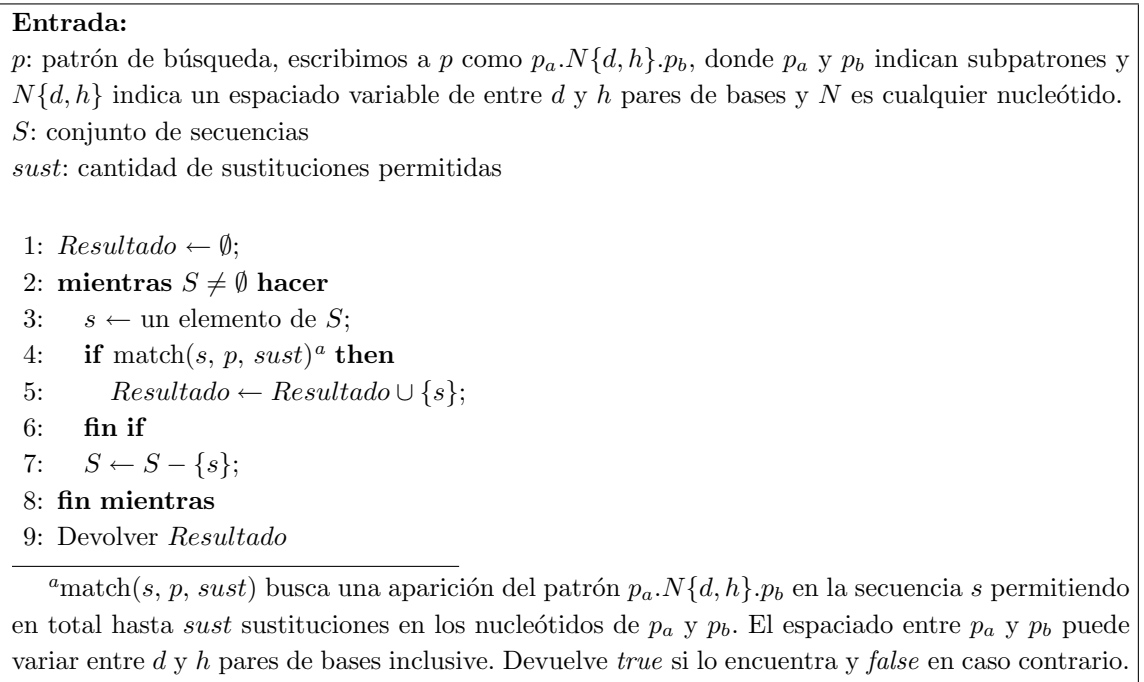


Figura 2.1: Pseudocódigo de los métodos fijos.

2.2. Métodos estadísticos

Existen métodos que calculan matrices de alineamiento a partir del alineamiento de un conjunto de secuencias de ADN [11, 73, 75, 74, 30]. Estos métodos consideran aspectos de teoría de la información y de estadística.

En particular, los análisis realizados por Hawley y McClure [26], Harley y Reynolds [25] y Lisser y Margalit [42], analizados en la Sección 1.4, muestran la importancia de los consensos TATAAT y TTGACA a partir de los resultados de los análisis estadísticos que realizaron a secuencias de *E. coli*. Bucher [11] presenta matrices de peso para definir los cuatro elementos importantes de los promotores de eucariotas y Penotti [58] realizó estudios estadísticos del TATA Box y TSS en ADN de humanos.

Consensus [29, 30] es un algoritmo utilizado para la determinación de alineamientos múltiples de secuencias y el descubrimiento de patrones que describen relaciones funcionales. Aplica los modelos y fórmulas descriptos en la Sección 1.3. El algoritmo *Patser* toma una matriz de alineamiento y un conjunto de secuencias y le asigna un puntaje a cada secuencia del conjunto, de acuerdo al peso correspondiente a cada uno de sus nucleótidos según la matriz de pesos del alineamiento. La combinación de ambos algoritmos puede ser utilizada para la búsqueda de promotores [29, 34]. A continuación describimos cada uno de ellos.

Consensus

En la Figura 2.2 se describe el algoritmo propuesto por Hertz y Stormo [30]. En la descripción mostrada asumimos que cada secuencia contribuye a lo sumo una vez al alineamiento y no se considera la secuencia complementaria.

Cada alineamiento almacenado por Consensus se resume en una matriz de alineamiento, cuyos elementos informan la cantidad de ocurrencias de los nucleótidos en cada posición. Si las secuencias a alinear son secuencias de promotores, se puede considerar que las matrices del alineamiento resultante almacenan una representación del promotor.

Como entrada se ingresan entre otros, un conjunto de secuencias a alinear y el ancho del alineamiento que se quiere obtener.

Como puede observarse en la figura, el algoritmo utiliza el *information content* como métrica para determinar qué matrices conservar en cada ciclo. A partir del cálculo de la probabilidad de obtención de un *information content* mayor o igual al observado, dadas la cantidad de secuencias del alineamiento y el ancho del alineamiento buscado, y de la distinta cantidad de alineamientos posibles calcula la frecuencia esperada del *information content*, lo que representa el significado estadístico del alineamiento [30]. La frecuencia esperada puede utilizarse para la comparación de alineamientos de distinto ancho y de distinta cantidad de secuencias. Cuánto menor es la medida, mejor es el alineamiento.

Los alineamientos almacenados que no consideran palabras de todas las secuencias a alinear son denominados *resultados intermedios*, mientras que aquellos que tienen una palabra de cada secuencia son denominados *resultados finales*.

Para evitar que la búsqueda sea un procedimiento muy largo, en lugar de utilizar un algoritmo exacto se opta por una solución de compromiso entre tiempo y precisión. A estos algoritmos se los conoce con el nombre de heurísticas. Este es un algoritmo goloso (*greedy*) debido a que en cada paso se intenta optimizar la solución del subproblema analizado (la alineación de las secuencias consideradas hasta el momento) y no se retrocede una vez tomada la decisión.

En la Figura 2.3 se ve ejemplifica la ejecución de Consensus.

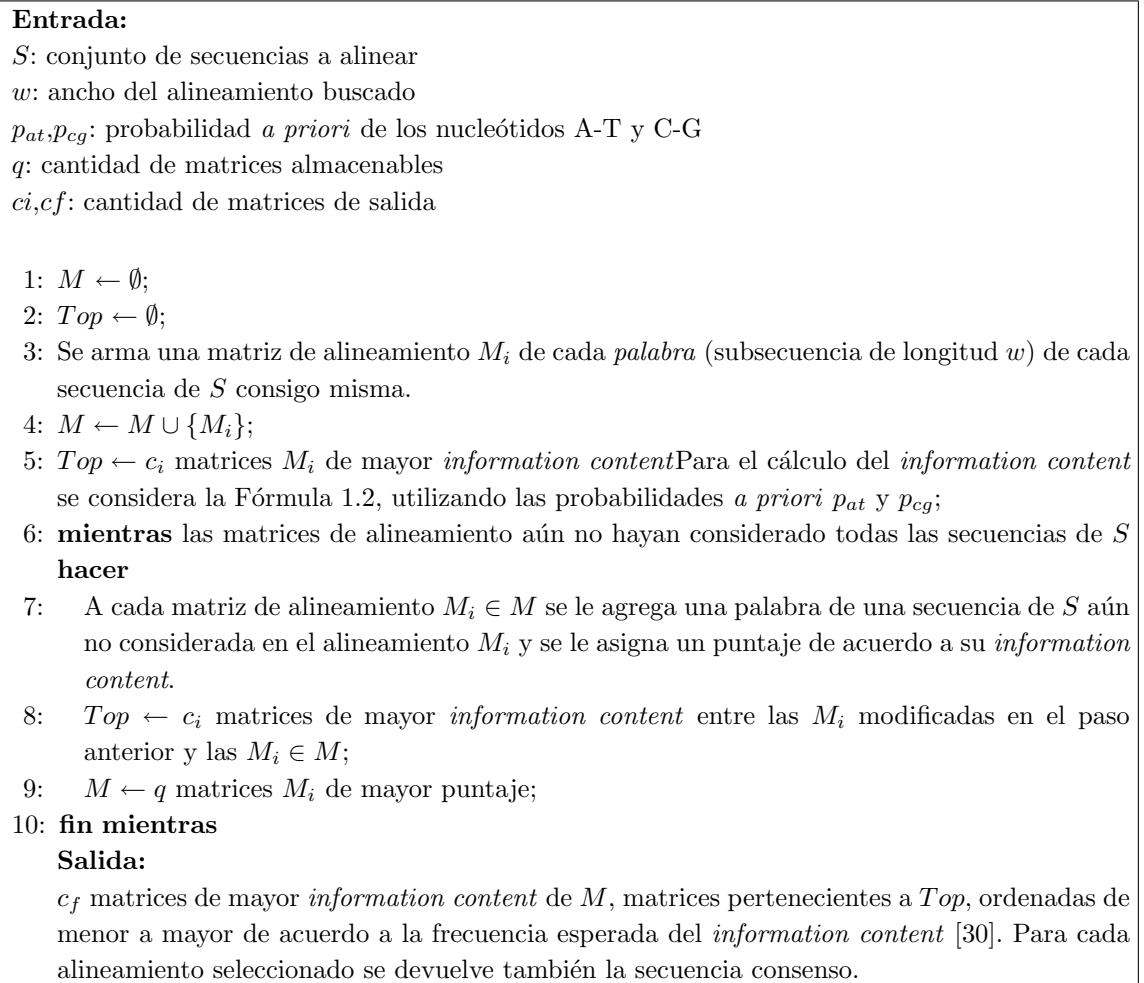


Figura 2.2: Algoritmo Consensus.

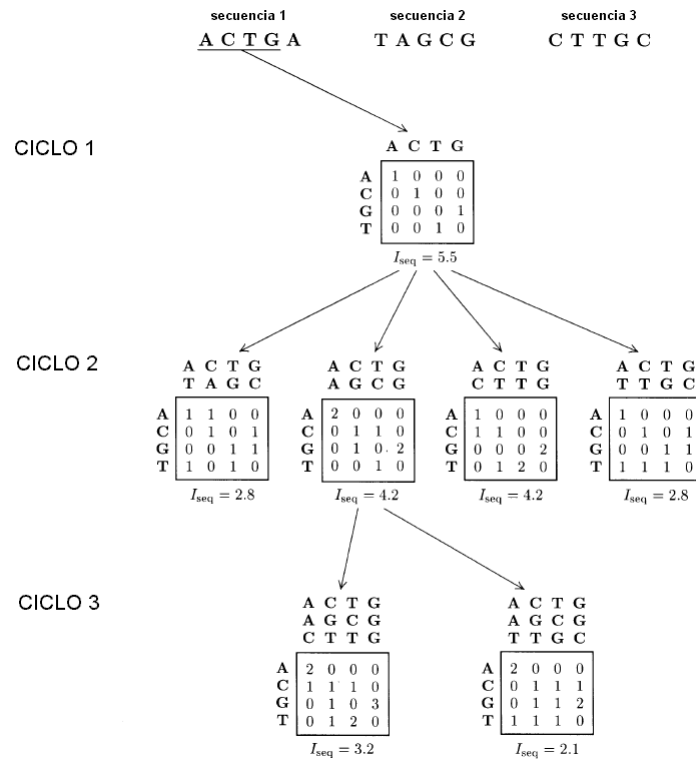


Figura 2.3: Ejemplo de ejecución del algoritmo Consensus para encontrar alineamientos de ancho cuatro de las tres secuencias de ADN de longitud cinco listadas arriba. Cada base tiene una probabilidad de aparición de 0.25. En el *ciclo 1* por una cuestión de simplicidad sólo se muestra la matriz de alineamiento originada a partir de la primer palabra de la *secuencia 1*. En la ejecución real, se crearía una matriz de alineamiento para cada una de las seis palabras de longitud cuatro existentes en las tres secuencias de longitud cinco. En el *ciclo 2* se agregan las cuatro matrices que se pueden crear agregando una secuencia adicional al alineamiento representado por la matriz resultante del *ciclo 1*, sin considerar la *secuencia 1*, dado que ya aportó una palabra. En el *ciclo 3* se muestran las dos matrices que pueden crearse agregando una tercer secuencia a la segunda matriz mostrada del segundo ciclo. I_{seq} es el *information content* del alineamiento de secuencias. Ejemplo adaptado de [30].

Patser

Este algoritmo, descrito en la Figura 2.4, toma una matriz de alineamiento o de pesos y a partir de éstos les da un puntaje a las palabras (subsecuencias de longitud fija) de las secuencias indicadas.

Entrada:

M : matriz de alineamiento con c columnas

p_{at}, p_{cg} : probabilidad *a priori* de los nucleótidos A-T y C-G

u : umbral (es vacío si se quieren obtener como resultado todos los puntajes calculados)

S : conjunto de secuencias a testear, de longitud l

$puntajes$: puntajes a devolver por cada secuencia (opciones: “el mejor” o “todos”). El “mejor” devuelve el mejor puntaje de todas las palabras y “todos” devuelve los puntajes de todas las palabras.

- 1: Convertir matriz de alineamiento M en matriz de pesos W mediante la Fórmula 1.1 y utilizando las probabilidades *a priori* p_{at} y p_{cg} .
- 2: desmarcar todos los elementos $s \in S$;
- 3: **mientras** existe $s \in S$ sin marcar **hacer**
- 4: $s \leftarrow$ un elemento no marcado de S ;
- 5: $com \leftarrow 1$;
- 6: **mientras** $com \leq l - c + 1$ **hacer** {Se calcula el puntaje para todas las subsecuencias de s de largo c (i.e. para todas las palabras de s)}
- 7: $puntaje(s_{com,com+c-1}) \leftarrow \sum_{j=1}^c v_{j+com-1}^T \cdot w_j$,
donde w_j es el vector de pesos de la columna j de la matriz de pesos W y

$$v_{j+com-1}^T = \begin{cases} (1, 0, 0, 0) & \text{si } s_{j+com-1} = \text{A} \\ (0, 1, 0, 0) & \text{si } s_{j+com-1} = \text{C} \\ (0, 0, 1, 0) & \text{si } s_{j+com-1} = \text{G} \\ (0, 0, 0, 1) & \text{si } s_{j+com-1} = \text{T} \end{cases} \quad (2.1)$$

- 8: $com \leftarrow com + 1$;
- 9: **fin mientras**
- 10: marcar s ;
- 11: **fin mientras**

Salida:

Para cada secuencia $s \in S$ devolver los puntajes acordes al parámetro $puntajes$ que superen el umbral u .

Figura 2.4: Algoritmo Patser.

Un ejemplo del resultado de este algoritmo para una matriz de 6 columnas y una secuencia de prueba de longitud 6 puede verse en la Figura 1.18.

2.3. Redes neuronales

Las redes neuronales artificiales tienen muchas ventajas sobre los métodos anteriormente descritos para la solución al problema expuesto: su *conocimiento* se basa en un *aprendizaje* realizado a partir de los datos que se toman como entrada y se realiza mediante la ejecución de algoritmos generalmente conocidos. La capacidad de *generalización* de la red permite encontrar soluciones para problemas para los cuales no hay un modelo matemático definido. Además, el costo computacional se concentra en el entrenamiento. Una vez realizado, se puede utilizar la red resultante en tiempo real. Finalmente, es sencillo reentrenar la red para que pueda mejorar las generalizaciones que hace a partir de la disponibilidad de nuevos datos.

Las redes neuronales artificiales (a partir de ahora las denominaremos redes neuronales (RN) o simplemente redes) constituyen un paradigma computacional alternativo al usual propuesto por von Neumann, que se basa en la ejecución de una secuencia de instrucciones. Están construidas a partir de un modelo muy simplificado del funcionamiento de las neuronas en el cerebro. En la Figura 2.5 puede verse el esquema de una neurona.

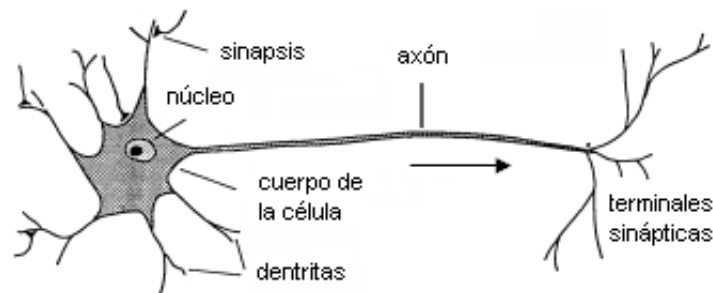


Figura 2.5: Esquema de una neurona, gráfico adaptado de [31].

Una neurona es una célula compuesta por un cuerpo o soma -en donde se encuentra el núcleo- y una fibra larga denominada axón. Conectados al cuerpo hay filamentos llamados dendritas, que -junto con el cuerpo- reciben las señales transmitidas por otras células. Generalmente el axón termina en una subdivisión en forma de árbol, en cuyos extremos se encuentran las terminales sinápticas, mediante las cuáles se realiza la transmisión de señales a otras neuronas. La transmisión de una señal de una célula a otra se da a partir de un proceso químico, en que el transmisor envía sustancias específicas desde las sinapsis a las célula receptoras. Esto hace variar el potencial eléctrico del cuerpo de la neurona receptora. Si el potencial llega a un

umbral determinado se envía un pulso mediante el axón, que a través de las terminales sinápticas pasa a otras células y se dice que la neurona *disparó*. En [27] se puede ver una descripción más completa del funcionamiento de las neuronas.

Modelo matemático de una neurona

Desde el punto de vista matemático una neurona es la unidad básica de procesamiento de información de una red neuronal. Un modelo de una neurona está dado por:

- un conjunto de conexiones denominadas sinapsis, que tienen pesos asociados. A cada entrada llegan señales, que se multiplican por el peso sináptico. Una señal ξ_j conectada a la neurona k se multiplica por el peso w_{kj} (en los subíndices denotamos primero la neurona y luego la señal).
- un sumador de las señales de entrada pesadas por las sinapsis de la neurona, que devuelve una combinación lineal de las entradas.
- una función de activación, que limita la amplitud de la salida de una neurona. La función de activación simboliza la activación de la neurona representada por la unidad.
- un umbral b_k , que sirve para variar la actividad de una unidad o neurona.

En la Figura 2.6 puede verse un esquema del modelo matemático de una neurona.

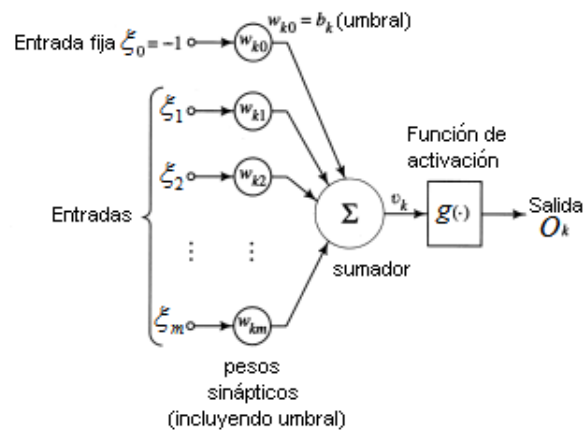


Figura 2.6: Esquema del modelo matemático de una neurona. Se toma al umbral como un peso más con entrada -1 .

El cómputo realizado por la neurona k puede describirse mediante la siguiente fórmula:

$$O_k = g\left(\sum_{i=1}^m w_{ki} \cdot \xi_i\right) - b_k, \quad (2.2)$$

donde g es la función de activación, ξ_i es uno de los componentes del patrón de entrada o la señal de salida de otra neurona ($\xi = \{\xi_1, \xi_2, \dots, \xi_m\}$), w_{ki} son los pesos sinápticos de la neurona k , b_k es el umbral y O_k es la señal de salida de la neurona.

Algunas posibles funciones de activación son la lineal, la escalón y la sigmoidea [27, 31]. Las funciones sigmoideas tienen forma de S, se saturan con valores grandes en los dos extremos y son estrictamente crecientes. Dos funciones sigmoideas comúnmente usadas son la función logística y la tangente hiperbólica. En la Ecuación 2.3 y la Figura 2.7 se muestran la fórmula y el gráfico de una función logística. Esta función tiene como imagen un rango continuo de valores entre 0 y 1 y es diferenciable.

$$g(v_k) = \frac{1}{1 + e^{-v_k}}, \quad (2.3)$$

donde $v_k = \sum_{i=0}^m w_{ki} \cdot \xi_i$.

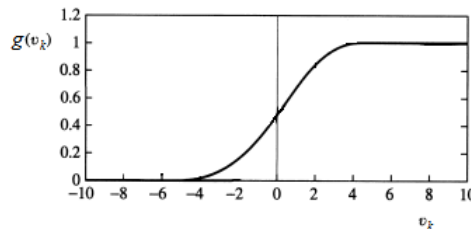


Figura 2.7: Función de activación logística.

Conexión entre neuronas

Las neuronas se conectan entre sí formando redes neuronales mediante distintas arquitecturas. En la arquitectura *feedforward* las neuronas están organizadas en capas (*layers*). En el modelo de *redes feedforward de una capa* (*single layer feedforward network*) existen solamente una capa de entrada y otra de salida, a estas redes se las denomina *perceptrones simples*. En el modelo de *redes multicapa feedforward* (*multi-layer feedforward networks*) existen también capas intermedias, denominadas capas ocultas (*hidden layers*), que, con la aplicación de funciones de activación no lineales,

permiten que la red aprenda tareas más complejas, extrayendo características a partir de los datos de entrada. Las conexiones se hacen desde neuronas de una capa a neuronas de capas posteriores² (no hay ciclos), por eso a este tipo de arquitecturas se las denomina *feedforward*, que significa que se alimentan hacia adelante. En la Figura 2.8 pueden verse un perceptrón simple y una red con una capa intermedia.

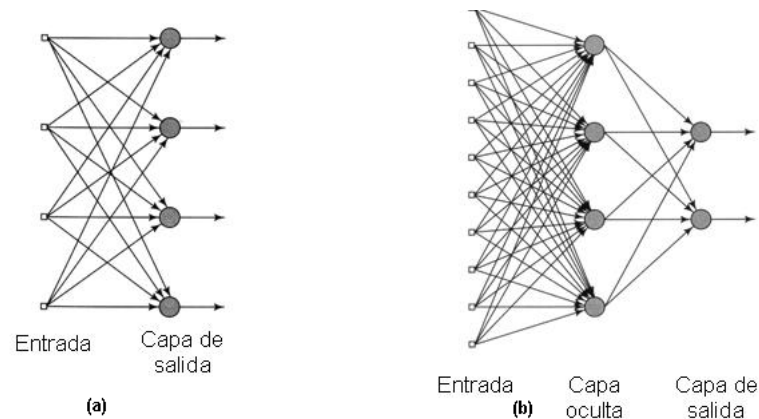


Figura 2.8: Arquitecturas de redes neuronales: (a) perceptrón simple, (b) red *feedforward* de dos capas. La entrada a la red se presenta en las neuronas de la capa de entrada y éstas alimentan a los nodos de la primer capa oculta (en caso de que exista). Cada uno de éstos realiza los cálculos y a su vez alimenta a otra capa oculta o a la capa de salida. En este gráfico ambas redes están totalmente conectadas.

Utilidad y funcionamiento de las redes neuronales

Las redes neuronales resuelven problemas de clasificación, descubrimiento de patrones y aproximación de funciones a partir de un conocimiento adquirido mediante un proceso de **aprendizaje** realizado en función de ejemplos que toman como entrada.

Se dice que una red está aprendiendo su tarea, si los resultados obtenidos iterativamente a partir de refinamientos sucesivos desde un punto inicial son cada vez mejores. El paradigma de aprendizaje supervisado se basa en la existencia de un maestro que tiene conocimiento del ambiente, representado mediante un conjunto de ejemplos entrada-salida. El conocimiento que tiene el maestro se transfiere a la RN mediante un algoritmo de aprendizaje que realiza ajustes iterativos en los pesos con el objetivo de minimizar la diferencia entre el resultado arrojado por la red y el resultado esperado (ejemplo de salida) para cada ejemplo de entrada. En la Figura 2.9 se puede ver un diagrama del aprendizaje supervisado.

²El orden en que mencionamos las capas de una red neuronal de N capas es: entrada, primer capa oculta, segunda capa oculta, ..., capa de salida (N-ésima capa).

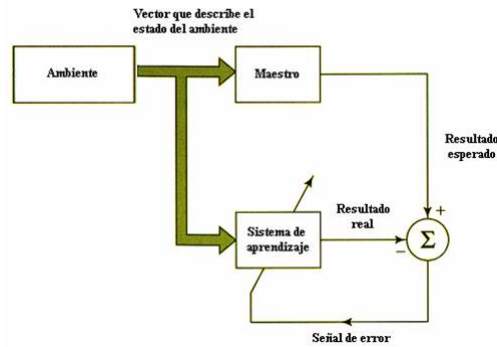


Figura 2.9: Diagrama explicativo del paradigma de aprendizaje supervisado. Gráfico adaptado de [27].

El conjunto de ejemplos entrada-salida utilizados por la red en el aprendizaje supervisado constituye lo que se llama el **conjunto de entrenamiento**. Notamos a estos pares entrada-salida como $\mu = (\xi, \zeta)$, a μ lo denominamos *patrón*. Para predecir la performance de un clasificador con nuevos datos necesitamos conocer su tasa de error en un conjunto independiente del utilizado para el entrenamiento. A este conjunto se lo denomina **conjunto de test**. La capacidad de la red de producir salidas razonables para entradas no pertenecientes al conjunto de entrenamiento se denomina **generalización** y la de recordar los ejemplos con los que fue entrenada se denomina **memorización**.

La habilidad de las redes neuronales para generalizar su aprendizaje depende de la selección adecuada de los conjuntos de entrenamiento y test [1, 84]. En un problema de clasificación de datos en dos clases también es importante la distribución de los datos que representan ejemplos de cada una de estas.

El error cuadrático para un patrón se calcula mediante la siguiente fórmula:

$$EC[w] = \frac{1}{2} \sum_{i=1}^M (\zeta_i - O_i)^2, \quad (2.4)$$

donde M es la cantidad de neuronas de la capa de salida de la red. Una medida de error (usualmente denominada *función de costo* o *función de energía*) muy utilizada para redes *feedforward* es el error cuadrático medio (ECM), que proviene de calcular el promedio del EC de todos los patrones de una época:

$$ECM[w] = \frac{1}{2N} \sum_{\mu=1}^N \sum_{i=1}^M (\zeta_i^\mu - O_i^\mu)^2, \quad (2.5)$$

donde N es la cantidad de patrones del conjunto de entrenamiento³. EC y ECM son función de los parámetros libres (pesos y umbrales) de la red. El objetivo del proceso de aprendizaje es ajustar estos parámetros para minimizar el ECM . El conocimiento adquirido por la red está definido por los valores que toman sus parámetros libres.

Un algoritmo de aprendizaje comúnmente utilizado para el entrenamiento de redes neuronales *feedforward* es el algoritmo de **back-propagation**. Este algoritmo propone un método para modificar los pesos con el objetivo de aprender un conjunto de entrenamiento de pares entrada-salida.

El aprendizaje puede ser **secuencial** (también llamado **on-line**), en que la actualización de los pesos se realiza después de la presentación de cada patrón del conjunto de entrenamiento, y **batch**, en que la actualización de los pesos se hace una vez presentados todos los patrones del conjunto de entrenamiento⁴. En el caso del aprendizaje secuencial, las actualizaciones se realizan a partir de los errores computados para cada patrón presentado a la red. El promedio de esas actualizaciones sobre todo el conjunto de entrenamiento es un estimado del cambio que resultaría si se modificasen los pesos a partir de la minimización del error cuadrático medio de todo el conjunto de entrenamiento. En [27] se explica la calidad del estimado.

Para realizar la actualización de los pesos de la red neuronal *feedforward* el algoritmo de *back-propagation* utiliza el método de descenso por gradiente. Esta es una técnica de optimización, basada en una heurística que dice que la manera de llegar a un mínimo es deslizándose pendiente abajo en la superficie que define la función de error. Para aplicar esta técnica se utiliza el gradiente de la función de error (ver Ecuación 2.4), para lo cual se requiere que la función sea continua y derivable⁵. El gradiente es un vector, que en cada coordenada tiene la derivada de la función de error en función del peso representado en esa coordenada. El gradiente de una función en un punto indica la dirección de máximo crecimiento de la función en ese punto. Por lo tanto, la dirección de máximo decrecimiento en ese punto está dada por la multiplicación del gradiente en ese punto aplicado a los pesos por -1 .

Para una unidad i de la capa de salida de la red es sencillo calcular el error ($\zeta_i^\mu - O_i$), dado que ζ_i se conoce (es la salida correspondiente al patrón μ del conjunto de entrenamiento). Sin embargo para una neurona j perteneciente a una capa intermedia ζ_j no es conocido, en este caso se realizan los cálculos a partir de una propagación *hacia atrás* del error. De esta forma tenemos distintas fórmulas para la actualización de los pesos de la red.

³En las Ecuaciones 2.4 y 2.5, $EC[w]$ y $ECM[w]$ valen cero si y sólo si $\forall i$ y $\forall \mu$ los pesos w satisfacen $\zeta_i^\mu = O_i^\mu$. Cualquier otra función $f(O_i^\mu, \zeta_i^\mu)$ diferenciable, que es minimizada cuando $O_i^\mu = \zeta_i^\mu$ puede reemplazar al término $(\zeta_i^\mu - O_i^\mu)^2$ en la *función de costo* [31].

⁴Se denomina *época* a la presentación de todos los patrones del conjunto de entrenamiento.

⁵Como se mencionó anteriormente, la función logística satisface estas condiciones.

La iteración n -ésima del algoritmo *on-line* consiste en los siguientes pasos:

Paso *forward* Se alimenta la capa de entrada de la red con un patrón perteneciente al conjunto de entrenamiento y cada unidad calcula su salida, a la cual le aplica la función de activación

Paso *backward* Se propaga el error hacia atrás actualizando los pesos y umbrales para reflejar el error de la predicción de la red. Se actualizan los pesos de acuerdo a la siguiente fórmula:

$$w_{ji}(n) \leftarrow w_{ji}(n) + \Delta w_{ji}(n), \text{ donde}$$

$$\Delta w_{ji}(n) = \eta \cdot \delta_j(n) \cdot O_i(n),$$

η es la velocidad de aprendizaje o *learning rate* y

$$\delta_j(n) = \begin{cases} e_j(n) \cdot g'_j(v_j(n)) & \text{si } j \text{ es una neurona de la capa de salida} \\ g'_j(v_j(n)) \cdot \sum_k \delta_k(n) \cdot w_{kj}(n) & \text{si } j \text{ es una neurona oculta} \end{cases}, \quad (2.6)$$

Las Ecuaciones 2.6 provienen de calcular el gradiente de la función de error (Fórmula 2.4). Un desarrollo claro de las mismas puede verse en el libro de Haykin [27].

El método del gradiente descendente, así como también otras técnicas de optimización pueden quedar fijos en **mínimos locales** de la función de costo en vez de llegar a mínimos globales (ver Figura 2.10).

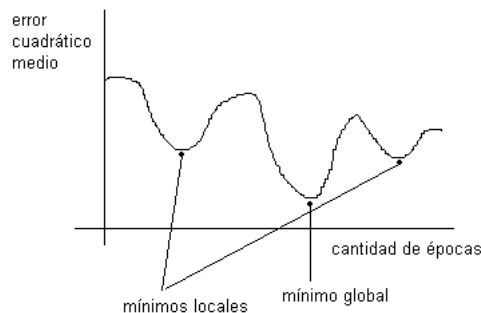


Figura 2.10: Ejemplo de presencia de mínimos locales y globales en una función de error.

La capacidad de generalización de una red está relacionada con el entrenamiento. En la Figura 2.11 se puede ver un ejemplo de una red que generaliza bien y de otra que no lo hace, esto último puede deberse al sobreentrenamiento (*overtraining* u

overfitting) [27, 84], que ocurre cuando el entrenamiento es demasiado largo. En estos casos, la red memoriza muy bien los elementos del conjunto de entrenamiento, pero la generalización es mala. La elección de un criterio de corte adecuado del algoritmo de aprendizaje puede evitar el *overtraining*. Una posibilidad para elegir el criterio de corte es detener el entrenamiento cuando el error cuadrático medio del conjunto de entrenamiento llega a un mínimo, pero esto puede derivar en un sobreentrenamiento. Una solución a este problema es utilizar, además de los conjuntos de entrenamiento y test, un tercer conjunto denominado *conjunto de validación* y detener el entrenamiento cuando el error cuadrático medio de éste llega a un mínimo [84, 6]. Esta constituye una mejor alternativa que el uso del conjunto de test, pero si se cuenta con pocos datos tiene como inconveniente que los utilizados para validación no se pueden utilizar para el entrenamiento ni el test [84]. Otros posibles criterios para detener el aprendizaje incluyen: hacerlo cuando todos los Δw_{ij} de la época anterior fueron menores a un valor determinado, cuando el porcentaje de elementos mal clasificados en la época anterior es menor que un valor determinado o cuando se ejecutó una cantidad determinada de épocas.

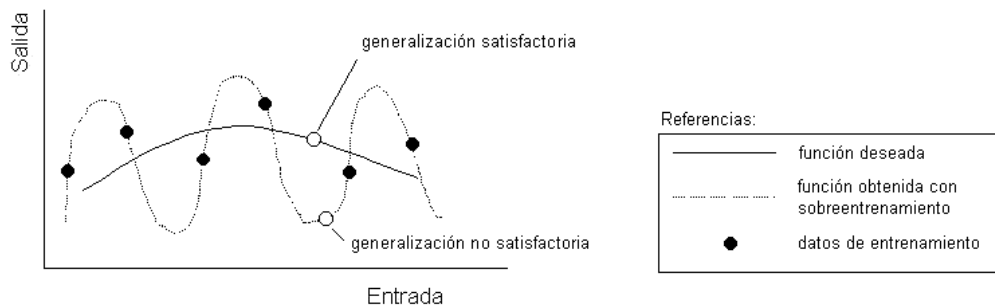


Figura 2.11: Memorización y generalización en una red. Se muestran ejemplos de dos entrenamientos distintos. En uno se aprende la función buscada (con memorización y generalización buenas) y en el otro hay sobreentrenamiento, lo que hace que los datos de entrenamiento sean aprendidos con exactitud y la generalización sea muy mala.

2.4. Comparación entre los distintos métodos

Una diferencia fundamental entre los diseños de una red neuronal y de un algoritmo de procesamiento de información basado en una secuencia programada de instrucciones es que para llegar al segundo es necesario contar con un modelo de las observaciones del ambiente, mientras que una red neuronal se diseña a partir de los datos sin la necesidad de proveer un modelo. Esto es muy útil en los casos en que

no es sencillo modelar un problema (por ej. porque no se lo conoce completamente o porque es muy complejo). Aun en los casos en que se tienen reglas claras, se puede ahorrar también en tiempo de diseño y programación. En este sentido los métodos estadísticos se parecen a las RN, en que no precisan un modelo del problema. En cambio, los métodos fijos se basan en el desarrollo de un algoritmo a partir del conocimiento del modelo. En este caso, se puede pensar que la falta de conocimiento de la estructura exacta de los promotores hace que los métodos fijos no sean adecuados para la resolución del problema de reconocimiento de promotores.

Consensus y las redes neuronales construyen modelos a partir de los datos. Los modelos difieren en que las redes neuronales son cajas negras y generalmente no es sencillo hacer deducciones a partir del conocimiento almacenado en los pesos de la red, mientras que Consensus hace una representación del conocimiento adquirido en una matriz, fácilmente comprensible. Sin embargo, las matrices de alineamiento tienen la desventaja de no poder modelar la distancia variable entre los dos subsecuencias conservadas [34].

En la siguiente sección vemos cómo comparar predicciones provenientes de distintos métodos o de distintas configuraciones de un mismo método.

2.5. Medidas de calidad de las predicciones

Para comparar la calidad de dos predicciones es necesaria la definición de una medida de calidad de las mismas. En la literatura y en los trabajos realizados en el tema se encontraron distintos métodos. A lo largo de esta sección analizaremos los más utilizados, mencionaremos cuál nos resulta más adecuado a nuestro problema y presentaremos una modificación a éste: el Coeficiente de correlación estandarizado o SCC.

Error cuadrático medio

Dado que el objetivo del entrenamiento de la red es obtener un mínimo de la función de energía cercano al mínimo global, posiblemente la métrica más intuitiva para medir la performance de una red depende de esta función. El error cuadrático medio (ver Ecuación 2.5) es una función de energía común para perceptrones en aprendizaje supervisado. Esta medida muestra un promedio del error obtenido como resultado de la evaluación que hace la red de cada uno de los patrones presentados, a partir de los resultados esperados conocidos.

Una posible métrica para determinar el grado de similitud entre dos redes es la comparación del error cuadrático medio arrojado para un mismo conjunto de patrones.

A pesar de ser un buen indicador y muy utilizado, por ejemplo, para detener el entrenamiento de la red, este indicador puede refinarse más con el objetivo de lograr una comparación más adecuada.

Al tener una relación positivos:negativos⁶ de, por ejemplo, 1:25 la comparación del resultado de dos redes o soluciones de acuerdo al error cuadrático medio posiblemente no es la mejor opción, ya que podría darse una situación en la que una red (red1) clasificó muy bien los positivos y mal los negativos y otra red (red2) clasificó muy mal los positivos y bien los negativos. Los errores cuadráticos medios de ambas redes podrían ser parecidos o uno mucho mayor que otro, pero el resultado de su comparación no discrimina entre los resultados de los datos positivos y los resultados de los datos negativos, lo que constituye una información importante. Una posible solución consiste en computar un error cuadrático medio para los patrones positivos y otro para los negativos. Nos vemos entonces en la necesidad de distinguir los errores de los datos positivos de los de los datos negativos.

Coefficiente de Correlación

Los resultados de cualquier método predictivo que clasifica un conjunto de instancias en dos conjuntos (positivos y negativos), caen en cuatro categorías: **verdaderos positivos**: predicciones que afirman la presencia de una propiedad existente en la realidad, **verdaderos negativos**: predicciones que afirman la ausencia de una propiedad inexistente en la realidad, **falsos positivos**: predicciones que afirman la presencia de una propiedad inexistente en la realidad y **falsos negativos**: predicciones que afirman la ausencia de una propiedad existente en la realidad. De ahora en más las llamaremos **TP**, **TN**, **FP** y **FN** respectivamente, a partir de los términos en inglés: *true positives*, *true negatives*, *false positives* y *false negatives*.

Una manera de determinar la bondad de la clasificación de una red u otro método predictivo es a partir de los valores de estos indicadores.

En nuestro caso, la predicción y la realidad son dos variables binarias con valores *promotor* o *no promotor*, por lo tanto se puede utilizar una tabla de contingencia de 2 x 2 para representar la relación entre la realidad y la predicción en un conjunto de test. En esta tabla se muestran los cuatro posibles resultados de la realización de una predicción. Cada elemento de la matriz muestra la cantidad de ejemplos de test para los cuales la realidad está representada en la columna y la predicción en la fila.

Donde,

$$\begin{aligned} TP + FN &= \text{promotores en la realidad} \\ FP + TN &= \text{no promotores en la realidad} \end{aligned} \tag{2.7}$$

⁶Denominamos *positivos* a los datos que satisfacen una condición y *negativos* a aquellos que no lo hacen. En el problema tratado los datos positivos son los promotores y los negativos aquellos, a los que consideramos no promotores.

| | | REALIDAD | | |
|------------|-------------|-----------|-------------|-----------|
| | | promotor | no promotor | |
| PREDICCIÓN | promotor | TP | FP | $TP + FP$ |
| | no promotor | FN | TN | $FN + TN$ |
| | | $TP + FN$ | $FP + TN$ | |

Tabla 2.1: Tabla de contingencia de 2x2.

Se puede ver que $TP + TN + FP + FN = N$, donde N es la cantidad de predicciones realizadas.

La comparación de dos redes a partir de los cuatro indicadores mencionados arriba no es trivial: se pretende que la cantidad de falsos positivos sea pequeña y que la cantidad de verdaderos positivos sea alta. Sin embargo ambas medidas están relacionadas (con más TP hay también más FP), entonces encontrar una buena solución sin resignar demasiado estas medidas no es tan sencillo.

Se han propuesto muchas medidas a tal efecto, probablemente las más utilizadas sean la **Sensitividad (Sn)** y la **Especificidad (Sp)** (Ecuación 2.8), que en nuestro caso representan la proporción de promotores correctamente predichos y la proporción de no promotores correctamente predichos respectivamente. Sin embargo seguimos teniendo dos medidas y lo ideal sería encontrar una medida escalar que capture toda esta información.

$$\begin{aligned} Sn &= \frac{TP}{TP+FN} \\ Sp &= \frac{TN}{TN+FP} \end{aligned} \quad (2.8)$$

Otra posible medida es calcular la correlación entre la predicción y la realidad. Una métrica que devuelve este resultado es el coeficiente de correlación (CC). Si P_i representa la predicción del patrón i ($P_i = 1$ si se predice que es un promotor y $P_i = 0$ si se predice que es no promotor) y S_i representa la realidad ($S_i = 1$ si es un promotor y $S_i = 0$ si no es un promotor). La correlación entre la predicción P_i y la realidad S_i está dada por la siguiente fórmula:

$$CC = \frac{\sum_{i=1}^N (S_i - \bar{S})(P_i - \bar{P})}{\sqrt{\sum_{i=1}^N (S_i - \bar{S})^2 \sum_{i=1}^N (P_i - \bar{P})^2}} \quad (2.9)$$

La Ecuación 2.9 es la ecuación del coeficiente de correlación, que es también conocido como el *coeficiente de correlación producto momento* y *coeficiente de correlación de Pearson*. Proviene de restarle la media de la variable a cada observación y dividir este valor por la desviación estándar, de esta forma se obtienen valores con igual media y desviación estándar, $Z_{si} = \frac{S_i - \bar{S}}{\sigma_S}$, $Z_{pi} = \frac{P_i - \bar{P}}{\sigma_P}$. El promedio de la multiplicación de los

valores estandarizados de las observaciones de las dos variables arroja el coeficiente de correlación mostrado en la Fórmula 2.9, $CC = \frac{\sum_{i=1}^N (Z_{si} \times Z_{pi})}{N}$.

En nuestro caso se cumple que

$$P_i \text{ y } S_i \text{ valen } 0 \text{ ó } 1. \quad (2.10)$$

A partir de esto podemos afirmar que se cumple

$$S_i \cdot P_i = 1 \Leftrightarrow S_i = 1 \wedge P_i = 1, \quad (2.11)$$

lo que implica que

$$\sum_{i=1}^N (S_i \cdot P_i) = TP. \quad (2.12)$$

Por otro lado, a partir de 2.10 también se llega a

$$S_i^2 = S_i. \quad (2.13)$$

A partir de 2.9, usando 2.12 y 2.13 se llega a la siguiente fórmula:

$$CC = \frac{(TP/N - \overline{PS})}{\sqrt{\overline{PS}(1 - \overline{S})(1 - \overline{P})}}. \quad (2.14)$$

Por otro lado,

$$\begin{aligned} \overline{S} &= \sum_{i=1}^N \frac{S_i}{N} \\ \overline{P} &= \sum_{i=1}^N \frac{P_i}{N} \end{aligned} \quad (2.15)$$

y utilizando 2.10, se puede ver que

$$\begin{aligned} \sum_{i=1}^N \frac{S_i}{N} &= \frac{TP+FN}{N} \\ \sum_{i=1}^N \frac{P_i}{N} &= \frac{TP+FP}{N}; \end{aligned} \quad (2.16)$$

utilizando 2.15 y 2.16 llegamos a

$$\begin{aligned} \overline{S} &= \frac{TP+FN}{N} \\ \overline{P} &= \frac{TP+FP}{N} \end{aligned} \quad (2.17)$$

Finalmente, utilizando las Ecuaciones 2.14 y 2.17, se llega a la fórmula del coeficiente de correlación más utilizada en este tipo de estudios [56, 72, 63, 47, 84, 83]:

$$CC = \frac{(TP \times TN) - (FN \times FP)}{\sqrt{(TP + FN) \times (TN + FP) \times (TP + FP) \times (TN + FN)}} \quad (2.18)$$

El CC varía entre 1 para resultados perfectamente correlacionados, pasando por 0 cuando no hay ninguna correlación y -1 cuando los resultados están perfectamente correlacionados negativamente.

Matthews [47] introdujo el uso del coeficiente de correlación para dos variables binarias mostrado en la Ecuación 2.18 a partir del coeficiente de correlación de Pearson mostrado en la Ecuación 2.9.

El CC es utilizado en muchos trabajos y está consensuado como una manera de comparar los resultados de redes neuronales y otros métodos predictivos [12], tiene la ventaja de considerar la relación entre los TP, TN, FP y FN en una sola medida y mide lo que buscamos: la correlación entra la predicción y la realidad.

Sin embargo, la Ecuación 2.18 tiene distintos resultados si la relación de ejemplos positivos vs. negativos difiere. En el CC pesan más aquellas categorías para las cuales hay más ejemplos. A modo de ejemplo del problema, analicemos el siguiente caso: si se tienen 40 TP y 80 FN, con un total de 120 positivos, el CC es distinto que si se tienen 10 veces más positivos, con resultados distribuidos de la misma manera: 400 TP y 800 FN, manteniendo la cantidad de TN y FP constantes. En ambos casos, la proporción de positivos correcta e incorrectamente predicha es la misma y la proporción de negativos correcta e incorrectamente predicha es la misma, sin embargo el CC difiere.

Dado que una métrica de evaluación de los resultados de un método predictivo debería ser independiente de la cantidad de positivos y negativos con los que fue probado, proponemos una medida similar a la existente que estandariza las entradas de forma tal, que no pese más la categoría para la cuál se utilizaron más datos en el conjunto de test. Llamamos a esta métrica Coeficiente de Correlación Estandarizado (SCC del inglés *Standardized Correlation Coefficient*).

Lo que hacemos es considerar la magnitud de los conjuntos de datos positivos y negativos utilizados, calculando los porcentajes de promotores y no promotores correctamente e incorrectamente predichos, como puede verse en las Fórmulas 2.19. En 2.21 se muestra la fórmula resultante.

$$TP' = \frac{TP \cdot 100}{p}, \quad TN' = \frac{TN \cdot 100}{r}, \quad FP' = \frac{FP \cdot 100}{r}, \quad FN' = \frac{FN \cdot 100}{p}, \quad (2.19)$$

donde p es la *cantidad de promotores* y r es la *cantidad de no promotores*. TP' es el porcentaje de promotores predichos correctamente, TN' es el porcentaje de no promotores predichos correctamente, FP' es el porcentaje de no promotores predichos incorrectamente y FN' es el porcentaje de promotores predichos incorrectamente. Nótese que:

$$TP' = Sn \times 100, \quad TN' = Sp \times 100, \quad FP' = (1 - Sp) \times 100 \quad \text{y} \quad FN' = (1 - Sn) \times 100. \quad (2.20)$$

$$SCC = \frac{(TP' \times TN') - (FN' \times FP')}{\sqrt{(TP' + FN') \times (TN' + FP') \times (TP' + FP') \times (TN' + FN')}} \quad (2.21)$$

Reemplazando los términos de la Ecuación 2.21 por las Ecuaciones 2.19 y asumiendo que $r = p \cdot q$, llegamos a

$$SCC = \frac{(TP \times TN) - (FN \times FP)}{\sqrt{p^2 \times (q \cdot TP + FP) \times (TN + q \cdot FN)}}. \quad (2.22)$$

El CC también puede escribirse como en la Ecuación 2.23.

$$CC = \frac{(TP \times TN) - (FN \times FP)}{\sqrt{q \cdot p^2 \times (TP + FP) \times (TN + FN)}} \quad (2.23)$$

A partir de las Ecuaciones 2.22 y 2.23 se puede ver que lo que estamos haciendo con el SCC es simplemente calcular los resultados para una relación positivos:negativos 1:1.

Otras medidas de calidad de las predicciones

Existen muchas otras métricas para la evaluación de la calidad de las predicciones de un método. A continuación explicamos varias de éstas, cuyas fórmulas pueden verse en la Tabla 2.2.

Consideremos el error cuadrático medio introducido más arriba. Si se determinan los FP y FN a partir de las predicciones, se puede calcular el ECM como $1/2 \times (FP + FN)/N$.

Burset y Guigó [12] computan la especificidad con una fórmula alternativa, a la que llamamos Sp_alt (Ecuación 2.24). Esta métrica es denominada *Positive Predictive Value* (PPV) por Benítez Bellón et al. [8], *Sen2* por Guigó et al. [23] y *Specificity* por Snyder y Stormo [72] y por Dong y Searls [17].

$$Sp_alt = \frac{TP}{TP + FP} \quad (2.24)$$

Esta medida representa de las secuencias predichas positivas, la proporción correctamente predicha (o la probabilidad de que una secuencia sea positiva dado que fue predicha como positiva). Se la puede ver también como una medida de cuán cuidadosa es una herramienta de forma tal de no hacer predicciones falsas.

El *Accuracy* [12] utilizado por Burset y Guigó [12] y Benítez-Bellón et al. [8] mide la proporción de todas las predicciones correctas. Benitez Bellon et al. [8] proponen también el *Overall Performance*, que es un promedio entre el *Accuracy* y el *Positive Predictive Value*(PPV) y mencionan como métrica útil el producto de estas medidas ($Accuracy \times PPV$).

Matthews [47] propone el uso del CC, como mencionamos anteriormente, e informa los valores de $Sn \times 100$, $Sp \times 100$ y CC en la evaluación de distintos métodos de predicción de hélices para el *T4 phage lysozyme*.

El CC se indefine cuando $TP + FN$, $FP + TN$, $TP + FP$ o $FN + TN$ valen cero (en el conjunto de test no hay positivos, no hay negativos, no hay predicciones positivas o no hay predicciones negativas). Por esto Burset y Guigó [12] proponen el uso de otras medidas de calidad de las predicciones, que puedan ser computadas en cualquier circunstancia: el *Simple matching coefficient* (SMC), que es la probabilidad de predicción correcta y el *Average Conditional Probability*, que llevado a un rango entre -1 y 1 lo denominan *Approximate Correlation*. Las medidas que finalmente utilizan para evaluar la performance de programas de predicción de estructura de genes son: Sn, Sp_{alt}, AC y CC.

Wu y McLarty [84] mencionan las siguientes medidas como mecanismos para la medición de la calidad de una predicción particular: Sn, Sp, PPV, *Negative Predictive Value* (NPV), *Accuracy* y CC. Witten y Frank [83] mencionan varias de estas medidas, algunas con distintos nombres.

En el Apéndice A se muestran y analizan brevemente los distintos indicadores aplicados a los resultados de uno de los métodos computacionales para el reconocimiento de promotores.

| Sigla | Nombre Completo | Fórmula | Referencias |
|---------------------|--|--|-------------------------------|
| Sn | Sensitividad | $\frac{TP}{TP+FN}$ | [47, 12, 34, 8] |
| Sp | Especificidad | $\frac{TN}{TN+FP}$ | [47, 34, 8] |
| PPV o Sp-alt | Positive Predictive Value | $\frac{TP}{TP+FP}$ | [12, 72, 8, 23, 17] |
| Acc o SMC | Accuracy o Simple Matching Coefficient | $\frac{TP+TN}{TP+FN+FP+TN}$ | [12, 8] |
| CC | Coefficiente de Correlación | $\frac{(TP \times TN) - (FN \times FP)}{\sqrt{(TP+FN) \times (TN+FP) \times (TP+FP) \times (TN+FN)}}$ | [56, 72, 63], [47, 84, 83] |
| ACP | Average Conditional Probability | $\frac{1}{4} \left(\frac{TP}{TP+FN} + \frac{TP}{TP+FP} + \frac{TN}{TN+FP} + \frac{TN}{TN+FN} \right)$ | [12] |
| AC | Approximate Correlation | $(ACP - 0,5) \times 2$ | [12] |
| OP | Overall performance | $\frac{Accuracy+PPV}{2}$ | [8] |
| AccPPV | | $Accuracy \times PPV$ | [8] |
| Per | Performance | $\frac{TP}{TP+FP} + \frac{Sn}{2}$ | [34] |
| ECM | Error cuadrático medio | $\frac{FP+FN}{2 \cdot N}$ | |
| SCC | Standardized correlation coefficient | $\frac{(TP' \times TN') - (FN' \times FP')}{\sqrt{(TP'+FN') \times (TN'+FP') \times (TP'+FP') \times (TN'+FN')}};$ $TP' = \frac{TP \cdot 100}{TP+FN}, TN' = \frac{TN \cdot 100}{TN+FP}, FP' = \frac{FP \cdot 100}{TN+FP}, FN' = \frac{FN \cdot 100}{TP+FN}$ | |

Tabla 2.2: Medidas de calidad de predicciones.

2.6. Breve revisión de soluciones existentes basadas en redes neuronales

Antes de comentar las soluciones existentes al problema de reconocimiento de promotores basadas en redes neuronales, mencionamos dos resultados obtenidos mediante métodos estadísticos.

En 1984 Staden [73] describe métodos computacionales para encontrar señales –entre otros, promotores– en secuencias de ácidos nucleicos. Para esto utiliza una matriz de pesos generada a partir de la distribución de bases en las secuencias de la compilación de Hawley y McClure. Los resultados de la utilización de la matriz de pesos para la clasificación de un conjunto de promotores de test, arrojan 81.6 % TP y 0.84 % FP.

En el mismo año, Mulligan et al. [51] utilizan otra matriz de pesos, basada en los mismos datos de Hawley y McClure para computar la búsqueda y evaluación de posibles promotores de *E. coli*. Los resultados, parecidos a los de Staden, arrojan 80.6 % TP y 0.85 % FP.

Hay muchas publicaciones referidas a la aplicación de redes neuronales al problema de reconocimiento de promotores en *E. coli*. En general se usaron perceptrones simples o de dos capas con *back-propagation* y una representación de 4 unidades para la entrada. Las entradas consideraron entre 44 y 58 bases. Hirst y Sternberg [32] y Presnell y Cohen [59] mencionan varias de ellas. También se comenta un trabajo en que se utilizaron redes neuronales con retardo temporal para el reconocimiento de promotores en eucariotas. Fickett y Hatzigeorgiou [19] presentan una revisión de métodos de reconocimiento de promotores en eucariotas.

Nakata et al. [52] utilizan un perceptrón simple en combinación con métodos de análisis bioquímicos, entre otros: ángulo de torsión, mapa de estabilidad térmica, temperatura de derretimiento, torcedura de la hélice. Como datos utilizan promotores de la compilación de Hawley y McClure. Para el entrenamiento utilizaron 57 promotores y 59 no promotores y para test 33 promotores y 43 no promotores. Los no promotores fueron generados aleatoriamente. Los resultados a los que llegaron para el conjunto de test son: 67 % de exactitud con el perceptrón sólo y 75 % si se tienen en cuenta los análisis bioquímicos.

Lukashin et al. [43] y Demeler y Zhou [16] atacan el mismo problema con redes neuronales más elaboradas. En ambos casos se implementaron dos redes neuronales *feedforward* de dos capas para el reconocimiento de las secuencias conservadas correspondientes a las regiones -10 y -35 . La salida de cada una de las redes se conectó a una unidad de salida, que es la que clasifica a la secuencia. Lukashin et al. utilizaron datos de la compilación de Harley y Reynolds. Para el entrenamiento se utilizaron

25 promotores fuertes y 250 secuencias generadas aleatoriamente como datos de no promotores y para el test 222 promotores, entre los cuales estaban los 25 utilizados para el entrenamiento y 2220 secuencias generadas aleatoriamente con distribución uniforme de bases. Los resultados obtenidos para el conjunto de test tienen entre un 94 % y 99 % de TP con entre 2 y 6 % de FP. La decisión de incluir en el conjunto de test secuencias utilizadas en el conjunto de entrenamiento no es buena, dado que los resultados no están mostrando el poder de generalización de la red. Demeler y Zhou utilizaron datos de la compilación de Hawley y McClure [26]. Para el entrenamiento utilizaron los 80 promotores de bacterias y de fagos y como datos de no promotores generaron secuencias aleatoriamente. Para el test utilizaron los 30 promotores de plásmidos, transposones y los creados por fusión o mutación y 1500 secuencias generadas aleatoriamente como no promotores. El tamaño de la ventana (cantidad de nucleótidos de entrada) fue de 44 pb. Se realizaron pruebas con distintas relaciones promotor:no promotor en el conjunto de entrenamiento (entre 1:1 y 1:20) y variando la cantidad de neuronas de la capa oculta. Variaron tiempos de entrenamiento y el esquema de codificación de los nucleótidos entre 2 y 4 bits, la última dio mejores resultados. Llama la atención que en el estudio, las diferencias en los resultados obtenidos al variar la cantidad de unidades en la capa oculta no sean significativas. La variación en la relación promotor:no promotor, en cambio, sí lo es. Si bien los resultados de la predicción de promotores para el conjunto de entrenamiento es buena, no informan los resultados para un conjunto de test independiente, que es una medida mucho más importante en este caso. Tampoco parece acertado separar los datos de promotores y fagos para el entrenamiento y de plásmidos y transposones para el test, posiblemente sería mejor utilizar una distribución uniforme, en donde haya datos de los cuatro tipos de ADN en ambos conjuntos. Por último, a pesar de las características negativas mencionadas, algo importante a destacar de este trabajo son las pruebas realizadas para encontrar una arquitectura óptima.

O'Neill [54, 55] es el primero en tratar con la variación en el espaciado entre las secuencias conservadas en las posiciones -10 y -35 , modelando y entrenando distintas redes neuronales para las distintas distancias posibles. Utiliza en [55] los datos de las compilaciones de Hawley y McClure y Harley y Reynolds. Como datos negativos utiliza datos generados aleatoriamente y con un preprocesamiento elimina secuencias “parecidas” a promotores, cuyo parecido es determinado a partir de otro programa de búsqueda de promotores. Entrenó tres redes *feedforward*, una para cada una de las distancias más frecuentes. En particular, la entrenada para el reconocimiento de secuencias con espaciado de 17 pb tiene 80 % de TP y menos que 0.1 % de FP [54]. La ventana de entrada considera 58 bases. Usa 5148 secuencias provenientes de 39 secuencias de promotores como positivos y 4000 secuencias (generadas aleatoriamente con 60 % de probabilidad de aparición de nucleótidos AT) como negativos. La ventana

de entrada considera 58 bases. La generación de 5148 secuencias a partir de transformaciones de las 39 secuencias originales no parece ser útil. De esta forma se puede estar trabajando con datos distantes de los reales (se consideran como promotores cadenas, que no se sabe si realmente lo son, además no se consideran en el estudio muchos promotores ya conocidos en la época). Por otro lado el preprocesamiento de secuencias generadas al azar con un programa de búsqueda de promotores puede ser útil, sin embargo, si el programa es bueno, podría directamente utilizarse ese para la predicción de promotores, y si no lo es puede estar agregando más ruido a los datos, que los que estos tendrían de no haberse aplicado el programa.

Horton y Kanehisa [33] desarrollaron una red neuronal podada [27, 31]. El trabajo se basa en un perceptrón en el que se borran los pesos, que durante el entrenamiento se manifiestan débiles. Utilizan los datos de la compilación de Harley y Reynolds. Como no promotores utilizan secuencias de regiones codificantes tomadas de GenBank⁷. Utilizan un código de 7 dimensiones para representar información de las bases y otros 7 bits para incluir información del entorno de las bases por cada ventana de 12 nucleótidos. De esta forma se puede informar, por ejemplo si la base está en una región rica en nucleótidos GC. Los resultados de su método son parecidos a los del método estadístico de Mulligan, un 80.6% de TP y con 0.86% de FP. Un trabajo interesante que realizaron fue mostrar la correlación entre las tasas de predicción de redes implementadas por otros investigadores y el *information content* de los conjuntos utilizados en cada caso.

Mahadelian y Ghosh [45] realizan la combinación de dos perceptrones simples con *back-propagation*, para buscar promotores con distancias de entre 15 y 21 pares de bases entre los consensos. La primer red neuronal predice los consensos de las regiones -10 y -35 . Luego se alinean los promotores y una segunda RN hace predicciones sobre una secuencia de 65 bases. Como entrenamiento se utilizaron 106 promotores y secuencias generadas aleatoriamente con 60% de probabilidad de aparición de As y Ts. Para test se utilizaron 126 promotores de bacterias, mutantes y fagos y 5000 aleatorios. Los resultados son: 98% TP en reconocimiento de promotores y 90.2% en reconocimiento de no promotores.

Pedersen y Engelbrecht [56] desarrollaron redes neuronales *feedforward* de dos capas para el análisis de promotores de *E. coli*. Predicen el TSS y miden *information content*. Utilizaron dos esquemas de codificación, uno con ventanas de entre 1 y 51 nucleótidos y el otro con una ventana de 65 nucleótidos conteniendo un “agujero” de 7 nucleótidos (i.e. hay 7 nucleótidos de la entrada que no están conectados con la capa oculta).

Reese [63] presenta una arquitectura de red neuronal con retardo temporal para el

⁷<http://www.ncbi.nlm.nih.gov/Genbank/>.

reconocimiento de promotores en el genoma de la *Drosophila melanogaster*, que es un organismo eucariota. Utiliza dos capas, una para el reconocimiento del TATA Box y otra para el reconocimiento del Inr. Previamente, entrena dos perceptrones simples, para el reconocimiento de cada una de estas secuencias y luego introduce los pesos resultantes como inicialización de los pesos de una tercer red, a la que entrena.

A los métodos enunciados anteriormente se les pueden hacer las siguientes críticas: varios trabajan con datos de la compilación de Hawley y McClure, cuando ya Harley y Reynolds habían publicado la suya. Los resultados que mencionamos son los informados por los autores. Muchos son poco confiables debido a que se utilizaron secuencias pertenecientes al conjunto de entrenamiento para el armado del conjunto de test. Los resultados no son comparables entre sí, para serlo deberían probarse con un mismo conjunto de datos y con las mismas medidas de calidad de las predicciones.

En nuestra propuesta procuramos que los conjuntos de entrenamiento y test estén equilibrados en cuanto a la cantidad de promotores fuertes y débiles. Utilizamos promotores exclusivamente de *E. coli*, sin plásmidos ni fagos, tampoco consideramos mutaciones, ya que las mutaciones puntuales nos hacen tener datos muy parecidos como entrada, habiendo entonces información casi duplicada sólo para aquellas secuencias en las que hubo mutaciones. Los datos utilizados en los conjuntos de entrenamiento y test son independientes (en el conjunto de test no hay secuencias utilizadas para el entrenamiento). Consideramos los datos más nuevos y correctos. No utilizamos un perceptrón simple, sino una red más compleja, que se adecua mejor al problema. Utilizamos información biológica para la construcción de la arquitectura.

Capítulo 3

CPR: un método conexionista para el reconocimiento de promotores

Como se explicó en el Capítulo 1, el problema de reconocimiento de secuencias de promotores en ADN no es un problema de clasificación de secuencias convencional. Los patrones biológicos a reconocer son imprecisos, por ejemplo la región -10 de un promotor puede estar compuesta por la secuencia TATGAT en lugar de TATAAT. A su vez, la distancia entre los motivos no es fija.

Con el objetivo de encontrar una solución al problema de clasificación de secuencias de ADN de procariotas en dos conjuntos (con y sin promotores) diseñamos una red neuronal con retardo temporal, que de aquí en adelante llamaremos TDNN, del inglés *Time Delay Neural Network*. Denominamos a este método *Reconocimiento conexionista de promotores* o *Conexionist Promoter Recognition* (CPR). El CPR mejora los resultados que se pueden obtener para el mismo problema con los métodos mencionados en el Capítulo 2. Esto lo mostraremos en el próximo capítulo.

En este capítulo explicamos los pasos seguidos para el diseño de la red. En la Sección 3.1 abordamos los problemas existentes para el reconocimiento de promotores desde el punto de vista biológico y computacional (ver también Capítulos 1 y 2). En la Sección 3.2 presentamos el método a utilizar. Aquí introducimos las TDNN y explicamos la adecuación de este modelo al problema concreto de predicción de promotores procariotas, la arquitectura seleccionada y el estudio sistemático realizado para optimizar la arquitectura de la red. Finalmente, en la Sección 3.3 mostramos y analizamos los resultados a los que llegamos con las pruebas realizadas para la optimización de la arquitectura de la red y de los datos a presentar, mediante un proceso al que denominamos *tuning*.

3.1. Problema

La clasificación de secuencias de promotores no es una clasificación tradicional [49]. Los promotores son patrones compuestos por dos subsecuencias conservadas. Podríamos, entonces, pensar que se tienen que buscar dos subsecuencias dentro de un patrón. Sin embargo no es trivial definir esta búsqueda, dado que las subsecuencias están vagamente definidas, lo que dificulta la posibilidad de determinar si una subsecuencia se corresponde o no con las secuencias conservadas con consenso TATAAT o TTGACA. Por ejemplo, TACTAT y TAAAGT son las regiones -10 o TATAAT definidas por Harley y Reynolds para los promotores *araI(c)X(c)* y *trxA*, y CTGGCG y TTTACG las definidas para la región -35 o TTGACA de los mismos promotores. Por otro lado, la variabilidad en la distancia entre ambas secuencias dificulta más la búsqueda de soluciones, dado que es una variable más a considerar. Por ejemplo, de encontrar una subsecuencia a con un “parecido” a TATAAT y dos secuencias b y c con un “parecido” a TTGACA, en que la distancia entre a y b es de 19 pb y la distancia entre a y c es de 15 pb, habría que definir un criterio acerca de cuál resulta mejor.

Las redes neuronales son apropiadas para el reconocimiento de los patrones vagos o *fuzzy*, en que cada nucleótido aparece con una determinada frecuencia y para los cuales sólo se cuenta con modelos probabilísticos [26, 25, 42]. Estas capturan la imprecisión y devuelven un valor de similitud. Por otro lado, es posible que existan otras características en las secuencias de promotores que aún no han sido encontradas, por ejemplo que las subsecuencias conservadas no tienen 6, sino que tienen más nucleótidos. Dado que la construcción de la red no está basada en un modelo preciso del problema, sino que se logra a través de un aprendizaje a partir de los ejemplos tomados como entrada, es posible que una red neuronal encuentre estas relaciones y haga inferencias a partir de ellas. Sin embargo, no todas las redes neuronales pueden tratar con el problema del espacio variable entre las dos secuencias conservadas. Una arquitectura TDNN es apropiada para esta problemática, debido a que permite encontrar patrones con corrimiento en el espacio, como explicaremos en la siguiente sección.

La selección adecuada de conjuntos de entrenamiento, validación y test influye en la capacidad de aprendizaje y generalización de las redes neuronales [84]. Una buena distribución de los datos de promotores entre estos conjuntos es importante. Si se entrena con datos de una especie y se testea con datos de otra especie u otro tipo de ADN, cuyas secuencias difieren, posiblemente los resultados no sean buenos. Por ejemplo, si tenemos un conjunto de secuencias con datos de fagos y de promotores de *E. coli*, lo ideal sería que los datos de ambos estén distribuidos en los conjuntos de entrenamiento y de test, y no que el conjunto de test esté compuesto exclusivamente por promotores de fagos y el de entrenamiento por promotores de *E. coli* [83]. Por último,

la relación promotores:no promotores a utilizar en el conjunto de entrenamiento de una red neuronal que hace clasificaciones con aprendizaje supervisado es otro tema a considerar [16, 1]. Como explicaremos más adelante, si utilizamos muchos más datos de no promotores que datos de promotores es muy posible que se aprenda a reconocer mejor los primeros.

A partir de ahora cuando nos referimos al conocimiento de los promotores lo hacemos, a no ser que se indique lo contrario, a las compilaciones de Hawley y McClure [26], Harley y Reynolds [25] y Lisser y Margalit [42]. Recordemos que a partir de éstas se sabe que los promotores tienen dos secuencias conservadas de largo 6 con distancia variable entre 15 y 21 pb entre ellas y el TSS aproximadamente 12 pb aguas abajo de la región -10 .

3.2. Método

Considerando los problemas planteados en la sección anterior mostramos a continuación el método utilizado para resolverlos. Describiremos la topología de las redes neuronales *Time Delayed* (TD), el algoritmo de aprendizaje que utilizan y la arquitectura de red propuesta para la solución al problema planteado a partir del conocimiento que se tiene de los promotores. Explicamos también las pruebas realizadas para ajustar la arquitectura de la red y los parámetros utilizados durante el entrenamiento.

Las TDNN son redes multicapa *feedforward*, que mediante una topología especial y una adecuación del algoritmo de aprendizaje de *back-propagation* logran detectar subsecuencias de la secuencia presentada como entrada, independientemente de su posición en la entrada. Denominamos a estas subsecuencias *features*. Un ejemplo del problema de detección de *features* independientemente de su posición en la entrada puede verse en la Figura 3.1. En este caso se desea reconocer el patrón TTGA, cualquiera sea su ubicación en las secuencias.

Cadena 1: **TTG**AAAAAAGTT

Cadena 2: GG**TTG**AAAAAAG

Figura 3.1: Detección de *features* independientemente de su posición en la entrada.

Para desarrollar detectores de *features* localizados, las redes TD restringen la conectividad entre las capas, de forma tal que cada unidad oculta se conecte a un conjunto limitado contiguo de las neuronas de la capa de entrada, denominado *receptive field*. Todas las unidades ocultas de una misma capa tienen los mismos pesos sinápticos en sus conexiones a los *receptive fields* [40]. A esta característica se

la denomina *weight sharing*. De esta forma, cada capa oculta está compuesta por una unidad replicada varias veces. Las réplicas de una unidad simulan el efecto de que ésta se va moviendo por la entrada para detectar un mismo *feature* en distintas posiciones. Es por esto que a las unidades ocultas las denominamos *feature detectors*. En la Figura 3.2 se puede ver un ejemplo de una TDNN con *receptive fields* y *weight sharing*.

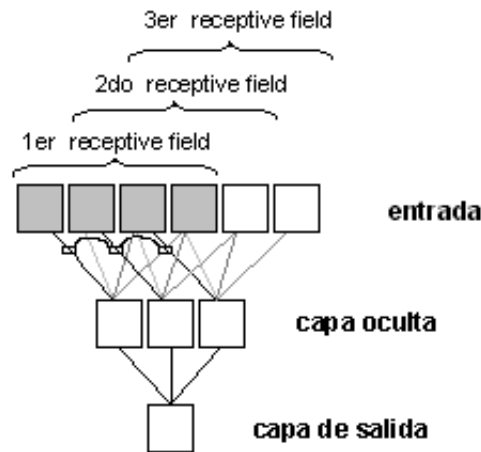


Figura 3.2: Red neuronal de 2 capas. La red tiene 6 unidades de entrada, 3 neuronas en la capa intermedia y una neurona en la capa de salida. Las neuronas de la capa intermedia no se conectan a todas las neuronas de la capa de entrada, sino sólo a una porción contigua de ésta, de 4 unidades, denominada *receptive field*. Dado que hay 6 entradas y el tamaño de los *receptive fields* es de 4 unidades, son necesarias $6 - 4 + 1 = 3$ neuronas en la capa intermedia, una conectada a cada uno de los *receptive fields*. Las unidades correspondientes al primer *receptive field* están dibujadas con color gris. Las neuronas de la capa oculta utilizan *weight sharing*, de esta forma el peso de la conexión entre la primer neurona de la capa oculta y la primer unidad del primer *receptive field* es igual al peso de la conexión entre la segunda neurona de la capa oculta y la primer unidad del segundo *receptive field* y al peso de la conexión entre la tercer neurona de la capa oculta y la primer unidad del tercer *receptive field*. Estos pesos están en color negro en la figura. Lo mismo sucede con las restantes unidades de los *receptive fields*. Llamamos a éstas **conexiones correspondientes** o *time-shifted*. Las neuronas de cada capa se numeran de izquierda a derecha.

Como **algoritmo de aprendizaje** se utiliza una modificación del algoritmo de *back-propagation*, al que llamamos *time delayed back-propagation*. Se hacen los pasos *forward* y *backward* del algoritmo, considerando todas las réplicas de las neuronas de la capas intermedias como si fuesen distintas neuronas. Esto origina distintos valores de corrección para las distintas conexiones. En lugar de cambiar los pesos de las

conexiones *time-shifted*¹ de manera separada, se actualizan todos ellos con un mismo valor: el promedio de los cambios de estos pesos. Esto hace que la red encuentre los *features* en la entrada sin importar en que lugar ocurrieron. En la Figura 3.5 se puede ver el pseudocódigo de algoritmo de aprendizaje utilizado, que explicaremos con más detalle más adelante.

Las TDNNs fueron descritas originalmente por Waibel et al. (1989) [81] y Lang y Waibel (1990)[39] para reconocimiento del habla (*speech recognition*). En [78] se puede ver una revisión de arquitecturas de redes neuronales TD utilizadas para este fin. Como ya se mencionó anteriormente, Reese [63] utiliza un TDNN para el descubrimiento de promotores en el genoma de la drosophila *Melanogaster* (eucariota). Utiliza dos capas para reconocer la TATA Box y el Inr. También se han utilizado para otros dominios [7].

CPR: Aplicación de una TDNN al reconocimiento de promotores en procariontas

La entrada de la red considera la cantidad de nucleótidos suficiente para representar un promotor. Se construyen dos capas ocultas para el reconocimiento de cada una de las subsecuencias conservadas: TATAAT y TTGACA. Se decidió no agregar una tercer capa para la señal CAP, debido a que ésta es muy débil, se encuentra muy fácilmente (ya que sólo tiene 2 bases) y el error experimental para determinar el TSS es de aproximadamente 2 pb [25], con lo que estimamos que sólo agregaría complejidad al problema y no significaría un aporte a la solución. Cada una de las capas ocultas tiene un *feature detector* replicado en el espacio para el reconocimiento de las secuencias consenso. Ambas capas ocultas se unen con una unidad de salida, cuyo valor de salida se pretende que sea 1 para un promotor y 0 para un no promotor. A continuación explicaremos la representación utilizada para los datos de entrada de la red y daremos detalles de la arquitectura.

Representación de datos de la entrada. Para la representación de los datos de entrada de la red utilizamos 4 unidades binarias por base, que están *prendidas* o *apagadas* (con valores 1 ó 0) de acuerdo al nucleótido que están representando. La representación de nucleótidos utilizada es la siguiente: A: (1, 0, 0, 0); C: (0, 1, 0, 0); G: (0, 0, 1, 0) y T: (0, 0, 0, 1). Se puede ver un ejemplo en la Figura 3.3. Esta representación denominada ortogonal, en la que las letras X_1, X_2, \dots, X_N del alfabeto a considerar (nucleótidos, aminoácidos, alfabeto inglés, etc.)² se codifican mediante los vectores binarios ortogonales $(1,0,\dots,0), (0,1,\dots,0), \dots (0,0, \dots, 1)$, tiene la virtud de que

¹Las conexiones *time-shifted* son aquellas que ocupan la misma posición relativa en distintos *receptive fields* (ver Figura 3.2).

²En nuestro caso el alfabeto está compuesto por nucleótidos.

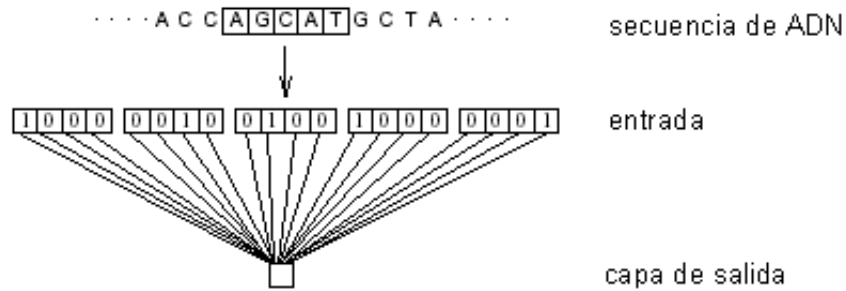


Figura 3.3: Ejemplo de representación de los datos de entrada de la red- Se muestra un perceptrón simple con una capa de entrada que considera 5 nucleótidos y una neurona en la capa de salida. Cada nucleótido de la cadena de ADN está representado por 4 entradas, por lo que se tienen $5 \times 4 = 20$ entradas. El esquema de codificación es: $A=(1,0,0,0)$, $C=(0,1,0,0)$, $G=(0,0,1,0)$ y $T=(0,0,0,1)$.

la distancia de Hamming³ entre todos los pares de vectores es la misma. Esto garantiza que no haya correlación entre los vectores de codificación de los distintos nucleótidos, lo que podría sesgar el procedimiento de aprendizaje [84, 6]. Una desventaja de esta codificación con respecto a una representación con $\log_2 N$ unidades por cada letra del alfabeto es que requiere más conexiones ($N \times$ cantidad de letras del alfabeto utilizadas como entrada de la red). Otra posible representación, que requiere aún menos conexiones, es representar cada nucleótido como un número real, en una sola entrada. De todas formas, en nuestro caso esta desventaja no es significativa, ya que debido al uso de *weight sharing* la cantidad de pesos es reducida. De esta forma, la ventaja de que la representación no imponga una correlación entre los vectores de codificación parece más importante que el incremento en la cantidad de pesos de la red. Brunak et al. [10], Mache et al. [44] y Pedersen y Engelbrecht [56] también representan los nucleótidos utilizando 4 unidades por base. Bohr et al. [9] utilizan la representación ortogonal para codificar aminoácidos. Cada aminoácido es representado con 19 ceros y 1 sólo uno. Qian y Sejnowski [62] y Demeler y Zhou [16] probaron entrenar redes con representaciones ortogonales (con aminoácidos y nucleótidos respectivamente) y con otras representaciones (entre otras, que cada entrada representase dos aminoácidos o dos nucleótidos) obteniendo en ambos casos mejores resultados con la primer representación. Por último en NETtalk [70], Sejnowski y Rosenberg también utilizan una representación binaria ortogonal para la asignación de fonemas a letras del alfabeto inglés. En este caso se utilizan 29 entradas por letra. En la Figura 3.4 se puede ver la arquitectura de la red neuronal, que pasamos a describir.

³Cantidad de bits que difieren en dos strings o vectores binarios. En nuestro caso la distancia de Hamming entre dos vectores A y B de 4 dimensiones con valores binarios es $\sum_{i=1}^4 |A_i - B_i|$.

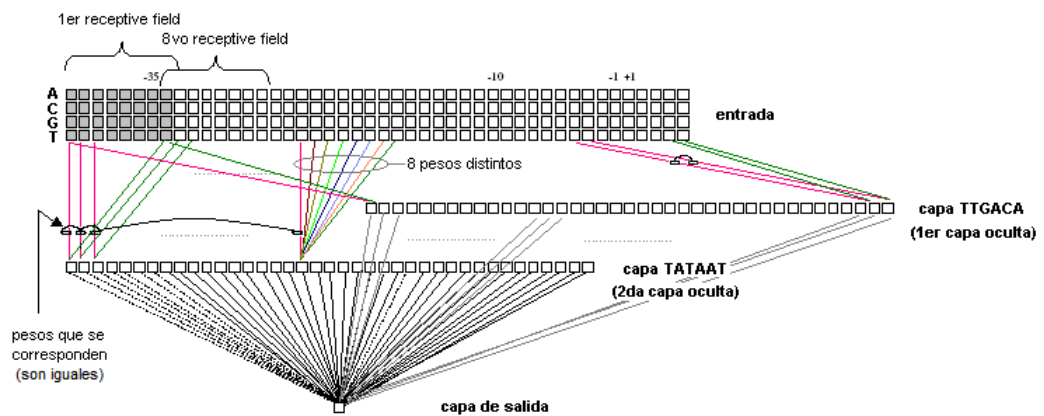


Figura 3.4: Arquitectura de la red neuronal. Los cuadrados simbolizan las unidades y las líneas las conexiones entre éstas. Arriba se muestra la capa de entrada, que considera 46 nucleótidos con una representación de 4 unidades por nucleótido. Más abajo se muestran las dos capas ocultas, denominadas *capa TATAAT* y *capa TTGACA*, y la capa de salida. Se puede apreciar el *weight sharing* de los distintos *receptive fields* marcado con distintos tonos para cada *correspondencia de pesos*. También se muestra el *receptive field* conectado a la primer neurona de cada una de las dos capas ocultas. A modo de simplificación se muestran las conexiones entre las unidades intermedias y los nucleótidos de entrada y no una conexión por cada una de las 4 unidades que representan a cada nucleótido.

Arquitectura

De acuerdo al conocimiento aportado por los análisis de datos realizados, mencionados en la Sección 3.1, un promotor consta en el peor caso⁴ de 6 pb (correspondientes a la región -35) + 6 pb (correspondientes a la región -10) + 21 pb (correspondientes al espacio entre las dos regiones) + 12 pb (correspondiente al espacio entre el TSS y la región -10) + 1 pb (correspondiente al TSS) = 46 pb. Por este motivo utilizamos una capa de entrada que considera 46 nucleótidos. Dado que cada nucleótido es representado por 4 unidades de entrada, tenemos una capa de entrada de $46 \times 4 = 184$ unidades. En muchos casos mencionaremos 46 entradas o nucleótidos, de esta forma nos estamos abstrayendo de la forma de representación de las mismas.

Como ya explicamos, la red neuronal tiene dos *feature detectors* replicados, dado que hay dos *features* a reconocer: los consensos TATAAT y TTGACA. Para detectar estos *features* se conecta cada unidad oculta a una porción contigua de nucleótidos de la entrada. La cantidad de nucleótidos que forman la porción contigua fue elegida a partir de la longitud de las secuencias consenso conocidas. De acuerdo a los análisis de las compilaciones de promotores de procariotas, los tamaños de los dos consensos más importantes son de 6 pares de bases cada uno. Sin embargo las secuencias conservadas podrían ser un poco más largas [25, 61], lo que nos motivó a realizar pruebas con *receptive fields* que considerasen 6 y 8 nucleótidos. Los resultados fueron levemente mejores para los segundos, lo que motivó a utilizar *receptive fields* de longitud 8 (32 unidades de entrada). De esta forma cada capa oculta tiene $46 - 8 + 1 = 39$ unidades. La capa de salida consiste en una unidad conectada a cada una de las 39 unidades de las capas ocultas.

En total hay $4 \times 46 + 2 \times 39 + 1 = 263$ unidades considerando la capa de entrada y $(1 \times 8 \times 4) \times 2 + (1 \times 39) \times 2 = 142$ pesos (ver Figura 3.4). De no haber utilizado *weight sharing* habría $(39 \times 46 \times 4) \times 2 + (1 \times 39) \times 2 = 14430$ pesos, o sea más de 100 veces la cantidad de pesos.

Como función de activación de las neuronas de las capas ocultas se utiliza la función logística presentada en la Ecuación 2.3. A la unidad de salida –una vez aplicada la función de activación logística– se le aplica una función escalón, que depende del valor de un umbral θ fijo (ver Fórmula 3.1). Las funciones aplicadas a los resultados de la función de activación de una neurona, reciben el nombre de *funciones de salida*.

$$f(O_1) = \begin{cases} 1 & \text{si } O_1 \geq \theta \\ 0 & \text{si } O_1 < \theta \end{cases} \quad (3.1)$$

Nos propusimos que la red maximice la correlación entre los valores predichos y los

⁴Con *peor caso* nos referimos al promotor con mayor longitud posible de acuerdo a los análisis de datos realizados.

reales. Dependiendo del umbral que se elija, esta relación cambia. Para la **elección del umbral** se presentaron todos los patrones del conjunto de test a la red. Los resultados arrojados por la red están entre el 0 y el 1, dado que son el resultado de la aplicación de la función logística a las entradas de la neurona de capa de salida. Tomamos el menor y el mayor de estos resultados y generamos valores pertenecientes a este rango, a los que utilizamos como umbrales de la función de salida de la neurona de salida de la red. Para cada uno de estos umbrales calculamos el coeficiente de correlación estandarizado (SCC), introducido en el Capítulo 2. Normalmente estamos interesados en el umbral que arrojó el SCC más alto y éste es el utilizado para dar el resultado final de la red. Sin embargo es común que existan otros requerimientos como ser un límite superior en la cantidad de FP y/o un límite inferior en la cantidad de TP. Es por esto que en los resultados de la experimentación mostramos los porcentajes de FP y TP y los SCCs para distintos umbrales.

La búsqueda de los promotores de procariotas mediante el uso de una arquitectura como la mencionada arriba tiene varias ventajas: por un lado cada capa oculta permite detectar un mismo *feature* a lo largo de toda la secuencia y es posible combinar el reconocimiento de los dos patrones conservados con la distancia variable entre los mismos. Por otro lado, la cantidad de pesos que se tienen al utilizar *weight sharing* y *receptive fields* pequeños es mucho menor a lo que sería de no contar con estas características. La limitación en la cantidad de parámetros libres de la red permite mejorar la generalización. Por otro lado se logra que la red no tenga que aprender esta información que ya se tiene [4].

Algoritmo de aprendizaje

En la Figura 3.5 mostramos el pseudocódigo del algoritmo de aprendizaje utilizado por la red y luego explicamos algunos pasos del mismo. Como se mencionó anteriormente, el algoritmo utilizado es una modificación del algoritmo de *back-propagation* (la modificación se realiza en la actualización de los pesos y puede verse en el paso 6). En la red, cada neurona de las capas ocultas 1 y 2 se conecta con un receptive field de la capa de entrada de 32 unidades. Cada capa oculta tiene 39 neuronas. Todas las neuronas de las capas ocultas se conectan con la neurona de salida. El subíndice j de los pesos $w_{ij}^{0,m}$ (peso entre la unidad j de la capa 0 y la unidad i de la capa m , $m \in \{1, 2\}$) se mueve entre 1 y 32 y el subíndice i entre 1 y 39. Por ejemplo, el subíndice i en w_{i2} es tal que $1 \leq i \leq 32$.

| |
|---|
| <p>Entrada: η: velocidad de aprendizaje, Conjunto de patrones de entrenamiento. $\mu = (\xi^\mu, \zeta^\mu)$ es un patrón del conjunto de entrenamiento, ξ es la entrada y ζ la salida esperada.</p> <p>Notación: V_i^m: salida de la unidad i de la capa m {Consideramos a los umbrales como otro peso de la red con entrada, entonces $V_0^m = -1$}, $w_{ij}^{n,m}$: peso sináptico entre neurona j de la capa n, hasta la neurona i de la capa m.</p> <ol style="list-style-type: none"> 1: Inicialización de los pesos y umbrales. 2: repetir {Presentación de los patrones del conjunto de entrenamiento ordenado de algún modo. Se le presenta a la red una época. Tomar un patrón μ y presentarlo a la entrada, de esta forma $V_k^0 = \xi_k^\mu$.} 3: mientras existe un patrón no procesado hacer 4: Paso forward. Calcular: $V_i^m = g(h_i^m) = g(\sum_{j=0}^{32} w_{ij}^{0,m} \cdot V_j^0)$, para $m \in \{1, 2\}$, $i \in \{1, 39\}$ $V_1^3 = g(h_1^3) = g((\sum_{j=1}^{39} (w_{1j}^{1,3} \cdot V_j^1 + w_{1j}^{2,3} \cdot V_j^2)) - w_{10}^3)$, para la capa de salida, donde g es la función de activación 5: Paso backward. Cómputo de los gradientes locales (δ) de la red $\delta_1^3 = g'(h_1^3)[\zeta_1^\mu - V_1^3]$ (capa de salida) $\delta_i^m = g'(h_i^m) \cdot w_{1i}^{m,3} \delta_1^3$, donde $m \in \{1, 2\}$ (capas ocultas) 6: Actualización de los pesos y umbrales. $\Delta w_{1j}^{n,3} \leftarrow \eta \cdot \delta_1^3 \cdot V_j^n$, donde $n \in \{1, 2\}$ $\Delta w_{ij}^{0,m} \leftarrow \eta \cdot \delta_i^m \cdot V_j^0$, donde $m \in \{1, 2\}$ $w_{1j}^{n,3} \leftarrow w_{1j}^{n,3} + \Delta w_{1j}^{n,3}$, donde $n \in \{1, 2\}$ $promedio_j^m \leftarrow \frac{\sum_{i=1}^{39} \Delta w_{ij}^{0,m}}{39}$, $j \in \{0, 32\}$, $m \in \{1, 2\}$ $w_{ij}^{0,m} \leftarrow w_{ij}^{0,m} + promedio_j^m$, donde $m \in \{1, 2\}$, $i \in \{1, 39\}$, $j \in \{0, 32\}$ {El cálculo del promedio permite mantener la igualdad entre los <i>pesos correspondientes</i>.} 7: fin mientras 8: hasta satisfacción del criterio de corte |
|---|

Figura 3.5: Algoritmo de aprendizaje *time delayed back-propagation*.

Inicialización de los pesos y umbrales. Dependiendo de los valores con los que se inicializan los pesos se puede llegar a distintas soluciones: si los pesos iniciales son muy grandes probablemente las neuronas de la red se saturan, lo que hace que los gradientes del algoritmo de *back-propagation* sean pequeños y el algoritmo sea lento. Con el objetivo de determinar la mejor inicialización de pesos de la red se realizaron pruebas con distintos valores. En la Sección 3.3 se presentarán y analizarán los resultados obtenidos con cada una de ellas.

Orden de presentación de los patrones. Todos los patrones se presentan una vez en cada ciclo. A partir de las pruebas realizadas vimos que es conveniente elegir los patrones para la presentación a la red en orden aleatorio, distinto en cada ciclo. De esta forma llegamos a resultados con menos error y en una menor cantidad de ciclos.

Velocidad de aprendizaje. La velocidad de aprendizaje η , especifica el tamaño del paso en el descenso por gradiente. Cuanto más chico es η , más pequeños son los cambios en los pesos sinápticos de la red entre dos iteraciones y más suave es la trayectoria en el espacio de pesos. El proceso de aprendizaje también será más lento, requiriéndose una mayor cantidad de iteraciones. Si, en cambio, el parámetro η es muy grande los cambios en los pesos sinápticos tienen tal forma que la red se hace inestable, debido a que se producen oscilaciones. En la Sección 3.3 se mostrarán resultados con distintas elecciones de η .

Criterio de corte. Para determinar un criterio de corte utilizamos un tercer conjunto, denominado conjunto de validación. Para el armado de este conjunto se utilizó un 20% del conjunto de entrenamiento (el conjunto de validación no se utilizó para entrenar). Graficamos el error cuadrático medio, definido en la Ecuación 2.5, de los conjuntos de entrenamiento y de validación en función de las épocas. A medida que avanza el entrenamiento, la función de error del entrenamiento y de la validación decrecen. En algún momento este último comienza a crecer y es aquí donde detuvimos el entrenamiento. Para asegurarnos que el mínimo al que estábamos llegando no era un mínimo local, del cual saldría luego para llegar a otro mínimo con menor valor que el primero en todos los casos se estudió que cuando empezaba a subir el error ya no bajaba. Se puede visualizar lo explicado en la Figura 3.6. La utilización del conjunto de validación como criterio para determinar cuándo detener el entrenamiento es una heurística útil para evitar el problema de **overtraining u overfitting**, explicado en el Capítulo 2.

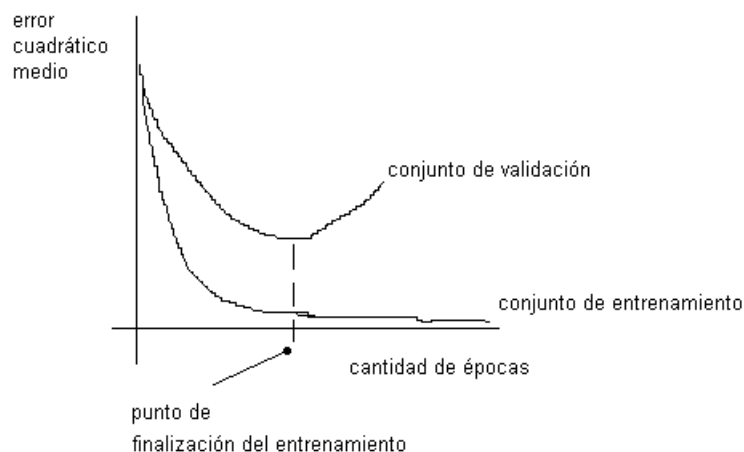


Figura 3.6: Gráfico del error cuadrático medio con los conjuntos de entrenamiento y validación. Está marcado el momento en que se detuvo el entrenamiento.

Dado que finalmente la inicialización elegida para los pesos es aleatoria (esto está explicado más adelante) y que para el entrenamiento de la red se presentan los patrones en orden aleatorio, los resultados de distintos entrenamientos con las mismas condiciones (mismos conjuntos de entrenamiento y test y mismos parámetros) no son iguales. Por esta razón se hicieron 5 ejecuciones considerando un mismo conjunto de validación, y se hizo un promedio de las épocas en que se terminó el entrenamiento. Una vez hecho esto, los datos del conjunto de validación se agregaron al conjunto de entrenamiento y se hizo una última ejecución –con el nuevo conjunto de entrenamiento– con tantas épocas como fueron arrojadas por el promedio. De esta forma evitamos dejar de utilizar para el entrenamiento los datos utilizados para la validación [84].

Conjuntos de Entrenamiento y de Test

Explicamos aquí las consideraciones tomadas para el armado de los conjuntos de entrenamiento y test.

Datos de promotores y de no promotores. Para el *tuning* de la red se utilizaron los datos de la compilación de Harley y Reynolds. Estos datos cuentan con la ventaja de tener marcadas las regiones -10 y -35 , lo que nos servirá para –más allá de la predicción acerca de si una secuencia es o no promotora– verificar si hay algún método de detección de estas subsecuencias (este tema será tratado en mayor profundidad en el Capítulo 5). Además, estos son los datos que han sido más utilizados hasta ahora [43, 55, 33]. En la Tablas B.2 y B.3 del Apéndice B pueden verse los promotores utilizados para los conjuntos de entrenamiento y de test.

En un caso de clasificación de datos en dos clases (como en el nuestro) necesitamos contar en el conjunto de entrenamiento con datos que pertenezcan a ambas. En nuestro caso esto se traduce a tener datos de promotores y de no promotores. Ya explicamos qué datos de promotores utilizamos, sin embargo no está claro que es un *no promotor*. Tampoco podemos estar seguros de que las secuencias de ADN que no fueron detectadas como promotores en experimentos realizados en los que se buscaban promotores sean efectivamente *no promotores*, dado que éstos pueden existir pero no haberse detectado en las circunstancias en las que se realizó el experimento. Lo que hacemos, entonces, es utilizar secuencias generadas aleatoriamente con la misma probabilidad de aparición de los nucleótidos A, C, G y T como no promotores. Si bien no tenemos la certeza de que así sea, consideramos a estas secuencias como *no promotores* y les damos éste nombre o bien *secuencias random o aleatorias*.

En el entrenamiento de redes neuronales para el reconocimiento de promotores procariontas la mayor parte de los trabajos utilizan el mismo criterio para la elección

de no promotores [52, 43, 16], algunos [54, 45] utilizan secuencias generadas aleatoriamente con 60% de probabilidad de aparición de As y Ts. Otros [55] utilizan un preprocesamiento para eliminar secuencias parecidas a promotores a partir de otros programas de búsqueda de promotores. En nuestro caso preferimos no realizar este procedimiento, dado que no conocemos la estructura de los promotores y creemos que podríamos estar introduciendo errores en el método.

Relación positivos:negativos. En la resolución de problemas de clasificación mediante redes neuronales no se conoce *a priori* la relación positivos:negativos óptima para un problema particular. En general esta relación se determina haciendo pruebas [1]. Para encontrar la mejor proporción realizamos pruebas con redes entrenadas con distintas proporciones de datos de promotores versus no promotores, en la Sección 3.3 mostraremos y analizaremos los resultados obtenidos. Se utilizó un conjunto de test con una relación promotor:no promotor de 1:25.

Abdelwahab et al. [1] y Demeler y Zhou [16] muestran los resultados experimentales al usar conjuntos de datos de distintos tamaños con distintas proporciones positivos:negativos en el entrenamiento de una red neuronal *feedforward* con *back-propagation*. En otros trabajos las relaciones positivo:negativo del conjunto de test utilizadas fueron 1:1.3 [52], 1:10 [43], 1:25 [63], 1:40 [45] y 1:50 [16].

3.3. Tuning

Con el objetivo de lograr una red con buenos resultados presentamos y analizamos en esta sección las pruebas realizadas para optimizar los parámetros de la red y los conjuntos de datos a utilizar por la misma para el aprendizaje y la generalización.

El procedimiento seguido para la elección de parámetros durante el entrenamiento fue el siguiente: se realizaron cambios en un parámetro (dejándose todos los demás parámetros fijos), se compararon distintas alternativas de arquitectura en función del mejor SCC alcanzado, y se eligió la mejor. El mismo proceso se realizó para cada uno de los parámetros involucrados. De esta forma estamos asumiendo que todos los parámetros son independientes, hipótesis que no necesariamente es cierta, pero si muy útil para realizar las pruebas. De todas formas se hicieron pruebas para verificar que lo que se había elegido como mejor opción continuaba siendo mejor al variar otro parámetro. Por ejemplo, una vez determinada la mejor relación positivos:negativos para el conjunto de entrenamiento, se hicieron pruebas adicionales con distintos tamaños de los *receptive fields*, verificándose que la relación positivos:negativos elegida seguía siendo la mejor al hacer esta prueba.

En todos los casos para la comparación de resultados calculamos el SCC para

distintos umbrales en cada una de las redes y nos quedamos con la que tenía SCC más alto en los conjuntos de test y de entrenamiento.

Relación positivos:negativos. Se entrenaron cuatro redes con distintos conjuntos de entrenamiento, con relaciones positivos:negativos de 1:50, 1:25, 1:10 y 1:3. Todas se probaron con el mismo conjunto de test, con el objetivo de permitir una comparación objetiva entre los distintos conjuntos de entrenamiento.

En la Tabla 3.1 se muestran los coeficientes de correlación estandarizados (SCC) más altos obtenidos con los conjuntos de entrenamiento y test resultantes de probar la red entrenada con los distintos conjuntos de entrenamiento.

| Experimento | Relación positivos:negativos | SCC test | SCC entrenamiento |
|-------------|------------------------------|----------|-------------------|
| 1 | 1:50 | 0.82 | 0.85 |
| 2 | 1:25 | 0.90 | 0.91 |
| 3 | 1:10 | 0.91 | 0.93 |
| 4 | 1:3 | 0.89 | 0.93 |

Tabla 3.1: SCCs de los conjuntos de entrenamiento y test para distintas relaciones positivos:negativos utilizadas en el conjunto de entrenamiento.

Como puede verse en la Tabla 3.1, el SCC más alto para el conjunto de test se obtuvo con la relación 1:10 y por esto ésta fue la relación seleccionada como óptima. El resultado con el conjunto de entrenamiento también es más alto para esta relación.

Otra conclusión a la que llegamos es la siguiente: al utilizar más negativos que positivos en el conjunto de entrenamiento, los primeros influyen más que los segundos en la minimización del error cuadrático medio, ya que en cada iteración se presentan más datos negativos que positivos. Como puede verse en la Figura 3.7, cuanto más negativos hay en los conjuntos de entrenamiento y de test, más grande es el error cuadrático medio de los positivos y menor el de los negativos. Al tener una relación positivos:negativos más pareja esta relación mejora.

Velocidad de aprendizaje. Se probó la red con los siguientes valores para el parámetro η : 2.0, 1.0, 0.4, 0.2, 0.1 y 0.05. Los mejores resultados se obtuvieron con $\eta = 0.2$, con el que se llegó a un menor error que con los otros y el promedio de épocas necesarias para el entrenamiento fue de 47. En la Figura 3.8 pueden verse las diferencias en los entrenamientos con distintos valores de η .

Resultado de inicialización de pesos. Se realizaron las siguientes pruebas: 1) inicialización con valores aleatorios en el intervalo $[-0.1,0.1]$ en los pesos entre todas

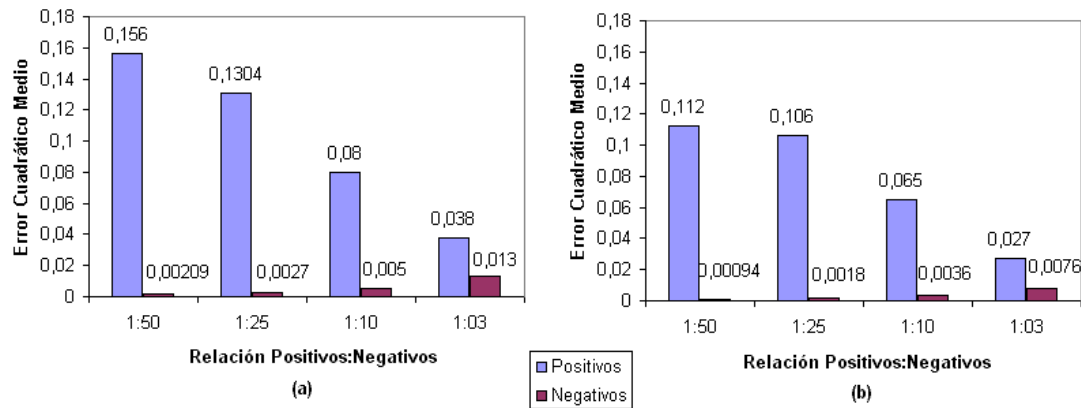


Figura 3.7: TDNN - Error cuadrático medio para datos positivos y negativos en distintas relaciones positivos:negativos. (a) Resultados con el conjunto de test, (b) Resultados con el conjunto de entrenamiento.

las capas (1-2, 1-3, 2-4, 3-4)⁵ 2) inicialización con valores aleatorios en el intervalo $[-0.01, 0.01]$ en los pesos entre todas las capas, 3) inicialización con valores aleatorios en el intervalo $[-0.1, 0.1]$ entre las capas 1-2 y 1-3 y aleatorios en el intervalo $[-0.01, 0.01]$ en las capas restantes y 4) inicialización con pesos resultantes de haber entrenado dos perceptrones para el reconocimiento de los consensos TATAAT y TTGACA entre la capa de entrada y las capas intermedias y a) valores en el intervalo $[-0.1, 0.1]$ y b) $[-0.01, 0.01]$ entre las capas intermedias y la de salida.

Con el objetivo de realizar la cuarta prueba, entrenamos dos perceptrones simples para el reconocimiento de los consensos TATAAT y TTGACA. El objetivo de esta prueba fue utilizar los pesos resultantes del entrenamiento de cada una de las redes para la inicialización de los pesos entre la entrada y las capas intermedias de la red neuronal TD. Como datos de promotores para el entrenamiento se utilizaron los hexámeros alineados por Harley y Reynolds [25] para las regiones -10 y -35 . Los resultados del entrenamiento del perceptrón para el reconocimiento del patrón TATAAT fueron buenos en la memorización y con el conjunto de test seleccionado (los SCCs fueron altos). Como se puede observar en la Figura 3.9, los pesos resultantes del entrenamiento de la red coinciden con el consenso TATAAT conocido. Los resultados del entrenamiento del perceptrón para el reconocimiento del patrón TTGACA no fueron buenos.

⁵Denominamos 1 a la capa de entrada, 2 y 3 a las capas ocultas y 4 a la capa de salida.

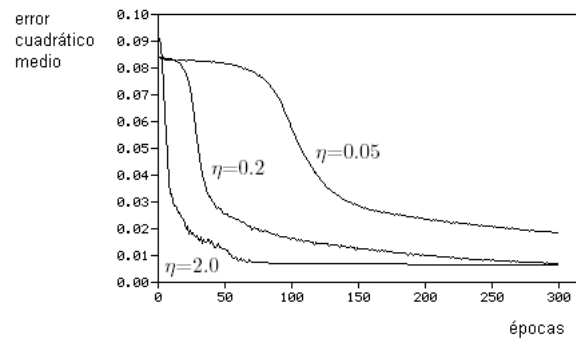


Figura 3.8: Error cuadrático medio en función de las épocas con el conjunto de entrenamiento para tres valores distintos de η . Con $\eta=2.0$, el error desciende rápidamente y tiene algunas oscilaciones, con $\eta=0.2$, el error desciende más lentamente y tiene menos oscilaciones y con $\eta=0.05$ el descenso es mucho más suave.

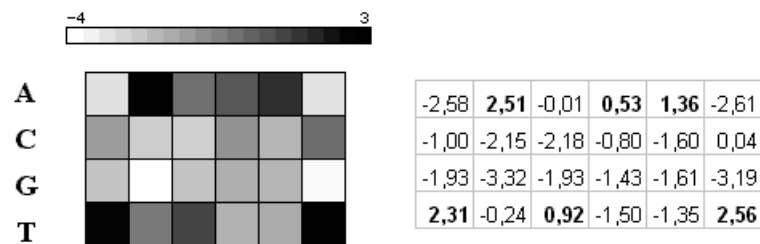


Figura 3.9: Diagrama de los pesos resultantes del mejor entrenamiento del perceptrón simple para el reconocimiento de la región -10 . El color de cada cuadrado difiere de acuerdo al peso que representa (el color negro representa al peso con valor más alto y color blanco el peso con valor más bajo). A la derecha se muestran los colores los valores de los pesos y a la izquierda los colores correspondientes.

Al igual que en los otros casos, en la comparación de las inicializaciones definimos que la mejor era aquella con la que se obtuvo un SCC más alto con los conjuntos de test y entrenamiento. Como puede verse en la Tabla 3.2, los mejores resultados fueron los obtenidos con la segunda prueba, seguidos por los de la tercer prueba. Los resultados de las distintas pruebas son, no obstante, bastante parecidos. En la Figura 3.10 se pueden ver los resultados para distintos umbrales (no sólo para el SCC más alto) y se puede comprobar que, efectivamente, los resultados de la segunda prueba son los mejores.

| Experimento | SCC test | SCC entrenamiento |
|-------------|----------|-------------------|
| 1 | 0.88 | 0.89 |
| 2 | 0.91 | 0.91 |
| 3 | 0.90 | 0.90 |
| 4 | 0.90 | 0.89 |

Tabla 3.2: SCCs más altos, resultantes de entrenar redes inicializadas con distintos valores en las distintas capas. Se muestran los resultados de la inicialización con 1) valores aleatorios en el intervalo $[-0.1,0.1]$ en todas las capas y 2) con valores aleatorios en el intervalo $[-0.01,0.01]$ en todas las capas, 3) valores aleatorios en el intervalo $[-0.1,0.1]$ entre la entrada y las capas intermedias y en el rango $[-0.01,0.01]$ entre éstas y la capa de salida, 4) resultados del entrenamiento de perceptrones TATAAT y TTGACA entre la entrada y la capa intermedia y valores pertenecientes al rango $[-0.01,0.01]$ entre ésta y la capa de salida.

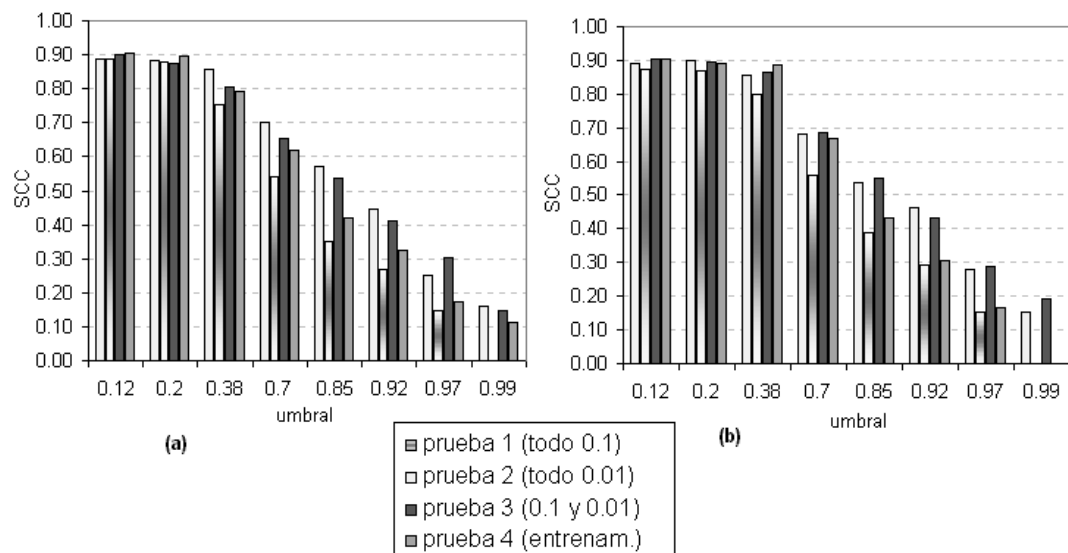


Figura 3.10: SCCs para distintos umbrales, resultantes de entrenar redes inicializadas con distintos valores en las distintas capas. Se muestran los resultados de la inicialización con 1) valores aleatorios en el intervalo $[-0.1,0.1]$ en todas las capas y 2) con valores aleatorios en el intervalo $[-0.01,0.01]$, 3) valores aleatorios en el intervalo $[-0.1,0.1]$ entre la entrada y las capas intermedias y en el rango $[-0.01,0.01]$ entre éstas y la capa de salida y 4) resultados del entrenamiento de perceptrones TATAAT y TTGACA entre la entrada y la capa intermedia y valores pertenecientes al rango $[-0.01,0.01]$ entre ésta y la capa de salida. (a) resultados para el conjunto de test, (b) resultados para el conjunto de entrenamiento.

Es interesante considerar que las inicializaciones que mejor resultaron coinciden con el comentario de Hertz et al. [31] que proponen elegir pesos aleatorios en el orden de $1/\sqrt{k_i}$, donde k_i es el número de entradas que alimentan a la neurona i . En nuestro caso tenemos $1/32 = 0,03125$ para los pesos que llegan a las capas intermedias y $1/39 = 0,0256$ para los pesos que llegan a la unidad de salida, ambas son del orden de 0.01.

Interpretación de pesos resultantes del entrenamiento. Generalmente las redes neuronales son consideradas *cajas negras*. Esto quiere decir que la representación interna del conocimiento de la red no tiene una interpretación clara. Es por esto que el estudio de los pesos resultantes del entrenamiento de la red no necesariamente brinda alguna información. Sin embargo, existen muchos trabajos que intentan extraer reglas a partir de los pesos [5]. A partir de la observación de los pesos resultantes del entrenamiento de la red encontramos algunas características interesantes. Como se puede ver en la Figura 3.11, los pesos de la segunda capa “tienen el consenso TGGTATAA”, i.e. en la posición que corresponde al primer nucleótido, el peso correspondiente a la entrada T es mayor al correspondiente a las entradas A, C y G, en la segunda posición el peso mayor es el correspondiente a la entrada G, y así siguiendo. Este consenso coincide en las últimas 5 letras con 5 de los 6 nucleótidos del consenso TATAAT y antes de éste contiene los nucleótidos TGG. Es interesante notar que en la compilación de Harley y Reynolds también aparece el consenso GG precediendo el consenso TATAAT, con una conservación menor que las restantes. Los pesos correspondientes a la primera capa no tienen un parecido con la región -35.

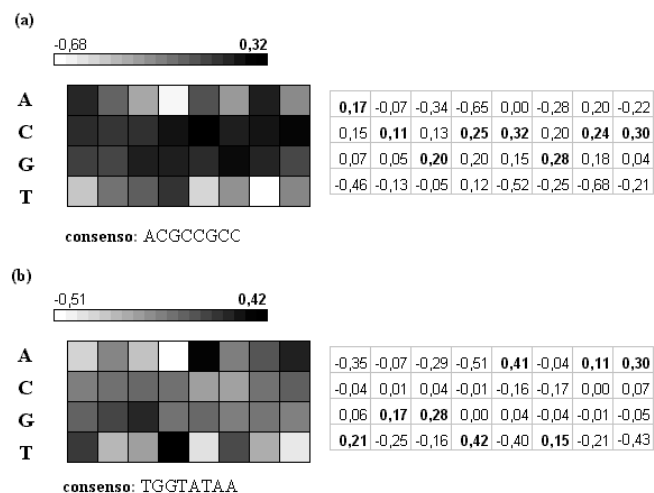


Figura 3.11: Pesos resultantes del entrenamiento de la red - Reconocimiento de subsecuencias conservadas. (a) pesos entre la entrada y la primera capa oculta, (b) pesos entre la entrada y la segunda capa oculta.

Varios de los valores más altos de la primer capa oculta están a una distancia de entre 15 y 21 pb de los valores más altos de la segunda capa oculta (ver Figura 3.12). Además estos valores son positivos. Si se considera que las neuronas de la segunda capa tienen la mayor activación con secuencias parecidas a TATAAT, se puede ver que en algunos casos los pesos resultantes del entrenamiento de la red están representando la estructura conocida del promotor.

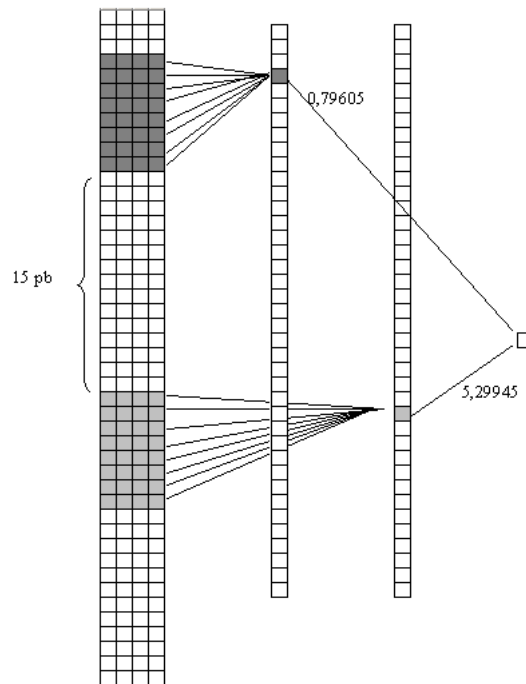


Figura 3.12: Pesos resultantes del entrenamiento de la red neuronal - Reconocimiento de distancias. Se muestran dos de los pesos más grandes de las capas ocultas.

A partir de los resultados obtenidos en las pruebas realizadas llegamos a las siguientes conclusiones: la diferencia entre los resultados con distintos tamaños de *receptive fields* no resultó significativa. Sin embargo permitió la detección de la extensión del patrón TATAAT en tres pares de bases; la variación en la proporción de positivos:negativos utilizada en el entrenamiento modifica los resultados, por esto es importante tenerla en cuenta. En nuestro caso la mejor relación resultó ser 1:10; finalmente, los resultados con distintas inicializaciones de pesos también difirieron, por lo que concluimos que es conveniente estudiar con qué valores se inicializa la red.

En este capítulo hemos propuesto el método *CPR* para el reconocimiento de promotores procariotas, y explicado porque es adecuada para la resolución del problema planteado. Analizamos el armado de la arquitectura de la red y los pasos importantes

a tener en cuenta en la construcción de la misma. Estudiamos los datos a utilizar y la partición en conjuntos de los mismos, y las mejores condiciones para solucionar el problema de descubrimiento de promotores. Hicimos las pruebas y adoptamos la mejor configuración. El método CPR es *modular*, dado que cuenta con dos capas ocultas, cada una encargada de reconocer distintas características.

Capítulo 4

Comparación del CPR con otros métodos computacionales para el reconocimiento de promotores

En el capítulo anterior construimos una TDNN para resolver el problema de reconocimiento de promotores con los datos de la compilación de Harley y Reynolds [25]. Actualmente hay nuevas compilaciones de datos disponibles que son más completas, correctas (los datos han sido corregidos) y precisas (sólo son datos de promotores de *E. coli*, no se incluyen datos de bacteriófagos, plásmidos ni promotores mutados).

Con el objetivo de hacer uso de las ventajas que proveen estos datos, reentrenamos la red obtenida en el Capítulo 3 con los mismos. En este capítulo mostramos los resultados obtenidos con el reentrenamiento y hacemos una comparación de éstos con los resultados obtenidos a partir del uso de los métodos fijos y estadísticos, descritos en el Capítulo 2, para la resolución de este problema. Para realizar la comparación utilizaremos en todos los métodos los mismos conjuntos de datos.

En la Sección 4.1 presentamos el problema, explicamos las ventajas de los nuevos datos disponibles y la forma en que haremos la comparación entre los distintos métodos. En las Secciones 4.2, 4.3 y 4.4 se explican las pruebas realizadas con los métodos fijos, Consensus y Patser y la red neuronal. En todos los casos se describen los parámetros elegidos, se explica la experimentación realizada, se presentan los resultados y se realiza un análisis de los mismos. Finalmente, en la Sección 4.5 se muestran tablas comparativas de los resultados de la aplicación de los distintos métodos a un mismo conjunto de test y se los analiza.

4.1. Problema

Para el entrenamiento de la red, explicado en el capítulo anterior, se utilizaron los datos de la compilación de Harley y Reynolds [25]. Estos han sido muy usados y resultan especialmente útiles porque tienen las regiones -35 y -10 marcadas. Sin embargo, en la actualidad se cuenta con los datos de la base de datos RegulonDB [68], que son más completos, correctos y específicos. Con el objetivo de hacer uso de estos datos, reentrenamos la red con los mismos y éstos son los que utilizaremos también para realizar las comparaciones entre los distintos métodos. Los datos de RegulonDB tienen varias ventajas respecto a los pertenecientes a compilaciones anteriores [26, 25, 42] analizadas en la Sección 1.4. Por un lado, ésta es la compilación más reciente y, como incluye a las compilaciones anteriores, es la más completa por el momento. Dado que está basada en la compilación de Lissner y Margalit [42], que hicieron una revisión y corrección exhaustiva de los datos publicados, los datos son correctos (tienen menos errores que los de compilaciones anteriores). Por otro lado, como explicamos en el Capítulo 1, los bacteriófagos tienen promotores fuertes y pueden no ser representativos de un promotor típico de *E. coli*. Por esto se puede considerar que los datos de RegulonDB no tienen el sesgo que tienen los de Harley y Reynolds [25], ya que su compilación no incluye promotores de bacteriófagos. Tampoco incluye datos de plásmidos. Además, los promotores de RegulonDB pertenecen a una misma cepa (K12), mientras que en la compilación de Harley y Reynolds no hay información al respecto. Es útil conocer esta característica y, en caso de estar utilizándose para el entrenamiento datos de más de una cepa, saber cómo difieren éstos entre sí. De esta forma es posible distribuirlos correctamente en los conjuntos de entrenamiento y test. Finalmente, la compilación de Harley y Reynolds incluye mutaciones. Creemos que la utilización de promotores con mutaciones puntuales para el entrenamiento de la red neuronal puede sesgar la muestra si no se tienen datos de mutaciones para todos los promotores.

Los promotores encontrados experimentalmente informados en RegulonDB tienen una longitud de 81 pb. En nuestra solución consideramos 46 pb. En esta longitud están incluidas ambas secuencias conservadas, la distancia entre éstas y el TSS [26, 25, 42]. Al contar los datos de RegulonDB con 81 pb (se muestran 60 pb aguas abajo del TSS encontrado experimentalmente y 20 pb aguas arriba del mismo, ie. el TSS está en la posición 61) surge el problema de seleccionar cuál de las 36 subsecuencias de longitud 46 considerar. Optamos por la subsecuencia que tiene al TSS en la posición 41 (ver Figura 4.1).

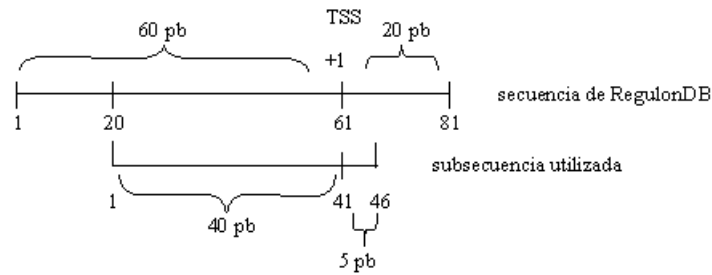


Figura 4.1: Subsecuencia de RegulonDB utilizada

Dos hechos nos permiten comprobar que ésta es una buena elección:

1. Revisando visualmente los datos de RegulonDB se ve que las secuencias con parecidos a TATAAT y TTGACA están contenidas entre los nucleótidos que se encuentran 40 pb aguas abajo del TSS.
2. Se testeó la red entrenada de acuerdo a los procedimientos descritos en el Capítulo 3 con los conjuntos de datos provenientes de cada uno de los 36 conjuntos de subsecuencias de RegulonDB. El mejor resultado se dio para las subsecuencias que se encuentran entre las posiciones 20 y 65 de las secuencias de RegulonDB (ver Figura 4.2 y Tabla 4.1). En la Tabla 4.1 se muestran (ordenadas descendientemente por promedio) las seis posiciones de inicio de los conjuntos de secuencias cuyo promedio dio más alto. Estas son contiguas (18–23). Observando el gráfico y la tabla se infiere que los mejores resultados se dan para las subsecuencias que comienzan en las posiciones 20, 19 y 21. Dado que el TSS está en la posición 61, estas subsecuencias se encuentran 41, 42 y 40 pb aguas abajo del TSS respectivamente. En *desde* y *hasta* se muestran las posiciones desde y hasta de la secuencia en que se encuentra la subsecuencia en cuestión y en *cantidad nucleótidos después del TSS* se muestra hasta cuántos nucleótidos después del TSS se está tomando en la subsecuencia en cuestión (por ej. la primer subsecuencia de la tabla empieza en la posición 20 y –dado que es de longitud 46– termina en la posición 65, además como el TSS se encuentra en la posición 61 de las cadenas de longitud 81 de RegulonDB, hay 4 pb desde el TSS hasta el final de la subsecuencia, $61 + 4 = 65$).

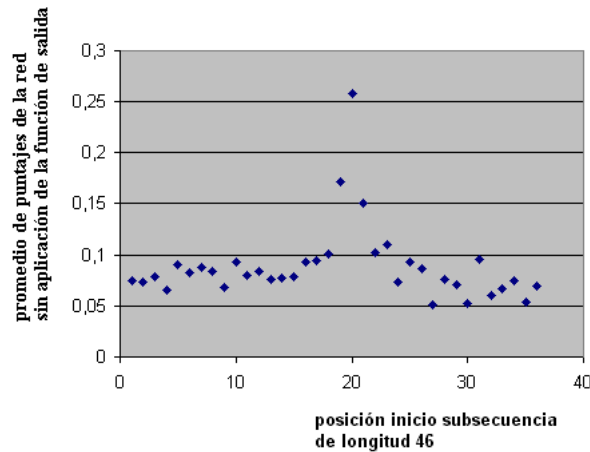


Figura 4.2: Resultados para distintas subsecuencias. Se muestra para cada una de las 36 subsecuencias de longitud 46 de los promotores reconocidos por la subunidad σ^{70} de RegulonDB el promedio del resultado arrojado por la salida de la red sin la aplicación de la función de salida (ver Fórmula 3.1).

| Desde | Hasta | Promedio | Desviación estándar | Cantidad nucleótidos después del TSS |
|-------|-------|----------|---------------------|--------------------------------------|
| 20 | 65 | 0.259 | 0.354 | 4 |
| 19 | 64 | 0.171 | 0.293 | 3 |
| 21 | 66 | 0.150 | 0.272 | 5 |
| 23 | 68 | 0.109 | 0.226 | 7 |
| 22 | 67 | 0.102 | 0.214 | 6 |
| 18 | 63 | 0.101 | 0.217 | 2 |

Tabla 4.1: Resultados de las seis subsecuencias de longitud 46 de RegulonDB en que el promedio de la red neuronal dio más alto.

Los distintos métodos tienen que ser analizados con los mismos conjuntos de datos. Para el reentrenamiento de la red precisamos conjuntos de entrenamiento y de test. Ambos tienen que contener datos de promotores y de no promotores. Consensus genera matrices de alineamiento a partir de un conjunto de secuencias y mediante una transformación de estas, Patser les asigna un puntaje a las palabras pertenecientes a secuencias de test. Para generar las matrices sólo se utilizan datos de promotores, dado que lo que se pretende es que éstas formen un modelo de un promotor. Finalmente, los métodos fijos buscan patrones definidos en un conjunto de secuencias. En este

caso no hay una instancia de “entrenamiento”. Con el objetivo de poder comparar los distintos métodos armamos dos conjuntos de datos: el de entrenamiento y el de test. Ambos cuentan con datos de promotores y de no promotores. Para el entrenamiento de la red neuronal utilizamos el conjunto de entrenamiento y para la generación de las matrices con Consensus se utilizará el mismo conjunto pero sólo con los datos de promotores. Los tres métodos son testeados con el conjunto de test. A continuación explicamos cómo fueron armados ambos conjuntos.

Datos de promotores y de no promotores. Se utilizaron datos de promotores reconocidos por subunidades σ^{70} de la ARN polimerasa. Esto incluye los datos de la tabla de promotores de RegulonDB versión 3.1 cuyo valor en el campo *sigma_factor* pertenece al conjunto {"Sigma70", "Sigma32, Sigma70", "Sigma70, Sigma38"}¹ [67, 68]. La tabla de promotores tiene 739 secuencias de promotores, al eliminarse todas aquellas no reconocidas por subunidades σ^{70} de la ARN polimerasa y aquellas que no tienen la secuencia del promotor informada, quedaron 580 secuencias. Al igual que en el entrenamiento de la red neuronal, en este caso también utilizamos secuencias generadas aleatoriamente con equiprobabilidad de aparición de los nucleótidos A,C,G y T como no promotores.

La **relación promotor:no promotor** para el conjunto de entrenamiento fue 1:10. Esta fue evaluada como la mejor relación posible para la red neuronal, cuyo entrenamiento y *tuning* fueron descriptos en el Capítulo 3 y también fue probada para esta red obteniéndose los mejores resultados con esta relación. Para el conjunto de test la relación promotor:no promotor utilizada fue de 1:25

El 70 % de los promotores se utilizó para armar el conjunto de entrenamiento y el 30 % restante para el de test. En general se toma una muestra mayor para el conjunto de entrenamiento que para el de test [83]. Los promotores fueron distribuidos aleatoriamente en ambos conjuntos con el objetivo de que pueda haber promotores de un mismo gen y promotores fuertes y débiles en ambos conjuntos. Como validación utilizamos un 20 % del conjunto de entrenamiento. En la Tabla B.4 del Apéndice B pueden verse los promotores utilizados para los conjuntos de entrenamiento y de test. Se muestra el identificador de la base de RegulonDB y el nombre del promotor.

En los trabajos previos para el reconocimiento de promotores procarionas con redes neuronales la partición de promotores en conjuntos de entrenamiento y test varía mucho: Nakata et al. [52] utilizan 63.33 % para entrenamiento y el resto para test. Lukashin et al. [43] utilizan 11 % para entrenamiento y 100 % para test (incluyen en el conjunto de test los promotores utilizados para el entrenamiento). Demeler y Zhou utilizan aproximadamente 73 % para entrenamiento y lo restante para el test. Finalmente, Mahadelian y Ghosh [45] utilizan 46 % para entrenamiento y 54 % para

¹Estos promotores son reconocidos por más de un factor σ .

test. Por otro lado, no todos los conjuntos de test son independientes de los de entrenamiento [43], ni los datos están distribuidos aleatoriamente entre los dos conjuntos como en nuestro caso [16, 45].

Finalmente, el conjunto de entrenamiento tiene 406 secuencias de promotores y 4060 secuencias generadas aleatoriamente como no promotores y el conjunto de test tiene 174 secuencias de promotores y 4350 de no promotores. No analizamos las secuencias reversas complementarias, sino solamente las secuencias en la dirección ingresada.

Evaluación de resultados

Para asegurar la validez de los resultados obtenidos a partir de los métodos computacionales de predicción de promotores es necesaria la realización de verificaciones experimentales en laboratorio. Dado que los experimentos son muy costosos, el investigador puede preferir obtener pocos resultados FP, a pesar de obtener de esa forma menos TP. Otra visión puede consistir en procurar conocer todos los resultados TP, de forma tal de poder realizar un análisis exhaustivo, sin darle importancia a que muchos de los resultados obtenidos mediante el método computacional sean FP. Ambos casos podrían verse como dos opiniones extremas, pero son útiles a efectos de ilustrar que la decisión del margen y tipo de error soportado la debería tener cada investigador. Esto quiere decir que los distintos errores (FP y FN) tienen distinto costo. Lo ideal es tener la posibilidad de evaluar distintos escenarios. Consensus y Patser, al igual que la red neuronal, arrojan como resultado un puntaje para cada secuencia. Para poder evaluar los resultados de estas métodos es necesario fijar un **umbral**, de forma tal que todas las secuencias cuyo puntaje lo supere sean consideradas promotoras y todas aquellas que no lo alcancen sean consideradas no promotoras.

En los resultados de los distintos métodos mostramos los coeficientes de correlación estandarizados (SCCs) que se obtienen para todos los umbrales evaluados, resaltando aquéllos que poseen los SCCs más altos y los que tienen tasas de FP menores o iguales al 1%. Utilizamos el SCC como criterio de evaluación en las comparaciones, dado que el método que llega a un mayor SCC es el que tiene predicciones más parecidas a la realidad. En los casos en que es posible, también mostramos los resultados de los métodos restringiendo los porcentajes de FP a valores bajos y los porcentajes de TP a valores altos.

Notación

El SCC, proviene de aplicar la Fórmula 2.21. S_n es la sensibilidad y S_p la especificidad (todas definidas en la Sección 2.5). Llamamos $\%TP$ al porcentaje de promotores correctamente predichos,

$$\%TP = \frac{TP \cdot 100}{TP + FN} \quad (4.1)$$

y $\%FP$ al porcentaje de no promotores incorrectamente predichos,

$$\%FP = \frac{FP \cdot 100}{TN + FP}. \quad (4.2)$$

Para la representación de cadenas de ADN utilizamos la siguiente notación:

- Se soportan los códigos de la IUPAC-IUB (*International Union of Pure and Applied Chemistry-International Union of Biochemistry*) para la ambigüedad de códigos del ADN, se puede ver la lista completa en la Tabla B.1 del Apéndice B. De acuerdo a este estándar, N representa cualquiera de los cuatro nucleótidos.
- $R\{j\}$ $j \geq 0$ representa j repeticiones del código R (R es uno de los códigos definidos por la IUPAC-IUB), por ejemplo $A\{6\}$ es equivalente a AAAAAA y $N\{2\}A = NNA$, que puede representar a: AAA, ACA, AGA, ATA y CCA entre otros.
- $R\{i,j\}$ $i, j \geq 0$ representa una cantidad variable de repeticiones entre i y j . Por ejemplo: $N\{0,60\}$ representa entre 0 y 60 nucleótidos.

A continuación explicaremos los experimentos realizados para cada uno de los métodos y presentaremos y analizaremos los resultados, para luego poder compararlos.

4.2. Métodos fijos

Como se explicó en la Sección 2.1, los métodos fijos desarrollados por Van Helden toman como entrada un patrón y un conjunto de secuencias y devuelven las secuencias en las que se encuentra el patrón. A cada uno de los nucleótidos que forma parte del patrón buscado se le da igual peso o significación y se permiten pocas sustituciones de nucleótidos con respecto al patrón buscado.

Recordemos que denominamos *sustitución* al cambio de un nucleótido por otro. Por ejemplo, si se busca el patrón TAT en la secuencia TAGAA sin permitirse ninguna sustitución no se lo encuentra. Si se permite una sustitución se lo encuentra en la

posición 1 (TAG), si se permiten 2 sustituciones se lo encuentra en la posición 1 (TAG), y en la posición 3 (GAA). Denominamos *delección* a la eliminación de un nucleótido en una secuencia e *inserción* al agregado de un nucleótido en una secuencia. En la implementación utilizada se permiten hasta dos sustituciones y también se permite la búsqueda de patrones que contengan espaciados de longitud variable, en cuyo caso el programa no permite sustituciones.

4.2.1. Experimentación

Utilizamos el programa *dna-pattern*, disponible en Regulatory Sequence Analysis Tools (RSA Tools)² [80].

Patrones de búsqueda

A partir de las conclusiones a las que se llegaron en los análisis de las compilaciones de datos de promotores se armaron los siguientes patrones de búsqueda:

- **TTGACAN{17}TATAAT:** respeta las secuencias consenso para las regiones -10 y -35 y el espaciado fijo entre ellas de 17 pb. Estos son los consensos y espaciado ideales de acuerdo a los análisis realizados por Harley y Reynolds, Lisser y Margalit y Hawley y McClure³.
- **TTGNNNN{17}TANNNT:** respeta los tres nucleótidos más conservados – de acuerdo a las compilaciones previamente mencionadas– de cada una de las regiones consenso, con espaciado fijo de 17 pb.
- **TTGNNNN{15,21}TANNNT:** el mismo caso que el anterior con un espaciado entre las dos regiones de entre 15 y 21 pares de bases (entre los que se encuentra más del 92 % de las regiones -10 y -35 de los promotores analizados por Harley y Reynolds)

Los dos primeros patrones fueron buscados sin permitirse sustituciones y con una y dos sustituciones. El último fue buscado sin sustituciones. Se puede notar que el segundo patrón de búsqueda es equivalente al primero con la admisión de una mayor cantidad de sustituciones. El tercer patrón de búsqueda es equivalente al segundo, con la admisión de hasta dos delecciones y hasta cuatro inserciones entre los patrones TTGNNN y TANNNT. Las secuencias en las que se buscan los patrones son las del conjunto de test.

²<http://rsat.ulb.ac.be/rsat/>, opción: dna-pattern.

³Recordemos que el grado de conservación de las bases a partir del estudio de Lisser y Margalit es el siguiente: $T_{69}T_{79}G_{61}A_{56}C_{54}A_{54} - 16_{17}17_{43}18_{17} - T_{77}A_{76}T_{60}A_{61}A_{56}T_{82}$. En N_x , x denota el porcentaje de apariciones del nucleótido N . En Y_x , x denota el porcentaje de apariciones de un espaciado de Y pares de bases entre las dos secuencias consenso.

Parámetros

No se permitieron *matchings* superpuestos, ya que no servían a nuestros objetivos (en esta prueba queremos saber cuántas secuencias son clasificadas correctamente, pero no en que posición de las mismas se encuentra el promotor).

4.2.2. Resultados

En la Tabla 4.2 se muestran los resultados de ejecutar los métodos fijos buscando los distintos patrones en el conjunto de test. Se pueden observar los porcentajes de falsos positivos, verdaderos positivos y el SCC para distinta cantidad de sustituciones.

| Patrón buscado | Cantidad sustituciones permitidas | % FP | % TP | SCC |
|-----------------------------|-----------------------------------|-------|-------|------|
| TTGACAN{17}TATAAT | 0 | 0 | 0 | – |
| | 1 | 0 | 0 | – |
| | 2 | 0.05 | 0.57 | 0.05 |
| TTGNNNN{17}TANNNT | 0 | 0.46 | 9.77 | 0.21 |
| | 1 | 7.59 | 35.63 | 0.34 |
| | 2 | 50.18 | 85.63 | 0.38 |
| TTGNNNN{15,21}TANNNT | 0 | 2.69 | 25.86 | 0.33 |

Tabla 4.2: Métodos fijos - Porcentajes de FP, TP y SCC en las pruebas realizadas con el conjunto de test.

Ninguna secuencia del conjunto de test contiene al patrón que las distintas compilaciones y análisis de datos asumen como el ideal (TTGACAN{17}TATAAT). Tampoco hay secuencias que contengan este patrón con una sustitución. Recién cuando se permiten dos sustituciones se encuentran algunas secuencias que lo contienen. Con el segundo patrón, que tiene menos restricciones, se obtienen mejores resultados. Finalmente, los resultados de las pruebas con patrones con espaciado variable llegan a un SCC parecido al de la prueba anterior con una sustitución, pero con menor cantidad de falsos positivos y de verdaderos positivos.

En esta tabla también podemos ver la importancia de considerar además del SCC los porcentajes de TP y de FP para el análisis de resultados. En este caso puede ser preferible la elección del patrón TTGNNNN{17}TANNNT con una sustitución que con dos sustituciones, ya que si bien en el segundo caso el SCC es más alto que en el primero, la cantidad de FP del primero es más baja (el costo de esta elección es que se tiene una menor cantidad de TP). Otro resultado que puede compararse realizando consideraciones parecidas es el del patrón TTGNNNN{15,21}TANNNT.

Finalmente, podemos observar que los SCCs resultantes son bajos. En el único caso en que hay una cantidad alta de verdaderos positivos (85.63%), la cantidad de falsos positivos también es muy alta (50.18%).

4.2.3. Análisis de resultados

El hecho de que ninguna secuencia del conjunto de test contenga el patrón TTGACAN{17}TATAAT, asumido como el ideal por las distintas compilaciones y análisis de datos, confirma el comentario que hicimos en el Capítulo 1, acerca de la posibilidad de que ninguna de las secuencias que originó la secuencia consenso tenga exactamente las mismas bases que ésta. A modo de ejemplo de lo que enunciamos, se puede ver que si buscamos el patrón TTGACAN{17}TATAAT permitiendo dos sustituciones, y existe una secuencia con patrón TTGgggN{17}TATAAT⁴ que es promotora, no la vamos a encontrar. Sin embargo, los nucleótidos *ACA* de la región -35 , cuya ausencia en esta cadena hace que no encontremos el patrón, tienen probabilidades de aparición de 0.56, 0.54 y 0.54 respectivamente [42], por lo cual la probabilidad de que una secuencia tenga los tres nucleótidos es $P(A) \times P(C) \times P(A)$ en las respectivas posiciones, que es igual a 0.16. Esta probabilidad es baja, y explica por qué es razonable que la búsqueda de un patrón que respete exactamente la secuencia consenso no tiene buenos resultados.

El hecho de darle más importancia a los nucleótidos con una conservación más alta y permitir sustituciones flexibiliza la búsqueda, y mejora los resultados. Como es de esperar, cuantas más sustituciones se permiten en los patrones de búsqueda se encuentran más resultados verdaderos positivos, pero esto incrementa también el número de falsos positivos.

Por otro lado, los resultados provenientes de la búsqueda del segundo patrón con dos sustituciones muestran que gran parte de los promotores reconocidos por la subunidad σ^{70} tienen al menos uno de los tres nucleótidos más altamente conservados de la región -35 , lo que es acorde a las conclusiones a las que arribaron Hawley y McClure y Harley y Reynolds (mencionadas en la Sección 1.4).

Las pruebas realizadas con el método de Van Helden arrojan los resultados que se podían suponer, debido a las limitaciones que tienen para la resolución de este problema: en este método los componentes del patrón (los nucleótidos) no son valorados con distintos pesos, es decir todos tienen la misma importancia. Esto dista de lo que sucede en la realidad, debido a que la afinidad de la polimerasa por el ADN no es uniforme [30]. Además, se realiza una búsqueda de la secuencia consenso. Estas secuencias son útiles para considerar qué nucleótidos tienen mayor frecuencia de aparición en cada

⁴Representamos con letras minúsculas a aquellos nucleótidos que no coinciden con el nucleótido definido por la secuencia consenso.

posición de un alineamiento, pero normalmente la probabilidad de aparición de una secuencia que contenga los mismos nucleótidos que el consenso en cada posición es muy baja, dado que proviene de multiplicar las frecuencias de aparición de cada uno de los nucleótidos en cada posición.

En conclusión, con esta herramienta, basada principalmente en la secuencia consenso armada, no se llega a resultados satisfactorios.

4.3. Método Consensus-Patser

Como se explicó en la Sección 2.2, Consensus busca alineamientos del ancho especificado como entrada. Como resultado de la búsqueda arroja las matrices de alineamiento de los mejores alineamientos obtenidos y sus secuencias consenso. Recordemos que estas matrices muestran la cantidad de apariciones de cada nucleótido en cada posición del alineamiento. Las matrices resultantes pueden ser *inales* o *intermedias* (de acuerdo a si consideran o no palabras de todas las secuencias). Para cada matriz se informa la frecuencia esperada, que permite la comparación de matrices de alineamiento de distintos anchos y generadas por distinta cantidad de secuencias. Cuanto menor frecuencia esperada tengan las matrices, más estadísticamente significativas son.

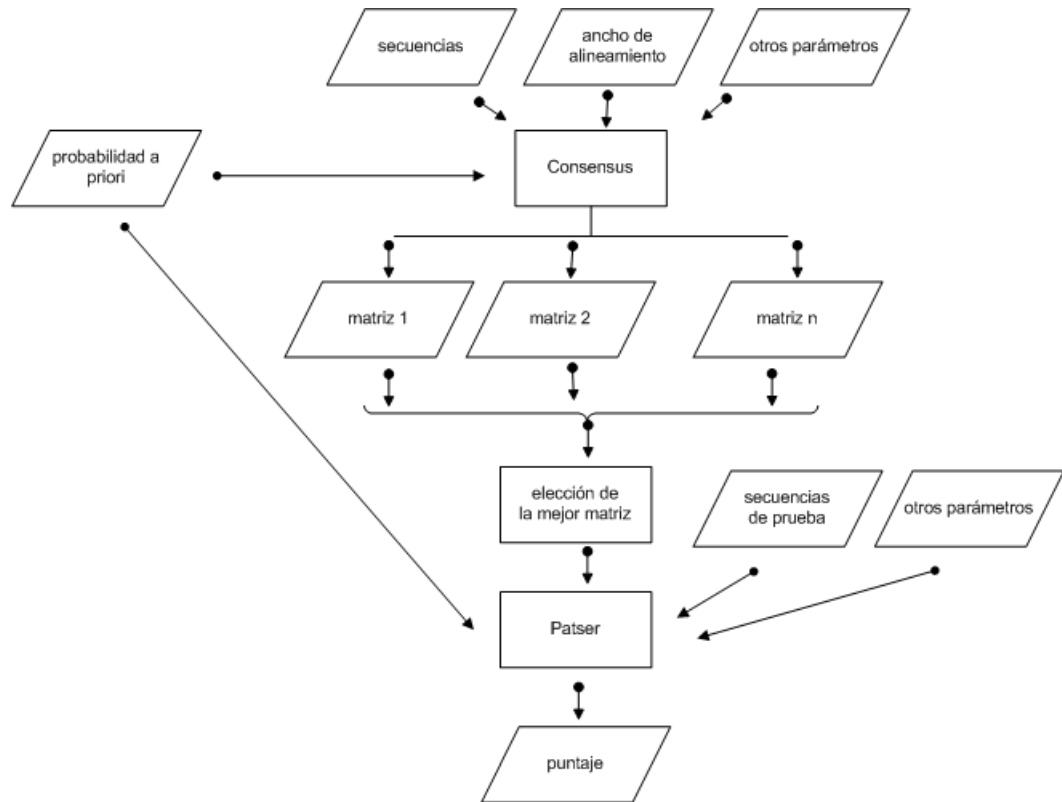
Patser toma como entrada una matriz de alineamiento y un conjunto de secuencias y a partir de la primera, les asigna un puntaje a todas las palabras (subsecuencias de longitud fija) de las secuencias indicadas. Para realizar la asignación de puntajes, Patser genera matrices de pesos (definidas en el Capítulo 1), que difieren para un mismo alineamiento si se considera distinta probabilidad de aparición de nucleótidos.

Denominamos *Método Consensus-Patser* a la aplicación de los siguientes pasos:

1. ejecución de Consensus para distintas condiciones de probabilidad *a priori*, anchos de alineamientos y longitud de secuencias a alinear,
2. selección de la mejor matriz obtenida con cada una de las condiciones,
3. ejecución de Patser para cada matriz seleccionada en el paso 2, y
4. elección del mejor resultado de Patser.

El esquema mostrado en la Figura 4.3 resume el método Consensus-Patser.

Recordemos que al utilizar Consensus queremos armar un modelo de los promotores y por esto no utilizamos datos de no promotores. Al utilizar Patser para evaluar las matrices generadas utilizamos datos de promotores y de no promotores, ya que queremos ver cuán bien se discrimina entre ambas secuencias.



Referencias

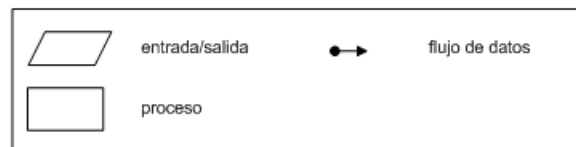


Figura 4.3: **Esquema del método Consensus-Patser (C-P)** - En este método elegimos la mejor matriz resultante de cada ejecución de Consensus (por simplicidad se muestra una sola ejecución de Consensus) y ejecutamos Patser con cada una de las elegidas. Ejecutamos Consensus y Patser con la misma probabilidad *a priori*. Elegimos los mejores resultados de Patser (dado que se muestra la ejecución de Patser para una única matriz, no se visualiza en el gráfico dicha elección).

La implementación utilizada de Consensus asume que la probabilidad de aparición *a priori* de cada nucleótido de las secuencias es independiente.

4.3.1. Experimentación

Notación. En las secuencias consenso se muestra para cada posición el nucleótido con mayor cantidad de apariciones en un alineamiento. La notación $[X.Y]$ en la posición z de la secuencia consenso de un alineamiento, donde $X, Y \in \{A, C, G, T\}$, indica que la cantidad de apariciones de X e Y en la posición z del alineamiento es exactamente igual. Por ejemplo, el consenso del alineamiento de la Figura 4.4 es A[C.G]G.

| | | |
|---|---|---|
| A | C | G |
| A | G | G |
| A | G | G |
| A | C | T |

Figura 4.4: Alineamiento.

Se consideraron dos posibles longitudes de secuencias y se realizaron dos búsquedas: 1) de palabras de ancho 46 en cadenas de longitud 46 y 2) de palabras de ancho variable entre 42 y 49 pb en secuencias de longitud 56. A efectos de realizar la segunda búsqueda se extendió la longitud de las secuencias tomadas de RegulonDB en 10 pb, 5 en cada uno de los extremos. Esto nos permite realizar una búsqueda más flexible.

Para cada una de las pruebas se realizaron ejecuciones considerando las siguientes probabilidades *a priori* de los nucleótidos:

- equiprobables (la probabilidad de aparición de cada uno es de 0.25),
- probabilidad de aparición de A y T 0.3 y de C y G 0.2. La frecuencia de cada base en el genoma de *E. coli* es esencialmente la misma, sin embargo las regiones promotoras son ricas en As y Ts, dado que el hecho de que el par AT esté unido por menos enlaces puente hidrógeno que el par CG, hace que los primeros se separen más fácilmente (ver Sección 1.1), y
- la frecuencia observada en las secuencias a alinear.

Se utilizó la versión 6c de Consensus⁵ desarrollada por Gary D. Stormo desde un script perl que corre repetidamente Consensus buscando alineamientos de distinto ancho.

⁵Consensus está disponible en la dirección <http://ural.wustl.edu/~jhc1/consensus/>.

Consensus

Parámetros. Para los demás parámetros (no referidos a la frecuencia de aparición de los nucleótidos) se respetan los valores por defecto: se ignora la secuencia complementaria, se devuelven las cinco mejores matrices intermedias y las cinco mejores del ciclo final, se almacenan como máximo 1000 matrices intermedias (i.e. los parámetros ci , cf y q del algoritmo mostrado en la Figura 2.2 toman los valores 5, 5 y 1000 respectivamente) y cada secuencia contribuye una y sólo una vez al alineamiento. La utilidad de estos parámetros fue descrita en la Sección 2.2.

En la Tabla 4.3 se muestra un resumen de la ejecuciones realizadas.

| Número de ejecución | Probabilidad <i>a priori</i> de nucleótidos | Ancho del alineamiento buscado | Longitud de las secuencias de entrada |
|---------------------|---|--------------------------------|---------------------------------------|
| 1 | ACGT = 0.25 | 46 | 46 |
| 2 | AT = 0.3, CG = 0.2 | | |
| 3 | observada | | |
| 4 | ACGT = 0.25 | entre 42 y 49 | 56 |
| 5 | AT = 0.3, CG = 0.2 | | |
| 6 | observada | | |

Tabla 4.3: Resumen de las ejecuciones realizadas con Consensus.

Criterios de selección de matrices. Entre todas las matrices obtenidas se seleccionó –para cada una de las ejecuciones– la que representa mejor la composición del promotor. Los criterios considerados para la elección de la mejor matriz de cada ejecución fueron los siguientes:

1. en todos los casos se prefirió el resultado final que los intermedios, ya que éste es formado a partir de todas las secuencias,
2. se prefirió la matriz con mejor representación del modelo conocido del promotor (conservación de las secuencias consenso y longitud entre éstas de 17 pb o entre 15 y 21 pb en su defecto),
3. menor frecuencia esperada de la matriz de alineamiento, y
4. menor desviación estándar de los puntajes resultantes de la aplicación de Patser con la matriz salida de Consensus al conjunto de entrenamiento, (en la ejecución de Patser se consideró la misma frecuencia que la utilizada en la ejecución de Consensus). Para el cálculo de la desviación estándar se consideraron todos los

puntajes (i.e. no se utilizó el umbral fijado por la aplicación). Nos interesa el resultado con menor desviación estándar, dado que éste es el alineamiento con distribución más compacta de puntajes.

Los criterios fueron evaluados en el orden expuesto.

Patser

Ejecutamos Patser para cada una de las matrices resultantes de la aplicación de Consensus, con el objetivo de quedarnos con la mejor matriz de las que preseleccionamos con los criterios anteriormente mencionados.

Para la **elección del umbral**, mencionada bajo el título *Evaluación de Resultados* de la Sección 4.1, se utilizó el SCC. Al igual que en la red neuronal, se generaron distintos valores pertenecientes al intervalo de resultados de los puntajes arrojados por Patser. Consideramos a estos valores como umbrales y para cada uno de ellos calculamos el SCC. Determinamos que la mejor matriz es aquella con la que se obtiene el SCC más alto para el conjunto de test.

Se ejecutó la versión 3d de Patser por línea de comandos desde el sistema operativo Linux⁶.

Parámetros. Patser permite utilizar como entrada una matriz de alineamiento (como fue mostrado en el Capítulo 2) o una matriz de pesos. Utilizamos la primer opción. Se utilizó la misma probabilidad *a priori* de nucleótidos que la utilizada para la ejecución de Consensus. De todas las posibles palabras de una secuencia, se devuelve aquella con mayor puntaje (ver parámetro *puntajes* en la Figura 1.18) y para todas éstas se imprimieron los puntajes, i.e. no se utilizó el umbral propuesto por la aplicación (ver parámetro *umbral* en la Figura 1.18), con el objetivo de poder elegirlo nosotros. Para los demás parámetros se utilizaron las opciones seteadas por defecto.

4.3.2. Resultados

Consensus

La ejecución de las seis pruebas planteadas en la Tabla 4.3, arrojó 45 matrices; 5 para cada una de las tres primeras y 10 para cada una de la tres últimas. A continuación explicamos cómo se seleccionó la mejor matriz de cada una de las ejecuciones a partir del procedimiento definido en la Subsección 4.3.1. Como anticipamos en dicha subsección, sólo analizamos los resultados de las matrices finales.

En la Tabla 4.4 se puede observar la matriz resultante y la secuencia consenso de la ejecución 1, en que se alinearon secuencias de longitud 46 con equiprobabilidad de

⁶Patser está también disponible en la dirección <http://ural.wustl.edu/jhc1/consensus/>.

aparición de las 4 bases. Recordemos que la matriz resultante de Consensus muestra para cada posición del alineamiento obtenido la cantidad de secuencias en que apareció cada uno de los nucleótidos. Por ejemplo, en la primer posición del alineamiento resultante de la ejecución número 1, que puede verse en la Tabla 4.4, hay 115 secuencias que tienen *A*, 70 secuencias que tienen *C*, 99 que tienen *G* y 122 que tienen *T*, esto suma 406, que es la cantidad de secuencias que se utilizaron como entrada. En la secuencia consenso que se muestra en esta tabla, se puede ver que se identificó la subsecuencia TAaAAT, muy parecida al consenso TATAAT. No se encontró ninguna subsecuencia parecida al consenso TTGACA.

Dado que en las primeras tres ejecuciones se hicieron alineamientos de ancho 46 en secuencias de longitud 46, hay un sólo alineamiento final posible, razón por la cual cada una de estas ejecuciones arroja como resultado una única matriz final. El único parámetro que difiere en estas tres ejecuciones –la probabilidad *a priori* de los nucleótidos A-T y C-G– no influye en la matriz de alineamiento, por lo tanto, las matrices de alineamiento finales de las tres ejecuciones son iguales. Sin embargo, la frecuencia esperada del *information content* de una matriz sí depende de la probabilidad de aparición de los distintos nucleótidos, dado que proviene de multiplicar la cantidad posible de alineamientos por la probabilidad de suceso del *information content* de la matriz, y esta última medida difiere para distintas probabilidades *a priori* de los nucleótidos (ver Fórmula 1.2). Como se puede comprobar en los resultados mostrados en la Tabla 4.5, la frecuencia esperada del *information content* es distinta en cada caso, obteniéndose los mejores resultados (frecuencia esperada más baja) en la primer ejecución, en que todos los nucleótidos tienen la misma probabilidad de aparición.

Para la búsqueda de alineamientos de distintas longitudes en secuencias de longitud 56 se obtuvo en cada caso más de una matriz final. En las Tablas 4.6, 4.7 y 4.8 se muestran las cinco secuencias consenso correspondientes a las matrices finales de las ejecuciones 4, 5 y 6 respectivamente. Se muestra también el ancho del alineamiento encontrado, la frecuencia esperada del *information content* y el promedio y desviación estándar de los puntajes resultantes de la ejecución de Patser con el mismo conjunto de datos y misma frecuencia de aparición utilizados para el armado del alineamiento.

Como puede observarse en la Tabla 4.6, los resultados de las matrices que representan los distintos alineamientos de la cuarta ejecución son muy parecidos: en todas las secuencias consenso de las matrices aparece la subsecuencia **TTGAaAN{17}TATAAT** (está marcado en negrita en la tabla) y la frecuencia esperada de todas las matrices tiene el mismo orden de magnitud, por esta razón nos quedamos con la matriz 2, que tiene menos desviación estándar.

En la Tabla 4.7 se pueden ver los resultados de la quinta ejecución de Consensus.

```

TTTTTTTTTTTTTTTT[A,T]TTTTTTTTTTTTTTTAAAAATATATCTCATTTTA

A 115 101 115 95 102 92 84 91 114 108 113 134 121 107 105 97 111 100 115 116 98 117 113
C 70 95 67 94 80 76 76 78 94 91 94 64 83 86 86 98 108 100 89 82 78 87 87
G 99 88 84 81 84 69 87 105 69 82 84 69 78 80 97 80 76 92 82 83 99 79 92
T 122 122 140 136 140 169 159 132 129 125 115 139 124 133 118 131 111 114 120 125 131 123 114

A 91 86 102 80 89 135 174 149 163 145 95 101 127 88 98 106 64 166 99 106 102 117 132
C 85 92 66 74 69 44 43 58 67 60 77 88 93 107 108 84 164 49 91 106 88 89 96
G 94 82 106 122 84 68 62 70 59 57 53 84 72 90 101 93 52 126 76 55 86 73 63
T 136 146 132 130 164 159 127 129 117 144 181 133 114 121 99 123 126 65 140 139 130 127 115

```

Tabla 4.4: Matriz de alineamiento resultante de las ejecuciones 1, 2 y 3 de Consensus (con secuencias de longitud 46). En el primer renglón se muestra la secuencia consenso del alineamiento y luego se muestra la matriz de alineamiento.

| Ejecución | Frecuencia esperada |
|-----------|------------------------|
| 1 | $1.73 \cdot 10^{-202}$ |
| 2 | $5.16 \cdot 10^{-85}$ |
| 3 | $2.91 \cdot 10^{-71}$ |

Tabla 4.5: Frecuencias esperadas de las matrices resultantes de las ejecuciones 1, en que las probabilidades *a priori* de los nucleótidos son iguales, 2, en que las probabilidades *a priori* de los nucleótidos AT y CG son 0.3 y 0.2 respectivamente y 3, en que la probabilidad *a priori* de los nucleótidos depende de la frecuencia observada en el conjunto de entrenamiento.

Las matrices 1 a 4 tienen los patrones TTGAaAN{17}TATAAT y la quinta TTGAatN{17}TATAAT, que es peor ya que hay dos nucleótidos en lugar de uno que no coinciden con el consenso conocido, descartamos entonces la matriz 5. La frecuencia esperada de las cuatro primeras matrices tiene el mismo orden de magnitud, entre éstas las mejores son la 3 y luego la 4 (la diferencia entre todas no es significativa). De estas dos nos quedamos con la tercer matriz, que es la que tiene menor desviación estándar.

Como puede verse en la Tabla 4.8, las cinco secuencias consenso arrojadas por la sexta ejecución contienen los patrones TTGAaAN{17}TATAAT. Dado que la frecuencia esperada de las cuatro primeras matrices es menor que la de la quinta matriz, nos quedamos en principio con estas cuatro y descartamos la de ancho 49. Como la frecuencia esperada de las primeras cuatro es del mismo orden de magnitud, nos quedamos con la primer matriz, que es la de menor desviación estándar.

Resumiendo, las matrices elegidas para cada ejecución son las mostradas en la Tabla 4.9. Una vez elegida la mejor matriz de cada ejecución de Consensus, ejecutamos Patser utilizando cada una de estas.

Patser

En la Tabla 4.10 mostramos para las distintas matrices de Consensus los mayores SCCs encontrados a partir de la ejecución de Patser y en la Figura 4.5 mostramos los SCCs alcanzados por Patser para los distintos umbrales, con las distintas matrices. En ésta se puede ver que los SCCs máximos no son puntos aislados y que las matrices con mayor SCC no sólo tienen SCCs mayores para un umbral determinado.

Elegimos los mejores resultados para las secuencias de longitud 46 y para las de longitud 56.

Como puede verse en la Tabla 4.10, la matriz que mejores resultados tiene es la de la ejecución 4. A su vez, entre las matrices generadas para las búsquedas realizadas

| Matriz | Prom. | Desv. estándar | Frecuencia esperada | Secuencia consenso | Ancho |
|--------|-------|----------------|------------------------|--|-------|
| 1 | 5.02 | 2.3078 | $4.94 \cdot 10^{-324}$ | TTATT[G.T]TTGAAAATTTTT[A.T]ATTTTTGGTATAATGCACCCATTTT[A.T]A | 49 |
| 2 | 5.02 | 2.3076 | $4.94 \cdot 10^{-324}$ | ATAATCTTGAAAATTTTT[A.T]ATTTTTGGTATAATGCACCCATTTT[A.T]A | 49 |
| 3 | 5.02 | 2.3109 | $4.94 \cdot 10^{-324}$ | TTATT[G.T]TTGAAAATTTTTATTTTTGGTATAATGCACCCATTTTAA | 49 |
| 4 | 5.02 | 2.3077 | $4.94 \cdot 10^{-324}$ | TTATTTTTGAAA[A.T]TTTTTTATTTTTGGTATAATGCACCCATTTT[A.T]A | 49 |
| 5 | 5.06 | 2.3408 | $9.88 \cdot 10^{-324}$ | ATAATCTTGAAAATTTCT[C.T]ATAATTTGGTATAATGCATCCCATTTT[A.T] | 48 |

Tabla 4.6: Resultados de la ejecución número 4 de Consensus, en donde los cuatro nucleótidos tienen probabilidad *a priori* 0.25 y las secuencias tienen longitud 56.

| Matriz | Prom. | Desv. estándar | Frecuencia esperada | Secuencia consenso | Ancho |
|--------|-------|----------------|------------------------|---|-------|
| 1 | 4.22 | 2.2464 | $5.91 \cdot 10^{-178}$ | ATAATCTTGAAA[A.T]TTTTT[C.G]TATTATCTGGTATAATGCATCCCATTTT | 48 |
| 2 | 4.22 | 2.2486 | $5.91 \cdot 10^{-178}$ | ATAATCTTGAAA[A.T]TTTTTGTATTATCTGGTATAATGCATCCCATTTT | 48 |
| 3 | 4.22 | 2.2419 | $6.37 \cdot 10^{-178}$ | ATAATCTTGAAAATTTTTGTATTATCTGGTATAATGGA[C.T]CCCATTTT | 48 |
| 4 | 4.21 | 2.2528 | $6.53 \cdot 10^{-178}$ | ATAATCTTGAAAATTTTTGTATTATCTGGTATAATGCATCCCATTTT | 48 |
| 5 | 4.25 | 2.2478 | $1.63 \cdot 10^{-175}$ | TAATCTTGAATTTTTCTGTATTATCTGGTATAATGCATCCCATTTT[A.T] | 47 |

Tabla 4.7: Resultados de la ejecución número 5 de Consensus, en que la probabilidad *a priori* de AT es 0.3 y de CG es 0.2 y las secuencias tienen longitud 56.

| Matriz | Prom. | Desv. estándar | Frecuencia esperada | Secuencia consenso | Ancho |
|--------|-------|----------------|------------------------|---|-------|
| 1 | 5.07 | 2.2897 | $2.59 \cdot 10^{-163}$ | ATAATCTTGAAAATTTTTTATTATTATTTGGTATAATGCATCCCATTTTAA | 48 |
| 2 | 5.06 | 2.3043 | $2.71 \cdot 10^{-163}$ | ATAATCTTGAAAATTTTTTATTATTATTGGTATAATGCATCCCATTTAA | 48 |
| 3 | 5.06 | 2.3013 | $2.87 \cdot 10^{-163}$ | ATAATCTTGAAAATTTTTTATTATTATTGGTATAATGCATCCCATTTAA | 48 |
| 4 | 5.06 | 2.3018 | $3.20 \cdot 10^{-163}$ | ATAATCTTGAAAATTTTTTATTATTATTGGTATAATGCATCCCATTTAA | 48 |
| 5 | 4.99 | 2.3116 | $1.94 \cdot 10^{-160}$ | TTATCTTGAAA[A.T]TTTTTTTATTATTGGTATAATGCATCCCATTTAA | 49 |

Tabla 4.8: Resultados de la ejecución número 6 de Consensus, en que la probabilidad *a priori* de los nucleótidos es la frecuencia observada en la secuencia de entrenamiento y las secuencias tienen longitud 56.

| Número de ejecución | Identificación de matriz | Tabla |
|---------------------|--------------------------|-------|
| 1 | la única final | 4.4 |
| 2 | la única final | 4.4 |
| 3 | la única final | 4.4 |
| 4 | matriz 2 | 4.6 |
| 5 | matriz 3 | 4.7 |
| 6 | matriz 1 | 4.8 |

Tabla 4.9: Resumen de matrices elegidas de cada ejecución de Consensus. En la columna *tabla* se muestra la tabla en la que se muestra la matriz.

en secuencias de longitud 56, ésta es la que tiene menor frecuencia esperada (Tablas 4.6, 4.7 y 4.8). Esta es la matriz que utilizaremos para la evaluación de los resultados del método Consensus-Patser con secuencias de longitud 56. En la evaluación de secuencias con longitud 46, los mejores resultados se obtienen con la matriz resultante de la primer ejecución, esta matriz será la utilizada para evaluación de los resultados del método Consensus-Patser con cadenas de longitud 46.

| Matriz | Máximo SCC conjunto de test |
|--------------------------------|-----------------------------|
| matriz final ejecución 1 | 0.56 |
| matriz final ejecución 2 | 0.40 |
| matriz final ejecución 3 | 0.50 |
| matriz 2 de ejecución 4 | 0.67 |
| matriz 3 de ejecución 5 | 0.52 |
| matriz 1 de ejecución 6 | 0.59 |

Tabla 4.10: Resultados de las ejecuciones de Patser.

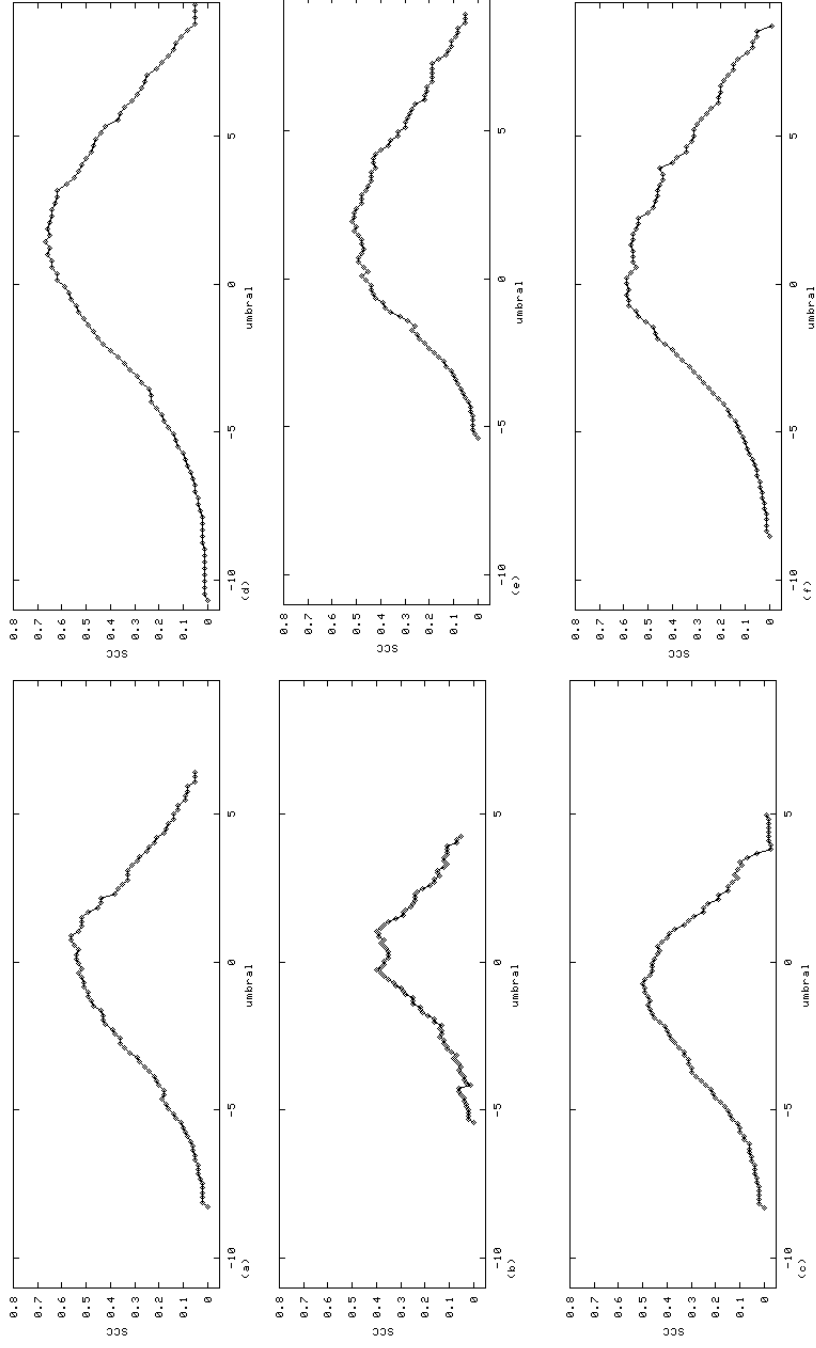


Figura 4.5: Resultados de las ejecuciones de Patser para las mejores matrices elegidas de las distintas ejecuciones de Consensus. En cada gráfico se muestran los SCCs obtenidos para distintos umbrales. En (a), (b) y (c) se consideran las matrices obtenidas mediante las ejecuciones de Consensus, en que se realizaron alineamientos de ancho 46 en secuencias de longitud 46 con distintas probabilidades *a priori* de los nucleótidos (correspondientes a las ejecuciones 1, 2 y 3 mostradas en la Tabla 4.3) y en (c), (d) y (e) se consideran las matrices obtenidas mediante las ejecuciones de Consensus en que se realizaron alineamientos de anchos entre 42 y 49 en secuencias de longitud 56 (correspondientes a las ejecuciones 4, 5 y 6 mostradas en la Tabla 4.3). En (a) y (d) se consideró que la probabilidad *a priori* de los cuatro nucleótidos es equiprobable. En (b) y (e) se consideró 0.3 como probabilidad *a priori* de A y T y 0.2 como probabilidad *a priori* de C y G. En (c) y (f) se consideró como probabilidad *a priori* la observada en las secuencias de entrada.

En las Figuras 4.6 y 4.7 y las Tablas 4.11 y 4.12 se muestran los resultados de la ejecución de Consensus y Patser con las matrices seleccionadas. Para los distintos umbrales considerados se muestran los porcentajes de falsos positivos, de verdaderos positivos y el SCC del conjunto de test ordenados ascendentemente por el porcentaje de falsos positivos. En los resultados de las pruebas realizadas con cadenas de longitud 56, el máximo SCC (0.67) tiene 81.61% TP y 15.06% FP (ver Tabla 4.12). En las tablas se resaltaron los resultados correspondientes a valores de FP menores o iguales al 1% y a los SCCs más altos.

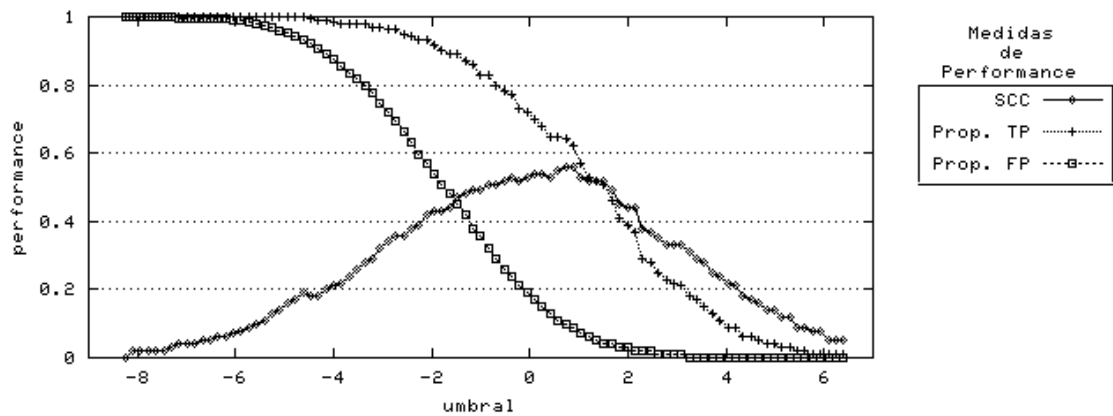


Figura 4.6: Proporciones de TP, SCCs y FP para distintos umbrales en el conjunto de test. Secuencias de Longitud 46.

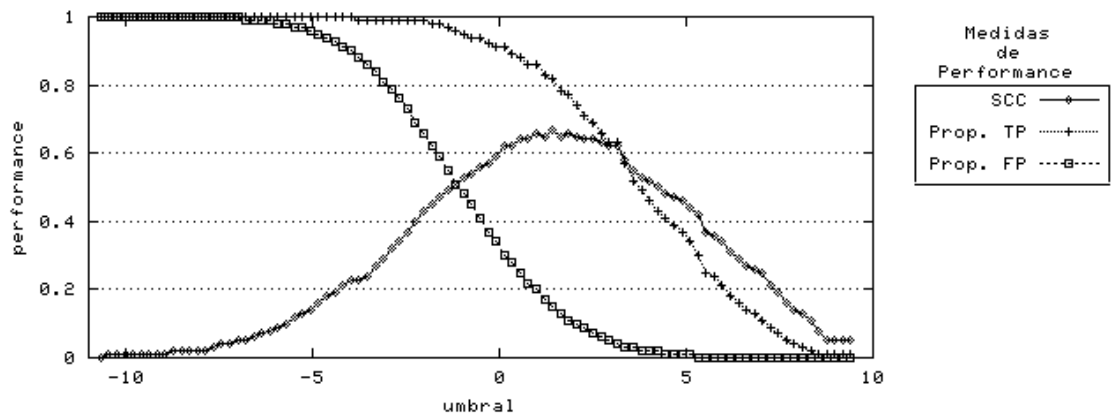


Figura 4.7: Proporciones de TP, SCCs y FP para distintos umbrales en el conjunto de test. Secuencias de Longitud 56.

| Umbral | % FP | % TP | SCC | Umbral | % FP | % TP | SCC |
|--------|-------|-------|------|--------|--------|--------|------|
| 6.40 | 0.00 | 0.57 | 0.05 | -1.80 | 50.87 | 90.23 | 0.43 |
| 6.24 | 0.00 | 0.57 | 0.05 | -1.95 | 54.00 | 91.95 | 0.43 |
| 6.09 | 0.00 | 0.57 | 0.05 | -2.11 | 56.92 | 93.10 | 0.42 |
| 5.93 | 0.00 | 1.15 | 0.08 | -2.27 | 59.84 | 93.10 | 0.39 |
| 5.77 | 0.00 | 1.15 | 0.08 | -2.43 | 63.43 | 94.25 | 0.38 |
| 5.61 | 0.00 | 1.72 | 0.09 | -2.58 | 66.41 | 94.83 | 0.36 |
| 5.46 | 0.00 | 1.72 | 0.09 | -2.74 | 69.26 | 96.55 | 0.36 |
| 5.30 | 0.00 | 2.87 | 0.12 | -2.90 | 71.86 | 96.55 | 0.34 |
| 5.14 | 0.00 | 2.87 | 0.12 | -3.06 | 74.64 | 97.13 | 0.32 |
| 4.98 | 0.02 | 4.02 | 0.14 | -3.21 | 77.84 | 97.13 | 0.29 |
| 4.83 | 0.05 | 4.02 | 0.14 | -3.37 | 79.86 | 97.70 | 0.28 |
| 4.67 | 0.07 | 5.17 | 0.16 | -3.53 | 81.75 | 97.70 | 0.26 |
| 4.51 | 0.07 | 5.75 | 0.17 | -3.69 | 83.61 | 97.70 | 0.24 |
| 4.35 | 0.07 | 6.32 | 0.18 | -3.85 | 85.66 | 97.70 | 0.22 |
| 4.20 | 0.09 | 8.62 | 0.21 | -4.00 | 87.72 | 98.28 | 0.21 |
| 4.04 | 0.11 | 9.20 | 0.22 | -4.16 | 89.22 | 98.85 | 0.20 |
| 3.88 | 0.16 | 11.49 | 0.24 | -4.32 | 90.76 | 98.85 | 0.18 |
| 3.72 | 0.21 | 12.64 | 0.25 | -4.48 | 92.02 | 99.43 | 0.18 |
| 3.56 | 0.30 | 14.94 | 0.28 | -4.63 | 93.24 | 100.00 | 0.19 |
| 3.41 | 0.37 | 16.67 | 0.29 | -4.79 | 94.44 | 100.00 | 0.17 |
| 3.25 | 0.44 | 18.39 | 0.31 | -4.95 | 95.29 | 100.00 | 0.16 |
| 3.09 | 0.57 | 20.69 | 0.33 | -5.11 | 96.07 | 100.00 | 0.14 |
| 2.93 | 0.78 | 21.84 | 0.33 | -5.26 | 96.92 | 100.00 | 0.13 |
| 2.78 | 1.17 | 22.99 | 0.33 | -5.42 | 97.45 | 100.00 | 0.11 |
| 2.62 | 1.43 | 25.29 | 0.35 | -5.58 | 97.98 | 100.00 | 0.10 |
| 2.46 | 1.63 | 27.59 | 0.37 | -5.74 | 98.32 | 100.00 | 0.09 |
| 2.30 | 1.93 | 29.31 | 0.38 | -5.90 | 98.71 | 100.00 | 0.08 |
| 2.15 | 2.23 | 36.78 | 0.44 | -6.05 | 99.03 | 100.00 | 0.07 |
| 1.99 | 2.85 | 38.51 | 0.44 | -6.21 | 99.26 | 100.00 | 0.06 |
| 1.83 | 3.45 | 40.80 | 0.45 | -6.37 | 99.38 | 100.00 | 0.06 |
| 1.67 | 3.82 | 45.98 | 0.49 | -6.53 | 99.43 | 100.00 | 0.05 |
| 1.51 | 4.41 | 50.57 | 0.52 | -6.68 | 99.56 | 100.00 | 0.05 |
| 1.36 | 5.20 | 52.30 | 0.52 | -6.84 | 99.61 | 100.00 | 0.04 |
| 1.20 | 6.21 | 53.45 | 0.52 | -7.00 | 99.66 | 100.00 | 0.04 |
| 1.04 | 7.20 | 56.90 | 0.53 | -7.16 | 99.72 | 100.00 | 0.04 |
| 0.88 | 8.51 | 62.07 | 0.56 | -7.31 | 99.84 | 100.00 | 0.03 |
| 0.73 | 9.68 | 63.79 | 0.56 | -7.47 | 99.89 | 100.00 | 0.02 |
| 0.57 | 11.15 | 64.94 | 0.55 | -7.63 | 99.95 | 100.00 | 0.02 |
| 0.41 | 13.10 | 64.94 | 0.53 | -7.79 | 99.95 | 100.00 | 0.02 |
| 0.25 | 14.85 | 67.82 | 0.54 | -7.94 | 99.95 | 100.00 | 0.02 |
| 0.10 | 17.03 | 70.11 | 0.54 | -8.10 | 99.95 | 100.00 | 0.02 |
| -0.06 | 19.24 | 71.84 | 0.53 | -8.26 | 100.00 | 100.00 | 0.00 |
| -0.22 | 21.22 | 72.99 | 0.52 | | | | |
| -0.38 | 23.84 | 77.01 | 0.53 | | | | |
| -0.53 | 26.37 | 78.16 | 0.52 | | | | |
| -0.69 | 29.43 | 80.46 | 0.51 | | | | |
| -0.85 | 32.09 | 82.76 | 0.51 | | | | |
| -1.01 | 35.63 | 83.33 | 0.49 | | | | |
| -1.17 | 38.30 | 85.63 | 0.49 | | | | |
| -1.32 | 41.52 | 87.36 | 0.48 | | | | |
| -1.48 | 45.20 | 89.08 | 0.47 | | | | |
| -1.64 | 48.05 | 89.08 | 0.44 | | | | |

Referencias


FP \leq 1%

SCCs más altos

Tabla 4.11: Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de test, ordenados ascendentemente por FP. Secuencias de longitud 46. Se resaltan los resultados correspondientes a cantidades de $FP \leq 1\%$ y a los SCCs más altos.

| Umbral | % FP | % TP | SCC |
|--------|--------|--------|------|
| 9.41 | 0.00 | 0.57 | 0.05 |
| 9.20 | 0.00 | 0.57 | 0.05 |
| 8.98 | 0.00 | 0.57 | 0.05 |
| 8.77 | 0.00 | 0.57 | 0.05 |
| 8.55 | 0.00 | 1.15 | 0.08 |
| 8.33 | 0.00 | 2.30 | 0.11 |
| 8.12 | 0.00 | 3.45 | 0.13 |
| 7.90 | 0.00 | 4.02 | 0.14 |
| 7.69 | 0.02 | 5.17 | 0.16 |
| 7.47 | 0.05 | 6.90 | 0.19 |
| 7.25 | 0.05 | 8.62 | 0.21 |
| 7.04 | 0.05 | 11.49 | 0.25 |
| 6.82 | 0.05 | 12.64 | 0.26 |
| 6.61 | 0.05 | 13.79 | 0.27 |
| 6.39 | 0.09 | 16.09 | 0.29 |
| 6.17 | 0.09 | 17.82 | 0.31 |
| 5.96 | 0.14 | 20.69 | 0.34 |
| 5.74 | 0.23 | 23.56 | 0.36 |
| 5.52 | 0.28 | 25.29 | 0.37 |
| 5.31 | 0.44 | 30.46 | 0.42 |
| 5.09 | 0.55 | 33.91 | 0.44 |
| 4.88 | 0.85 | 36.78 | 0.46 |
| 4.66 | 1.08 | 38.51 | 0.47 |
| 4.44 | 1.36 | 40.80 | 0.48 |
| 4.23 | 1.59 | 43.10 | 0.50 |
| 4.01 | 1.98 | 45.98 | 0.52 |
| 3.80 | 2.37 | 48.85 | 0.53 |
| 3.58 | 2.83 | 52.30 | 0.55 |
| 3.36 | 3.49 | 56.90 | 0.58 |
| 3.15 | 4.16 | 62.64 | 0.62 |
| 2.93 | 4.90 | 63.22 | 0.62 |
| 2.72 | 5.82 | 66.09 | 0.63 |
| 2.50 | 7.03 | 68.97 | 0.64 |
| 2.28 | 8.51 | 70.69 | 0.64 |
| 2.07 | 10.05 | 74.14 | 0.65 |
| 1.85 | 11.45 | 77.01 | 0.66 |
| 1.64 | 13.03 | 78.16 | 0.65 |
| 1.42 | 15.06 | 81.61 | 0.67 |
| 1.20 | 17.29 | 82.76 | 0.65 |
| 0.99 | 19.93 | 85.63 | 0.66 |
| 0.77 | 22.05 | 86.21 | 0.64 |
| 0.56 | 24.92 | 87.93 | 0.64 |
| 0.34 | 27.66 | 89.08 | 0.62 |
| 0.12 | 30.37 | 90.80 | 0.62 |
| -0.09 | 33.86 | 91.38 | 0.59 |
| -0.31 | 36.99 | 91.95 | 0.57 |
| -0.52 | 40.76 | 93.68 | 0.56 |
| -0.74 | 44.74 | 94.25 | 0.54 |
| -0.96 | 47.98 | 95.40 | 0.53 |
| -1.17 | 51.47 | 95.98 | 0.51 |
| -1.39 | 54.78 | 96.55 | 0.49 |
| -1.61 | 58.55 | 97.70 | 0.47 |
| -1.82 | 62.16 | 98.28 | 0.45 |
| -2.04 | 65.86 | 98.85 | 0.43 |
| -2.25 | 69.24 | 98.85 | 0.40 |
| -2.47 | 72.64 | 98.85 | 0.37 |
| -2.69 | 75.98 | 98.85 | 0.34 |
| -2.90 | 78.74 | 98.85 | 0.32 |
| -3.12 | 81.22 | 98.85 | 0.29 |
| -3.33 | 83.93 | 98.85 | 0.27 |
| -3.55 | 85.86 | 98.85 | 0.24 |
| -3.77 | 88.07 | 99.43 | 0.23 |
| -3.98 | 89.95 | 100.00 | 0.23 |
| -4.20 | 91.45 | 100.00 | 0.21 |
| -4.41 | 92.74 | 100.00 | 0.19 |
| -4.63 | 94.02 | 100.00 | 0.18 |
| -4.85 | 95.13 | 100.00 | 0.16 |
| -5.06 | 95.98 | 100.00 | 0.14 |
| -5.28 | 96.64 | 100.00 | 0.13 |
| -5.49 | 97.36 | 100.00 | 0.12 |
| -5.71 | 97.86 | 100.00 | 0.10 |
| -5.93 | 98.23 | 100.00 | 0.09 |
| -6.14 | 98.71 | 100.00 | 0.08 |
| -6.36 | 99.03 | 100.00 | 0.07 |
| -6.57 | 99.22 | 100.00 | 0.06 |
| -6.79 | 99.45 | 100.00 | 0.05 |
| -7.01 | 99.56 | 100.00 | 0.05 |
| -7.22 | 99.68 | 100.00 | 0.04 |
| -7.44 | 99.72 | 100.00 | 0.04 |
| -7.66 | 99.82 | 100.00 | 0.03 |
| -7.87 | 99.89 | 100.00 | 0.02 |
| -8.09 | 99.95 | 100.00 | 0.02 |
| -8.30 | 99.95 | 100.00 | 0.02 |
| -8.52 | 99.95 | 100.00 | 0.02 |
| -8.74 | 99.95 | 100.00 | 0.02 |
| -8.95 | 99.98 | 100.00 | 0.01 |
| -9.17 | 99.98 | 100.00 | 0.01 |
| -9.38 | 99.98 | 100.00 | 0.01 |
| -9.60 | 99.98 | 100.00 | 0.01 |
| -9.82 | 99.98 | 100.00 | 0.01 |
| -10.03 | 99.98 | 100.00 | 0.01 |
| -10.25 | 99.98 | 100.00 | 0.01 |
| -10.46 | 99.98 | 100.00 | 0.01 |
| -10.68 | 100.00 | 100.00 | 0.00 |

Referencias

 FP \leq 1%


 SCCs más altos

Tabla 4.12: Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de test, ordenados ascendentemente por FP. Secuencias de longitud 56. Se resaltan los resultados correspondientes a cantidades de $FP \leq 1\%$ y a los SCCs más altos.

En la Tabla 4.13 se muestran los resultados de la ejecución de Patser con la mejor matriz para cadenas de longitud 46 y 56 con el conjunto de entrenamiento. Sólo se muestra el porcentaje de verdaderos positivos, dado que, como se explicó anteriormente, no se utilizaron datos de no promotores para realizar el entrenamiento y por esta razón entre los resultados no hay ni FP ni TN.

En la Tabla 4.14 se muestra un resumen de los resultados de la ejecución del método Consensus-Patser con el conjunto de test con la mejor matriz arrojada por Consensus para las cadenas de longitud 56.

4.3.3. Análisis de resultados

Todas las matrices finales de las ejecuciones de Consensus, que tomaron como entrada secuencias de longitud 56 (ejecuciones 4,5 y 6), con excepción de una, contienen la secuencia consenso TTGAaAN{17}TATAAT. Esta secuencia coincide en 11 de los 12 nucleótidos fuertemente conservados según las compilaciones y análisis de datos existentes y en particular tiene los 6 nucleótidos con mayor conservación (TTG _ _ _ y TA _ _ T _).

Para las secuencias de longitud 49 y 56 los mejores resultados se dieron en el caso en que los cuatro nucleótidos tenían la misma probabilidad de aparición.

La matriz que mejores resultados tiene es aquella generada a partir del alineamiento de secuencias de longitud 56 y con probabilidad *a priori* de las bases = 0.25 (Tabla 4.10). A la vez esta es la matriz que tiene menor frecuencia esperada (como se puede ver en la Tabla 4.6).

Los resultados son mejores que los de los métodos fijos. Esto tiene sentido ya que para el cálculo de puntajes que determinan si una secuencia es o no promotora, se tiene en cuenta la matriz de pesos y no sólo los consensos. Esto nos permite ver que la utilización de la información de los nucleótidos que no son los más frecuentes (de forma tal de no limitarnos a la secuencia consenso) es importante.

En las Tablas 4.6, 4.7 y 4.8 puede observarse un resultado interesante: en las 15 secuencias consenso mostradas, los consensos TATAAT están precedidos por el consenso TGG y sucedidos por el consenso GCA. De acuerdo a estos resultados el consenso de la región -10 estaría formado por la secuencia: TGGTATAATGCA. También puede observarse que el espacio entre las secuencias TTGAAA y TATAAT resaltadas está mayormente compuesto por As y Ts.

| Umbral | % TP | Umbral | % TP | Umbral | % TP | Umbral | % TP |
|--------|--------|--------|-------|--------|--------|--------|-------|
| -3.69 | 100.00 | 2.89 | 29.06 | -5.00 | 100.00 | 3.91 | 62.56 |
| -3.56 | 99.75 | 3.01 | 27.83 | -4.83 | 99.75 | 4.08 | 61.33 |
| -3.44 | 99.75 | 3.14 | 25.12 | -4.66 | 99.75 | 4.25 | 59.36 |
| -3.31 | 99.75 | 3.27 | 23.89 | -4.49 | 99.75 | 4.43 | 56.16 |
| -3.18 | 99.26 | 3.39 | 21.67 | -4.31 | 99.75 | 4.60 | 53.69 |
| -3.06 | 98.77 | 3.52 | 19.95 | -4.14 | 99.75 | 4.77 | 51.72 |
| -2.93 | 98.77 | 3.65 | 18.23 | -3.97 | 99.75 | 4.94 | 50.25 |
| -2.80 | 98.77 | 3.77 | 16.50 | -3.80 | 99.75 | 5.11 | 47.54 |
| -2.68 | 98.52 | 3.90 | 15.52 | -3.63 | 99.51 | 5.28 | 45.32 |
| -2.55 | 98.52 | 4.03 | 14.04 | -3.46 | 99.51 | 5.45 | 42.61 |
| -2.43 | 98.28 | 4.15 | 13.30 | -3.29 | 99.01 | 5.63 | 40.39 |
| -2.30 | 97.54 | 4.28 | 12.07 | -3.11 | 99.01 | 5.80 | 38.42 |
| -2.17 | 97.04 | 4.41 | 11.08 | -2.94 | 99.01 | 5.97 | 35.47 |
| -2.05 | 96.55 | 4.53 | 9.85 | -2.77 | 99.01 | 6.14 | 33.50 |
| -1.92 | 96.31 | 4.66 | 8.37 | -2.60 | 99.01 | 6.31 | 30.54 |
| -1.79 | 94.83 | 4.78 | 7.39 | -2.43 | 99.01 | 6.48 | 27.59 |
| -1.67 | 93.60 | 4.91 | 7.14 | -2.26 | 99.01 | 6.65 | 24.88 |
| -1.54 | 92.86 | 5.04 | 5.91 | -2.09 | 99.01 | 6.83 | 23.40 |
| -1.41 | 91.13 | 5.16 | 4.93 | -1.92 | 98.77 | 7.00 | 21.43 |
| -1.29 | 89.90 | 5.29 | 4.19 | -1.74 | 98.52 | 7.17 | 17.24 |
| -1.16 | 88.67 | 5.42 | 3.69 | -1.57 | 98.03 | 7.34 | 15.52 |
| -1.03 | 87.68 | 5.54 | 3.20 | -1.40 | 97.78 | 7.51 | 13.55 |
| -0.91 | 86.45 | 5.67 | 2.96 | -1.23 | 97.29 | 7.68 | 11.58 |
| -0.78 | 85.96 | 5.80 | 2.46 | -1.06 | 96.80 | 7.85 | 10.59 |
| -0.65 | 84.24 | 5.92 | 2.46 | -0.89 | 96.55 | 8.03 | 9.61 |
| -0.53 | 83.50 | 6.05 | 1.97 | -0.72 | 96.06 | 8.20 | 8.13 |
| -0.40 | 82.02 | 6.18 | 1.97 | -0.54 | 96.06 | 8.37 | 7.14 |
| -0.27 | 80.79 | 6.30 | 1.48 | -0.37 | 95.32 | 8.54 | 6.65 |
| -0.15 | 78.08 | 6.43 | 1.23 | -0.20 | 94.83 | 8.71 | 3.94 |
| -0.02 | 76.85 | 6.56 | 0.99 | -0.03 | 94.58 | 8.88 | 2.96 |
| 0.10 | 75.62 | 6.68 | 0.74 | 0.14 | 93.84 | 9.05 | 2.71 |
| 0.23 | 74.38 | 6.81 | 0.74 | 0.31 | 93.84 | 9.22 | 2.22 |
| 0.36 | 72.66 | 6.94 | 0.74 | 0.48 | 93.10 | 9.40 | 2.22 |
| 0.48 | 70.20 | 7.06 | 0.49 | 0.66 | 92.12 | 9.57 | 1.97 |
| 0.61 | 68.23 | 7.19 | 0.25 | 0.83 | 91.63 | 9.74 | 1.72 |
| 0.74 | 66.50 | 7.31 | 0.25 | 1.00 | 90.64 | 9.91 | 1.48 |
| 0.86 | 64.53 | 7.44 | 0.25 | 1.17 | 89.90 | 10.08 | 1.48 |
| 0.99 | 62.56 | 7.57 | 0.25 | 1.34 | 88.42 | 10.25 | 1.48 |
| 1.12 | 60.59 | 7.69 | 0.25 | 1.51 | 87.68 | 10.42 | 0.74 |
| 1.24 | 58.62 | 7.82 | 0.25 | 1.68 | 86.45 | 10.60 | 0.25 |
| 1.37 | 55.91 | 7.95 | 0.25 | 1.86 | 85.71 | 10.77 | 0.25 |
| 1.50 | 54.19 | 8.07 | 0.25 | 2.03 | 84.24 | 10.94 | 0.25 |
| 1.62 | 52.71 | 8.20 | 0.25 | 2.20 | 83.25 | 11.11 | 0.25 |
| 1.75 | 50.00 | | | 2.37 | 81.53 | | |
| 1.88 | 48.52 | | | 2.54 | 78.57 | | |
| 2.00 | 46.06 | | | 2.71 | 77.09 | | |
| 2.13 | 42.86 | | | 2.88 | 74.38 | | |
| 2.25 | 40.64 | | | 3.05 | 72.41 | | |
| 2.38 | 39.16 | | | 3.23 | 71.67 | | |
| 2.51 | 36.95 | | | 3.40 | 69.95 | | |
| 2.63 | 35.22 | | | 3.57 | 68.47 | | |
| 2.76 | 32.02 | | | 3.74 | 66.26 | | |

(a)

(b)

Tabla 4.13: Consensus-Patser - Porcentajes de TP, SCCs y umbrales para distintos porcentajes de FP en el conjunto de entrenamiento. (a) secuencias de longitud 46 y (b) secuencias de longitud 56. Sólo se muestra el porcentaje de TP, dado que al no utilizarse datos de no promotores para realizar el entrenamiento, entre los resultados no hay ni FP ni TN.

| Cjto. | 0 %FP | | 1 % FP | | 2 % FP | | 5 % FP | | 100 %TP | | 80 %TP | | máximo SCC | | |
|-------------|-------|------|--------|------|--------|------|--------|------|---------|------|--------|------|------------|-------|-------|
| | %TP | SCC | %TP | SCC | %TP | SCC | %TP | SCC | %FP | SCC | %FP | SCC | SCC | %FP | %TP |
| test | 4.6 | 0.15 | 37.93 | 0.47 | 46.55 | 0.52 | 64.37 | 0.62 | 88.39 | 0.25 | 13.86 | 0.66 | 0.67 | 15.06 | 81.61 |

Tabla 4.14: Resultados de la aplicación del método Consensus-Patser al conjunto de test para 0%, 1%, 5% de FP y 80% y 100% de TP.

4.4. Conexionist Promoter Recognition

En el Capítulo 3 realizamos el entrenamiento de una red neuronal con los datos de la compilación de Harley y Reynolds. En esta sección explicaremos el reentrenamiento realizado con los datos de RegulonDB.

No hay un método que sea el mejor para reentrenar una red neuronal con nuevos datos. Algunas de las alternativas están incluidas entre las siguientes: 1) utilizar como pesos iniciales de la nueva red los pesos resultantes del entrenamiento original y reentrenar la red con los datos nuevos [53], 2) modificar los pesos de la red entrenada originalmente con alguna función y luego reentrenarla [53], 3) entrenar una red nueva. Otro factor a considerar es qué datos utilizar para el reentrenamiento, si se reentrena sólo con los datos nuevos (desconocidos para la red) es posible que la red “se olvide” de lo que aprendió en el primer entrenamiento, pero si se pretende que la red también reconozca los datos con la que se la entrenó originalmente es posible que haya que incluirlos entre los datos del segundo entrenamiento, ya que de lo contrario posiblemente los olvidará [53].

Optamos por utilizar la red cuya construcción y *tuning* fue descrita en el Capítulo 3 utilizando como inicialización los pesos resultantes del entrenamiento. Como datos para el entrenamiento y test se utilizaron sólo los de RegulonDB versión 3.1. Se cree que esta elección constituye una ventaja, ya que éstos no cuentan con datos de fagos, que como se explicó anteriormente sesgan los datos. Por otro lado, los datos de RegulonDB incluyen a los datos de promotores de *E. coli* de Harley y Reynolds, pero en la versión corregida por Lisser y Margalit. Por esta razón los datos de un mismo promotor difieren bastante en ambas compilaciones.

4.4.1. Experimentación

Los parámetros de ejecución de la red neuronal pueden verse en la Tabla 4.15. El algoritmo *time delayed backpropagation* implementa el algoritmo de *back-propagation* modificado, cuyo pseudocódigo fue mostrado en el Capítulo 3. Se realizó el entrenamiento con los siguientes valores de η : 2.0, 0.2, 0.05, 0.01, 0.009 y 0.005. Con $\eta=0.01$ se obtuvieron mejores resultados en menor cantidad de épocas. El orden utilizado para la activación de las neuronas fue el orden dado por la topología de la red: primero se

| Variable | Selección |
|---|---|
| Función de activación de neuronas ocultas y de salida | logística |
| Shuffle | on |
| Algoritmo de aprendizaje | <i>time delayed back-propagation</i> $\eta=0.01$ |
| Inicialización de pesos y umbrales | pesos resultantes del entrenamiento de la red descrita en el Capítulo 3 |

Tabla 4.15: Parámetros utilizados en el entrenamiento de la red neuronal.

procesan las dos capas ocultas y luego la capa de salida. Como función de activación se utilizó la función logística mostrada en la Fórmula 2.3. Para la neurona de la capa de salida se utilizó la función escalón definida en el Capítulo 3 (Fórmula 3.1). La opción *shuffle= on* indica que en cada ciclo los patrones son presentados a la red en orden aleatorio. El promedio de épocas necesaria para el entrenamiento fue 800.

La red neuronal fue implementada con el SNNS (*Stuttgart Neural Network Simulator*) versión 4.1.


4.4.2. Resultados

En las Tablas 4.16 y 4.17 pueden verse los resultados de la evaluación de la red con los conjunto de test y de entrenamiento. Para distintos umbrales se muestran los porcentajes de TP y de FP y el SCC. Los resultados están ordenados crecientemente por porcentaje de FP. En las Figuras 4.8 y 4.9 se pueden ver gráficamente los resultados de sendas tablas.

| Umbral | % FP | % TP | SCC |
|--------|------|-------|------|
| 1 | 0.00 | 3.45 | 0.13 |
| 0.99 | 0.02 | 41.38 | 0.51 |
| 0.98 | 0.05 | 49.43 | 0.57 |
| 0.97 | 0.05 | 52.87 | 0.6 |
| 0.96 | 0.07 | 53.45 | 0.6 |
| 0.95 | 0.09 | 54.60 | 0.61 |
| 0.94 | 0.14 | 55.17 | 0.62 |
| 0.93 | 0.14 | 59.20 | 0.65 |
| 0.91 | 0.14 | 59.77 | 0.65 |
| 0.9 | 0.16 | 60.92 | 0.66 |
| 0.89 | 0.21 | 62.64 | 0.67 |
| 0.88 | 0.23 | 63.79 | 0.68 |
| 0.87 | 0.25 | 64.37 | 0.69 |
| 0.86 | 0.28 | 65.52 | 0.69 |
| 0.85 | 0.28 | 66.09 | 0.7 |
| 0.84 | 0.28 | 66.09 | 0.7 |
| 0.83 | 0.28 | 66.09 | 0.7 |
| 0.82 | 0.32 | 66.67 | 0.7 |
| 0.81 | 0.32 | 67.24 | 0.71 |
| 0.8 | 0.32 | 67.24 | 0.71 |
| 0.79 | 0.34 | 68.39 | 0.72 |
| 0.78 | 0.37 | 69.54 | 0.73 |
| 0.77 | 0.39 | 69.54 | 0.73 |
| 0.76 | 0.44 | 71.26 | 0.74 |
| 0.74 | 0.46 | 71.26 | 0.74 |
| 0.73 | 0.46 | 71.26 | 0.74 |
| 0.72 | 0.46 | 71.26 | 0.74 |
| 0.71 | 0.48 | 71.84 | 0.74 |
| 0.7 | 0.48 | 72.41 | 0.75 |
| 0.69 | 0.48 | 72.99 | 0.75 |
| 0.68 | 0.51 | 73.56 | 0.76 |
| 0.67 | 0.51 | 74.71 | 0.77 |
| 0.66 | 0.51 | 74.71 | 0.77 |
| 0.65 | 0.51 | 74.71 | 0.77 |
| 0.64 | 0.53 | 75.29 | 0.77 |
| 0.63 | 0.53 | 75.29 | 0.77 |
| 0.62 | 0.55 | 75.86 | 0.78 |
| 0.61 | 0.55 | 75.86 | 0.78 |
| 0.6 | 0.55 | 75.86 | 0.78 |
| 0.59 | 0.57 | 76.44 | 0.78 |
| 0.57 | 0.57 | 76.44 | 0.78 |
| 0.56 | 0.60 | 76.44 | 0.78 |
| 0.55 | 0.62 | 77.01 | 0.78 |
| 0.54 | 0.64 | 77.01 | 0.78 |
| 0.53 | 0.64 | 77.59 | 0.79 |
| 0.52 | 0.64 | 77.59 | 0.79 |
| 0.51 | 0.64 | 78.74 | 0.8 |
| 0.5 | 0.64 | 78.74 | 0.8 |
| 0.49 | 0.69 | 80.46 | 0.81 |
| 0.48 | 0.69 | 80.46 | 0.81 |
| 0.47 | 0.69 | 80.46 | 0.81 |
| 0.46 | 0.69 | 81.03 | 0.82 |
| 0.45 | 0.71 | 81.61 | 0.82 |
| 0.44 | 0.71 | 82.18 | 0.83 |
| 0.43 | 0.74 | 82.18 | 0.83 |
| 0.41 | 0.78 | 82.18 | 0.83 |
| 0.4 | 0.78 | 82.76 | 0.83 |
| 0.38 | 0.80 | 83.33 | 0.84 |

| Umbral | % FP | % TP | SCC |
|--------|--------|--------|------|
| 0.39 | 0.80 | 83.33 | 0.84 |
| 0.37 | 0.83 | 83.33 | 0.84 |
| 0.36 | 0.83 | 83.33 | 0.84 |
| 0.35 | 0.85 | 83.33 | 0.84 |
| 0.34 | 0.87 | 83.33 | 0.84 |
| 0.33 | 0.90 | 83.91 | 0.84 |
| 0.32 | 0.90 | 84.48 | 0.84 |
| 0.31 | 0.90 | 84.48 | 0.84 |
| 0.3 | 0.94 | 85.06 | 0.85 |
| 0.29 | 0.99 | 85.06 | 0.85 |
| 0.28 | 0.99 | 85.06 | 0.85 |
| 0.27 | 1.10 | 85.06 | 0.85 |
| 0.26 | 1.15 | 85.06 | 0.85 |
| 0.24 | 1.22 | 85.06 | 0.85 |
| 0.23 | 1.22 | 85.63 | 0.85 |
| 0.22 | 1.26 | 85.63 | 0.85 |
| 0.21 | 1.29 | 85.63 | 0.85 |
| 0.2 | 1.31 | 85.63 | 0.85 |
| 0.19 | 1.33 | 85.63 | 0.85 |
| 0.18 | 1.38 | 86.21 | 0.85 |
| 0.17 | 1.43 | 86.21 | 0.85 |
| 0.16 | 1.49 | 87.36 | 0.86 |
| 0.15 | 1.61 | 87.36 | 0.86 |
| 0.14 | 1.72 | 87.36 | 0.86 |
| 0.13 | 1.82 | 87.93 | 0.87 |
| 0.12 | 1.86 | 88.51 | 0.87 |
| 0.11 | 2.00 | 89.66 | 0.88 |
| 0.1 | 2.09 | 90.23 | 0.88 |
| 0.09 | 2.18 | 90.80 | 0.89 |
| 0.07 | 2.32 | 90.80 | 0.89 |
| 0.06 | 2.57 | 91.38 | 0.89 |
| 0.05 | 2.90 | 92.53 | 0.9 |
| 0.04 | 3.43 | 93.68 | 0.9 |
| 0.03 | 4.18 | 94.25 | 0.9 |
| 0.02 | 5.01 | 94.83 | 0.9 |
| 0.01 | 6.83 | 95.98 | 0.89 |
| 0 | 100.00 | 100.00 | 0 |

Referencias

 FP ≤ 1%


 SCCs más altos

Tabla 4.16: Red neuronal - Porcentajes de TP, FP y SCCs para distintos umbrales en el conjunto de test, ordenados ascendentemente por FP. Se resaltan los resultados que tienen un porcentaje de falsos positivos menor al 1% y los que tienen los SCCs más altos.

| Umbral | % FP | % TP | SCC |
|--------|--------|--------|------|
| 1.00 | 0.00 | 5.67 | 0.17 |
| 0.99 | 0.00 | 42.12 | 0.52 |
| 0.98 | 0.00 | 47.78 | 0.56 |
| 0.97 | 0.02 | 51.97 | 0.59 |
| 0.96 | 0.02 | 54.19 | 0.61 |
| 0.95 | 0.02 | 55.17 | 0.62 |
| 0.94 | 0.02 | 56.90 | 0.63 |
| 0.93 | 0.02 | 58.62 | 0.64 |
| 0.91 | 0.05 | 61.33 | 0.66 |
| 0.90 | 0.05 | 62.32 | 0.67 |
| 0.89 | 0.05 | 63.05 | 0.68 |
| 0.88 | 0.05 | 64.29 | 0.69 |
| 0.87 | 0.05 | 65.27 | 0.70 |
| 0.86 | 0.05 | 67.00 | 0.71 |
| 0.85 | 0.05 | 67.73 | 0.72 |
| 0.84 | 0.05 | 68.23 | 0.72 |
| 0.83 | 0.05 | 68.97 | 0.72 |
| 0.82 | 0.05 | 69.21 | 0.73 |
| 0.81 | 0.05 | 70.44 | 0.74 |
| 0.80 | 0.05 | 70.69 | 0.74 |
| 0.79 | 0.05 | 71.43 | 0.74 |
| 0.78 | 0.05 | 72.41 | 0.75 |
| 0.77 | 0.05 | 73.15 | 0.76 |
| 0.76 | 0.05 | 73.89 | 0.76 |
| 0.74 | 0.05 | 74.38 | 0.77 |
| 0.73 | 0.05 | 74.63 | 0.77 |
| 0.72 | 0.05 | 75.12 | 0.78 |
| 0.71 | 0.07 | 75.12 | 0.77 |
| 0.70 | 0.07 | 75.86 | 0.78 |
| 0.69 | 0.07 | 75.86 | 0.78 |
| 0.68 | 0.07 | 75.86 | 0.78 |
| 0.67 | 0.10 | 76.35 | 0.78 |
| 0.66 | 0.12 | 76.60 | 0.79 |
| 0.65 | 0.12 | 77.09 | 0.79 |
| 0.64 | 0.12 | 77.83 | 0.80 |
| 0.63 | 0.15 | 77.83 | 0.80 |
| 0.62 | 0.15 | 77.83 | 0.80 |
| 0.61 | 0.15 | 78.57 | 0.80 |
| 0.60 | 0.15 | 78.57 | 0.80 |
| 0.59 | 0.15 | 79.06 | 0.81 |
| 0.57 | 0.15 | 79.31 | 0.81 |
| 0.56 | 0.15 | 79.31 | 0.81 |
| 0.55 | 0.17 | 80.05 | 0.81 |
| 0.54 | 0.20 | 80.54 | 0.82 |
| 0.53 | 0.20 | 81.03 | 0.82 |
| 0.52 | 0.20 | 81.28 | 0.83 |
| 0.51 | 0.22 | 81.77 | 0.83 |
| 0.50 | 0.22 | 82.51 | 0.84 |
| 0.49 | 0.22 | 82.51 | 0.84 |
| 0.48 | 0.22 | 83.00 | 0.84 |
| 0.47 | 0.22 | 83.25 | 0.84 |
| 0.46 | 0.22 | 83.25 | 0.84 |
| 0.45 | 0.22 | 83.25 | 0.84 |
| 0.44 | 0.22 | 83.74 | 0.85 |
| 0.43 | 0.22 | 84.24 | 0.85 |
| 0.41 | 0.22 | 84.48 | 0.85 |
| 0.40 | 0.22 | 84.73 | 0.85 |
| 0.39 | 0.25 | 84.73 | 0.85 |
| 0.38 | 0.25 | 85.22 | 0.86 |
| 0.37 | 0.25 | 85.47 | 0.86 |
| 0.36 | 0.25 | 85.96 | 0.87 |
| 0.35 | 0.27 | 86.21 | 0.87 |
| 0.34 | 0.32 | 86.21 | 0.87 |
| 0.33 | 0.39 | 87.19 | 0.87 |
| 0.32 | 0.47 | 87.19 | 0.87 |
| 0.31 | 0.47 | 87.44 | 0.88 |
| 0.30 | 0.49 | 88.18 | 0.88 |
| 0.29 | 0.52 | 88.42 | 0.88 |
| 0.28 | 0.54 | 88.92 | 0.89 |
| 0.27 | 0.57 | 88.92 | 0.89 |
| 0.26 | 0.64 | 89.66 | 0.89 |
| 0.24 | 0.64 | 89.66 | 0.89 |
| 0.23 | 0.67 | 90.15 | 0.90 |
| 0.22 | 0.69 | 90.15 | 0.90 |
| 0.21 | 0.71 | 90.89 | 0.90 |
| 0.20 | 0.71 | 91.38 | 0.91 |
| 0.19 | 0.74 | 91.38 | 0.91 |
| 0.18 | 0.81 | 91.38 | 0.91 |
| 0.17 | 0.96 | 91.38 | 0.91 |
| 0.16 | 1.03 | 91.38 | 0.91 |
| 0.15 | 1.11 | 92.12 | 0.91 |
| 0.14 | 1.28 | 92.12 | 0.91 |
| 0.13 | 1.35 | 92.36 | 0.91 |
| 0.12 | 1.40 | 92.61 | 0.91 |
| 0.11 | 1.50 | 93.10 | 0.92 |
| 0.10 | 1.65 | 93.60 | 0.92 |
| 0.09 | 1.82 | 94.09 | 0.92 |
| 0.07 | 1.95 | 94.09 | 0.92 |
| 0.06 | 2.22 | 94.33 | 0.92 |
| 0.05 | 2.59 | 94.58 | 0.92 |
| 0.04 | 3.03 | 95.32 | 0.92 |
| 0.03 | 3.87 | 95.57 | 0.92 |
| 0.02 | 5.02 | 96.80 | 0.92 |
| 0.01 | 7.29 | 97.54 | 0.90 |
| 0.00 | 100.00 | 100.00 | 0.00 |

Referencias

- FP \leq 1%
- SCCs más altos
- SCCs más altos y FP \leq 1%

Tabla 4.17: Red neuronal - Porcentajes de TP, FP y SCCs para distintos umbrales en el conjunto de entrenamiento, ordenados ascendentemente por FP. Se resaltaron los resultados que tienen un porcentaje de falsos positivos menor al 1%, los que tienen los SCCs más altos, y la intersección de ambos.

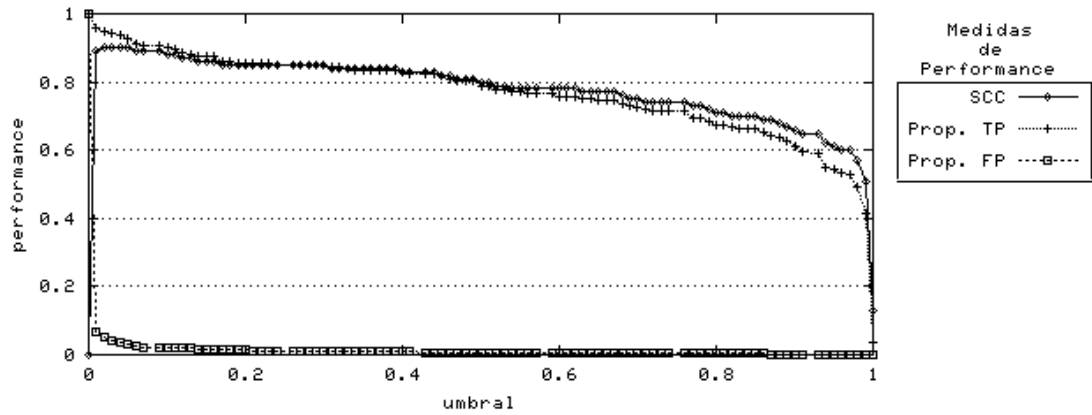


Figura 4.8: Red neuronal - SCCs y proporciones de TP y FP para los distintos umbrales con el conjunto de test.

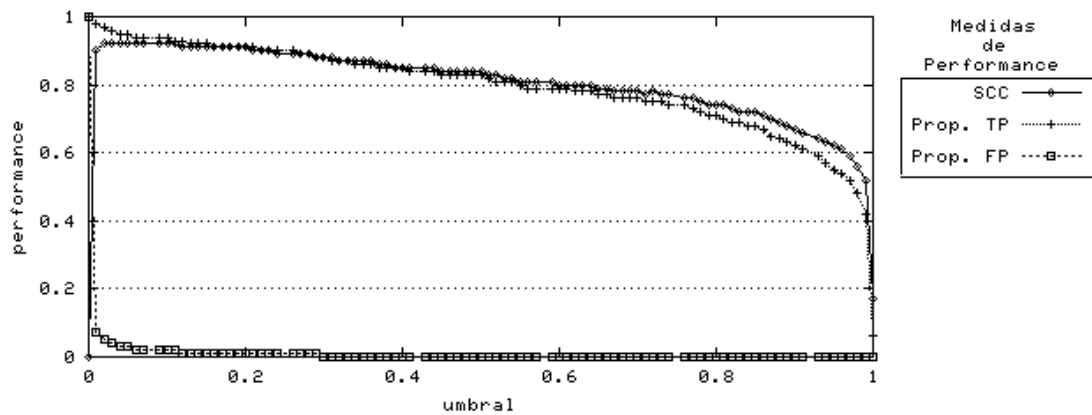


Figura 4.9: Red neuronal - SCCs y proporciones de TP y FP para los distintos umbrales con el conjunto de entrenamiento.

En la Tabla 4.18 se puede ver un resumen de los resultados para los conjuntos de entrenamiento y test.

| Cjto. | 0% FP | | 1% FP | | 2% FP | | 5% FP | | 100% TP | | 80% TP | | máximo SCC | | |
|-------|-------|------|-------|------|-------|------|-------|------|---------|------|--------|------|------------|------|-------|
| | %TP | SCC | %TP | SCC | %TP | SCC | %TP | SCC | %FP | SCC | %FP | SCC | SCC | %FP | %TP |
| test | 25.29 | 0.38 | 85.06 | 0.85 | 89.66 | 0.88 | 94.83 | 0.90 | 51.47 | 0.57 | 0.69 | 0.81 | 0.90 | 2.90 | 92.53 |
| entr. | 50 | 0.58 | 91.38 | 0.91 | 94.09 | 0.92 | 96.80 | 0.92 | 51.13 | 0.57 | 0.17 | 0.81 | 0.92 | 1.50 | 93.10 |

Tabla 4.18: Resumen de los resultados de la red neuronal con los conjuntos de entrenamiento y test para 0%, 1%, 5% de FP y 80% y 100% de TP.

4.4.3. Análisis de resultados

El SCC con el conjunto de test llega a valores altos con un máximo de 0.9 (Tabla 4.16 y Figura 4.8). La cantidad de resultados verdaderos positivos es muy alta y la de falsos positivos muy baja.

Se analizaron los pesos resultantes del reentrenamiento y los resultados fueron muy parecidos a los descriptos en el Capítulo 3 (Figura 3.11).

4.5. Comparación de los distintos métodos

A continuación mostraremos y analizaremos los resultados de la aplicación de los distintos métodos tratados en este capítulo.

4.5.1. Resultados de los distintos métodos

En la Tabla 4.19 se puede ver una comparación de los resultados de los métodos CPR y Consensus-Patser con el conjunto de test. Para cada uno de los métodos se muestra el %TP y el SCC con 0 % 1 % y 5 % de falsos positivos. También se muestran el %FP y el SCC con 100 % y 80 % de verdaderos positivos. Se eligieron estos números estimando que pueden adecuarse a las distintas necesidades de los usuarios de estos datos.

| | 0 %FP | | 1 % FP | | 2 % FP | | 5 % FP | | 100 %TP | | 80 %TP | |
|------------|--------------|-------------|--------------|-------------|--------------|-------------|--------------|-------------|--------------|-------------|-------------|-------------|
| | %TP | SCC | %TP | SCC | %TP | SCC | %TP | SCC | %FP | SCC | %FP | SCC |
| CPR | 25.29 | 0.38 | 85.06 | 0.85 | 89.66 | 0.88 | 94.83 | 0.90 | 51.47 | 0.57 | 0.69 | 0.81 |
| C-P | 4.6 | 0.15 | 37.93 | 0.47 | 46.55 | 0.52 | 64.37 | 0.62 | 88.39 | 0.25 | 13.86 | 0.66 |

Tabla 4.19: Resultados de la aplicación de los métodos CPR y Consensus-Patser (C-P) al conjunto de test para 0 %, 1 %, 5 % de FP y 80 % y 100 % de TP.

Allí se puede observar que la red tiene mejores resultados que Consensus-Patser. Para los casos en que están fijos los porcentajes de falsos positivos, la cantidad de verdaderos positivos fue más alta con CPR. En los casos en que los porcentajes de los verdaderos positivos están fijos CPR arrojó menos falsos positivos que el método Consensus-Patser. Estos resultados implican que el SCC también sea más alto en los resultados de CPR, lo cuál también puede verse en la tabla.

En la Tabla 4.20 se muestran los máximos SCCs obtenidos por cada uno de los métodos con el conjunto de test y el porcentaje de FP y de TP.

| Método | Máximo SCC | %FP | %TP |
|--------|------------|-------|-------|
| CPR | 0.90 | 2.90 | 92.53 |
| C-P | 0.67 | 15.06 | 81.61 |
| MF | 0.33 | 2.69 | 25.86 |

Tabla 4.20: Máximos SCCs alcanzados por los métodos CPR, Consensus-Patser (C-P) y métodos fijos (MF).

Como puede observarse, el SCC obtenido con el método CPR es más alto que el obtenido con los otros dos métodos. Con el método CPR se reconoce el 93.97% de los promotores de los conjuntos de entrenamiento y test. Con C-P se reconoce el 86.12% y con los métodos fijos el 24.82%. Estos resultados se obtuvieron utilizando para CPR y C-P los umbrales correspondientes a los mayores SCCs obtenidos con el conjunto de test (Tabla 4.20). Para los métodos fijos se utilizaron las condiciones que arrojan el mayor SCC (Tabla 4.20).

En la Sección 4.3 observamos que en todas las secuencias consenso mostradas, los consensos TATAAT están precedidos por el consenso TGG y sucedidos por el consenso GCA. Además, en la Sección 3.3 habíamos explicado que a partir de la observación de los pesos resultantes del entrenamiento de la RN, se podía inferir que entre los pesos de la entrada y una de las capas se representaba la subsecuencia TGGTATAA (i.e. la entrada TGGTATAA es la que activa más fuertemente a las neuronas de dicha capa, ver Figura 3.11)⁷. También habíamos remarcado que el consenso TATAAT mostrado por Harley y Reynolds estaba precedido por el consenso GG y sucedido por el consenso G, en ambos casos con menor conservación que el consenso conocido. Resulta interesante ver, entonces, que con los dos métodos estudiados encontramos que el consenso puede ser un poco mayor al conocido.

4.5.2. Análisis de resultados de los distintos métodos

Una ventaja que tiene el CPR sobre los otros métodos es la posibilidad de búsqueda de los dos motivos conservados sin contar con el conocimiento de la distancia que los separa, basándose en los ejemplos seleccionados para la realización del entrenamiento. Consensus, en cambio genera una matriz de alineamiento, en donde la distancia entre las secuencias conservadas es fija. Una forma de permitir la variación de distancias entre ambas secuencias es generando matrices de alineamiento para cada uno de los motivos con Consensus y luego ejecutando Patser, tomando como matrices de alineamiento a todas las generadas a partir de los motivos separados a las distancias que se crean apropiadas. Por ejemplo, si se generó la matriz M_1 de 4×6 para

⁷Estos resultados coinciden con los obtenidos luego del reentrenamiento con datos de RegulonDB.

TTGACA y la matriz M_2 de 4×6 para TTGACA, ejecutar Patser con una matriz M_4 , armada a partir de juntar las matrices M_1 , una matriz M_3 y M_2 . La matriz M_3 tendría 4 filas y la cantidad de columnas tendría que estar entre 15 y 21. De esta forma la matriz M_4 tendría 4 filas y la cantidad de columnas estaría entre 27 y 33. De esta forma, se evitaría además “dejar fijos” los nucleótidos que se encuentran entre las dos subsecuencias. Huerta y Collado-Vides hacen esto, sin embargo los resultados no mejoran significativamente [34]. La variabilidad de distancia sí puede ser tratada con los métodos fijos utilizados, siempre especificando entre qué límites se encuentra (en este caso éstos fueron 15 y 21), la red es más flexible en este sentido, ya que no existe la necesidad de especificar una longitud determinada.

Una limitación importante de los métodos fijos utilizados radica en que a todos los nucleótidos del patrón elegido se les da igual peso, sólo se consideran los nucleótidos más frecuentes y se buscan las cadenas que contienen los patrones con todas las letras del consenso, que tienen probabilidad baja de ser encontrados. Esto nos lleva a pensar en la pérdida de información que conlleva la generación de una secuencia consenso y vemos aquí las limitaciones en el uso de la misma. Hay posibilidades de hacer mejoras en estos métodos, asignando distintos pesos a los distintos nucleótidos de cada patrón a partir del conocimiento que se tiene actualmente de los mismos. Sin embargo el diseño de un modelo que considere esto y la distancia variable entre patrones tampoco es el ideal, ya que depende de datos probabilísticos y no se tiene un modelo que describa claramente el problema. Esto nos hace ver nuevamente las ventajas de contar con una red neuronal, con la que sin tener un modelo preciso del problema es posible en algunos casos llegar a buenas soluciones. En muchos casos no es posible o no conviene construir este modelo por falta de conocimiento del problema o por ser éste muy complejo.

Capítulo 5

CPR-MOSS: una metodología de dos niveles para la identificación de múltiples promotores

El método Conexionist Promoter Recognition (CPR) arroja soluciones precisas, pero no permite individualizar fácilmente las subsecuencias correspondientes a las regiones -10 y -35 que componen el promotor. Más de un promotor puede intervenir en diferentes procesos de regulación en una misma región y todos son posibles candidatos hasta la realización de mutaciones dirigidas. La identificación de las subsecuencias correspondientes a las regiones -10 y -35 es útil debido a que permite detectar los posibles promotores en una misma región y realizar estas experimentaciones.

Generalmente, como puede verse en la compilación de Harley y Reynolds, en que para algunos promotores hay más de una combinación TTGACA-TATAAT marcada¹, es posible encontrar más de una solución. Se podría, entonces realizar una búsqueda de patrones similares a TATAAT y TTGACA dentro de un promotor y preferir aquéllos que se encuentran a una distancia de entre 15 y 21 pb. Huerta y Collado-Vides [34] utilizan Consensus y Patser para buscar los motivos TATAAT y TTGACA y luego buscan promotores haciendo combinaciones de las matrices, que representan a los motivos, separadas por todas las posibles distancias. Además de la necesidad de búsqueda exhaustiva para todas las posibles distancias que separan a las subsecuencias, surge la necesidad de elegir entre dos distintas soluciones con objetivos que no son fácilmente comparables. Por ejemplo, una solución puede tener alto grado de *matching* con la secuencia TATAAT, pero bajo grado de *matching* con TTGACA y mala distancia entre los dos (por ej. 21 pb), y otra solución puede tener buen *matching*

¹Con *combinación TTGACA-TATAAT* ó *combinación TTGACA-distancia-TATAAT* nos referimos a secuencias con un cierto parecido a los consensos TTGACA-TATAAT separados por alguna distancia válida (entre 15 y 21 pb).

de TTGACA, pero malo de TATAAT y distancia óptima (17 pb). En el trabajo mencionado falta una metodología para la ponderación de los objetivos. Este problema se puede ver como un problema de múltiples objetivos, que se traducen en optimizar un grado de *matching* entre los motivos TATAAT y TTGACA y las posibles distancias entre los mismos.

En este capítulo presentamos una metodología de dos niveles, el primero es un método de búsqueda global y el segundo realiza una búsqueda local. Denominamos a esta metodología CPR-MOSS, a partir de las siglas de *Conexionist promoter Recognition-Multiobjective Scatter Search*.

En la Sección 5.1 presentamos el problema que nos planteamos en este capítulo, en la Sección 5.2 exponemos la metodología de dos niveles propuesta. En la Sección 5.3 explicamos los detalles de la experimentación realizada y en la Sección 5.4 mostramos y analizamos los resultados obtenidos. Por último, en la Sección 5.5 comentamos algunos detalles de la aplicación desarrollada.

5.1. Problema

En los capítulos anteriores hemos presentado un método para la predicción de secuencias promotoras en procariotas construida con redes neuronales artificiales con resultados precisos. Sin embargo, además de conocer la ubicación del promotor en una secuencia de ADN, también resulta útil poder identificar las secuencias conservadas en cada uno de los promotores predichos. Por ejemplo, en la Figura 5.1 se pueden ver dos posibles combinaciones TTGACA-TATAAT subrayadas para el promotor *colE1-B*: la marcada como principal (bajo las leyendas *región -35* y *región -10*) a una distancia de 15 pb y los alternativos a 16 pb de distancia. La red neuronal encuentra este promotor, pero no define cuáles son las subsecuencias que se corresponden con las regiones TATAAT y TTGACA ni a qué distancia se encuentran.

región -35
región -10
colE1-B TTATAAAATCCTCT TTGACT TTTAAAA CAATAAGT TAAAAA TAAATACTGTAA

Figura 5.1: Promotor *colE1-B* y las regiones TATAAT y TTGACA originales y alternativas marcadas por Harley y Reynolds. Las originales son las que están bajo las etiquetas *región -35* y *región -10*.

En el Capítulo 3 hicimos un breve análisis de los pesos resultantes del entrenamiento de la red, mediante el cuál detectamos la representación en los pesos de una de las secuencias conservadas. Sin embargo, para conocer la ubicación de las secuencias

conservadas habría que considerar las posibles distancias entre las mismas, y entre todas las posibilidades habría que elegir las mejores combinaciones. Esta tarea exige un nuevo procesamiento de los datos.

Para realizar este procesamiento utilizamos un algoritmo evolutivo multiobjetivo, que a partir de ciertas decisiones fundamentadas en los resultados de investigaciones anteriores [26, 25, 42] elige combinaciones TTGACA-distancia-TATAAT haciendo una optimización heurística de los tres objetivos.

Una característica de este algoritmo es que necesita contar con una especificación clara del problema a resolver (se necesita saber qué objetivos se quieren optimizar). Esto, como ya se explicó, no es sencillo en la problemática que estamos tratando. Por otro lado, estos métodos tienen el problema de tener baja performance, que decrece a medida que el tamaño de la secuencia de entrada crece. Recordemos que la red neuronal no precisa contar con un modelo del problema a resolver y su tiempo de ejecución, una vez entrenada, es muy bajo.

5.2. Metodología CPR-MOSS

Para resolver el problema planteado en la sección anterior proponemos una metodología de dos niveles: CPR-MOSS [14]. El primer nivel está formado por el método CPR, que predice la presencia y ausencia de promotores y el segundo nivel por un algoritmo evolutivo, MOSS [69], que busca determinados patrones específicos en los promotores predichos por la red neuronal. Un esquema de la metodología de dos niveles puede verse en la Figura 5.2.

El uso de la metodología propuesta, nos permite aprovechar las fortalezas de cada método, obteniendo resultados más completos que con el método CPR sólo, con una performance en tiempo razonable. También evitamos la dependencia del modelo del método MOSS.

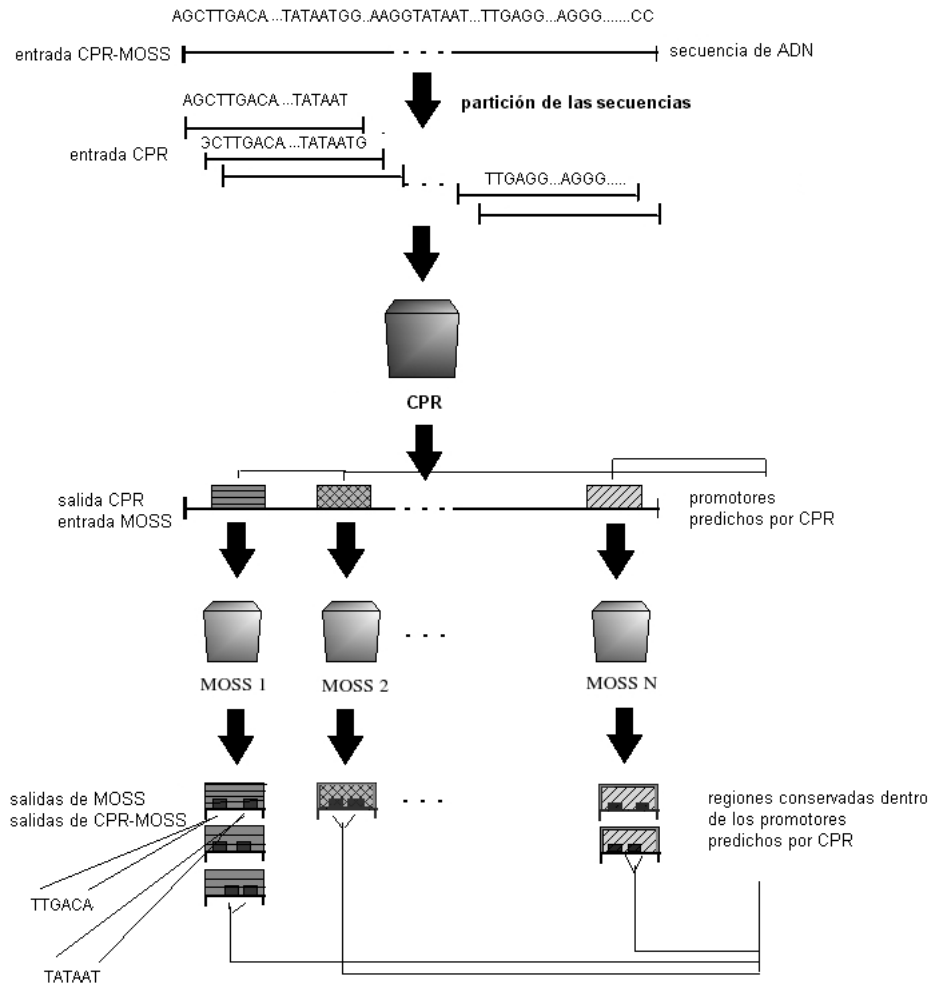


Figura 5.2: Esquema de la metodología CPR-MOSS - La entrada es una secuencia de ADN y la salida es el conjunto de promotores predichos con la locación de sus secuencias conservadas dentro de cada promotor. Primero, se parte la secuencia de ADN en subsecuencias de longitud fija. Cada una de esas subsecuencias se utiliza como una entrada de la red neuronal CPR. La salida de la red neuronal es el conjunto de promotores predichos. Cada una de las predicciones de la red neuronal es usada como entrada para el algoritmo evolutivo MOSS, que devuelve un conjunto de posibles ubicaciones de las secuencias conservadas.

5.2.1. Introducción a conceptos de algoritmos evolutivos y optimización multiobjetivo

Antes de presentar el método utilizado para la detección de los patrones fuertemente conservados a distancias variables dentro de una secuencia de ADN, haremos una introducción a conceptos de algoritmos evolutivos y de problemas multiobjetivo.

Algoritmos evolutivos

Las metaheurísticas son procesos iterativos que guían a una heurística, combinando distintos conceptos para explorar el espacio de búsqueda. La *Computación Evolutiva* es una técnica *metaheurística* que se basa en el empleo de modelos de procesos evolutivos para el diseño e implementación de soluciones a problemas de optimización. Los distintos modelos computacionales que se han propuesto dentro de esta filosofía suelen recibir el nombre genérico de *Algoritmos Evolutivos*. Estos algoritmos son una analogía de la teoría darwiniana de la evolución de las especies y toman esta idea para buscar una o más soluciones óptimas entre un conjunto de posibles soluciones.

La *Teoría de la Evolución* explica el origen y la transformación de los seres vivos como el producto de la acción de dos principios fundamentales: la selección natural y el azar. La selección natural regula la variabilidad de la recombinación y mutación aleatorias de los genes: toda la variedad que observamos en la naturaleza viviente se basa en la capacidad de los seres vivos de producir copias de sí mismos, en que el proceso de reproducción actualiza muchas variantes, y en que, en la interacción con el ambiente, algunas de ellas son seleccionadas para sobrevivir y producir las copias subsiguientes [20].

Un algoritmo evolutivo (AE) es un algoritmo probabilístico que mantiene una **población** de individuos, $P(t) = \{x_1^t, \dots, x_n^t\}$ para la iteración t . Cada individuo representa una solución potencial del problema en cuestión y es representado por una estructura de datos S . Cada solución x_i^t es evaluada para dar cierta medida de su aptitud. Luego se forma una nueva población (iteración $t + 1$), seleccionando los individuos más aptos (**paso de selección**). Algunos miembros de la nueva población sufren transformaciones (**paso de transformación**) mediante operadores genéticos para formar nuevas soluciones. Existen transformaciones unarias m_i denominadas **mutaciones**, que crean individuos a partir de un pequeño cambio en un individuo ($m_i : S \rightarrow S$) y transformaciones c_j , denominadas de **cruce** (*crossover*), que crean individuos nuevos combinando partes de uno o más individuos ($c_j : S \times \dots \times S \rightarrow S$). Después de un cierto número de iteraciones el algoritmo converge, se espera que el

mejor individuo represente una solución cercana al óptimo [48]. En la Figura 5.3 puede verse el diagrama de flujo de un AE básico.

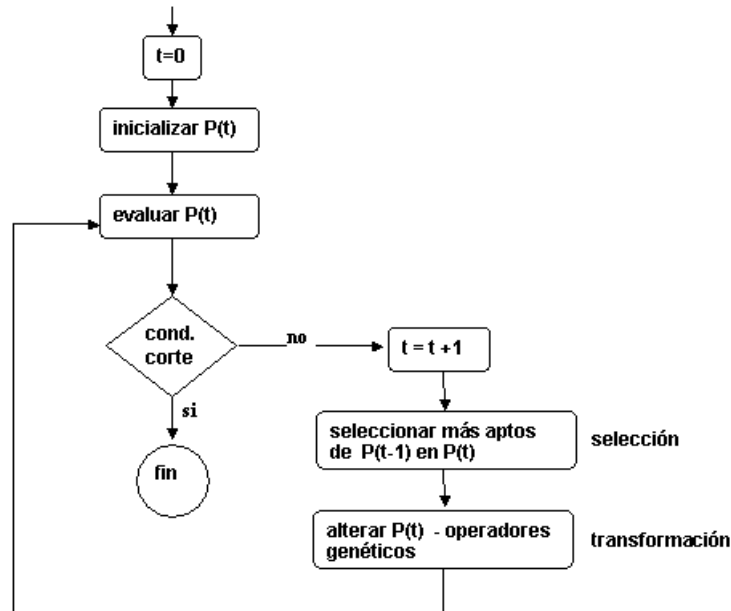


Figura 5.3: Diagrama de flujo de un algoritmo evolutivo básico.

Cada individuo de la población es llamado **cromosoma**. A su vez se conoce con el nombre de **gen** a cada porción del cromosoma que tiene un significado. Se denomina **generación** a cada iteración del algoritmo.

De esta forma, los AEs exploran un espacio de soluciones candidatas en busca una solución que optimice una cierta función relevante para el problema tratado. A esta función se la conoce con el nombre de o función de aptitud o función de *fitness*.

Optimización multiobjetivo

La búsqueda de los patrones conservados en una secuencia, puede arrojar varias soluciones satisfactorias. Algunas serán mejores que otras en uno de los patrones, pero peores en el otro y la distancia entre los mismos también variará. No es trivial determinar cuál de todas es la mejor solución, dado que ninguno de los objetivos a optimizar (secuencias conservadas y distancia) predomina sobre el otro. El hecho de contar con muchos objetivos a optimizar, sin la posibilidad de determinar –entre las soluciones en que distintos objetivos fueron optimizados– cuál es la mejor, nos permite ver al problema de reconocimiento de patrones en secuencias promotoras como un problema de optimización *multiobjetivo*. Por otro lado, la existencia de más de una

solución, determina que la solución del problema planteado es además *multimodal*. A continuación introduciremos algunos conceptos necesarios [15]:

Definición 1 *Un problema de optimización multiobjetivo se define de la siguiente manera:*

$$\left. \begin{aligned} f_m(x), \quad m = 1, 2, \dots, M; \\ g_j(x) \geq 0, \quad j = 1, 2, \dots, J; \\ h_k(x) = 0, \quad k = 1, 2, \dots, K; \\ x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n, \end{aligned} \right\}$$

donde f_m son las funciones de aptitud a maximizar o minimizar² y M corresponde al número de objetivos que tiene el problema, de esta forma hay una función por objetivo. A estas funciones también se las denomina funciones objetivo. g_j son restricciones de desigualdad, h_k son restricciones de igualdad y n es el número de variables de decisión. Una solución x está formada por un vector de n variables de decisión: $x = (x_1, x_2, \dots, x_n)^T$, donde cada una toma un valor entre un mínimo x_i^L y un máximo x_i^U .

Las soluciones x posibles, son aquellas que satisfacen todas las $j + k$ restricciones y los $2n$ límites de las variables de decisión. Con el objetivo de poder comparar dos soluciones distintas definimos el concepto de *dominancia*, que utilizaremos más adelante para comparar soluciones. $f_i(x) \triangleleft f_i(y)$ denota que la solución x es mejor que la solución y en el objetivo i y $f_i(x) \triangleright f_i(y)$ denota que la solución x es peor que la solución y en el mismo objetivo. Por ejemplo, si se quiere minimizar la función objetivo, el operador \triangleleft equivale al operador ' $<$ ' y si se la quiere maximizar equivale a ' $>$ '.

Definición 2 *Se dice que una solución x domina a una solución y si y sólo si se cumplen las siguientes condiciones:*

1. $\forall j : 1 \leq j \leq M : f_j(x) \not\triangleright f_j(y)$ (la solución x no es peor que la solución y en ningún objetivo).
2. $\exists j : 1 \leq j \leq M : f_j(x) \triangleleft f_j(y)$ (la solución x es mejor que la solución y en al menos un objetivo).

Si la solución x domina a la y , también se puede escribir que x es *no-dominada* por y . La relación de dominancia no es reflexiva, simétrica ni antisimétrica y es transitiva (es una relación de orden parcial estricto [15]). Por otro lado, el que la solución p no domine a la solución q no implica que q domine a p .

²Nótese que los problemas de maximización pueden convertirse en problemas de minimización multiplicando la función objetivo por -1 .

En la Figura 5.4 se puede ver un ejemplo de un problema de optimización de dos objetivos con cinco soluciones diferentes. Se asume que f_1 debe ser maximizada y f_2 minimizada. Usando la Definición 2 es posible determinar qué soluciones son mejores que otras en términos de los dos objetivos. Por ejemplo, la solución 1 es mejor que la 2 en ambos objetivos, por lo tanto la solución 1 domina a la 2. La solución 5 domina a la 1. Se puede ver también que ni la solución 3 domina a la 5, ni la 5 domina a la 3, lo que nos lleva a hacer una tercera definición.

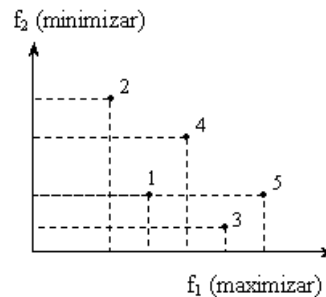


Figura 5.4: Población con cinco soluciones.

Definición 3 Entre un conjunto de soluciones P , el conjunto de soluciones no-dominadas, P' , es aquel formado por todas las soluciones no dominadas por ningún elemento de P .

En el ejemplo anterior, las soluciones 3 y 5 constituyen el conjunto de soluciones no-dominadas.

5.2.2. MOSS

Se realizó una adaptación de la búsqueda dispersa (*scatter search*) en un problema multiobjetivo.

Búsqueda dispersa. El algoritmo de búsqueda dispersa [46] es una metaheurística basada en la idea de combinar las soluciones guardadas en el conjunto conocido como *Conjunto de Referencia* (RefSet) y aplicar *búsqueda local* a los individuos resultantes. Este conjunto acumula las *buenas* soluciones encontradas durante el proceso de búsqueda, desde el punto de vista de la calidad y de la diversidad.

Las técnicas de búsqueda dispersa, como otros AEs mantienen una población de soluciones potenciales (también denominadas *puntos de referencia*). Se arman subconjuntos de puntos de referencia que generan descendencia (también denominada

puntos de prueba) mediante métodos de combinación y se seleccionan los mejores miembros para la nueva población (o conjuntos de referencia).

Enfoque multiobjetivo. Con el fin de definir el problema de optimización multiobjetivo instanciamos la Definición 1. 1) $M = 3$. Hay tres objetivos: f_1 es la función objetivo para la región -35 , f_2 es la función objetivo para la región -10 y f_3 corresponde a la distancia entre ambas regiones (ver Ecuaciones 5.1 y 5.6). 2) $J = 2$. Tenemos dos restricciones g_1 y g_2 : la distancia entre las regiones no puede ser menor que 15, ni mayor que 21 pb. 3) $K = 0$. No hay restricciones de igualdad. 4) Sólo se mantienen soluciones válidas en cada generación y las regiones -10 y -35 tienen que estar ubicadas dentro de la secuencia buscada.

La función de aptitud de ambos patrones se calculó utilizando la información existente acerca de la conservación de los nucleótidos. La imagen de esta función pertenece al intervalo $[0,1]$, 0 corresponde al peor resultado y 1 al mejor.

$$f_1 = f_2 = \frac{\text{amount} - \text{min}}{\text{max} - \text{min}} \quad (5.1)$$

Sean $observedseq_i$ el nucleótido presente en la posición i de una secuencia, $knownseq_i$ el nucleótido más frecuentemente observado en la posición i de una secuencia y $observedfreq_i$ y $knownfreq_i$ sus respectivas frecuencias de aparición, calculamos:

$$\text{max} = \frac{\sum_{i=1}^6 \text{knownfreq}_i}{100} \quad (5.2)$$

$$\text{min} = \frac{\sum_{i=1}^6 (100 - \text{knownfreq}_i)/3}{100} \quad (5.3)$$

$$\text{cost}(a, b) = \begin{cases} \text{knownfreq}_i/100 & a = b \\ (100 - \text{knownfreq}_i)/300 & a \neq b \end{cases} \quad (5.4)$$

$$\text{amount} = \sum_{i=1}^6 \text{cost}(\text{observedseq}_i, \text{knownseq}_i) \quad (5.5)$$

La distancia entre las secuencias consenso es medida como una función triangular [37, 57] con centro en el número 17. En este punto toma el valor máximo y a la derecha e izquierda de este punto el valor de la función decrece, como puede verse en la Ecuación 5.6 y la Figura 5.5.

$$f_3 = \begin{cases} -0,25 \times \text{distance} + 5,25 & 17 \leq \text{distance} < 21 \\ 0,5 \times \text{distance} - 7,5 & 15 < \text{distance} < 17 \\ 0 & \text{en otro caso,} \end{cases} \quad (5.6)$$

donde *distance* representa la distancia entre ambos consensos.



Figura 5.5: Gráfico de la función objetivo f_3 .

Componentes: Operador de combinación y búsqueda local

Cada individuo es representado mediante **bloques**. Cada bloque corresponde a una de las regiones conservadas del promotor, por lo tanto, tendremos dos bloques por individuo. Un bloque se representa mediante dos números enteros que definen la posición de la secuencia en la que se encuentra el mismo. El primer número representa el nucleótido inicial en donde se encuentra el bloque en la secuencia objetivo (aquella en la que se está buscando), mientras que el segundo número representa el tamaño del bloque (ver Figura 5.6). El **operador de combinación** consiste en un *cruce*. Este

Fenotipo

```

          ttgaca                tataat
gtttatTTtaatgTTtaccCCCataaccacataatcgCGttacact
  ↑                ↑
posición 6        posición 29
    
```

Genotipo

```

Gen 0      Gen 1
[(6,6)]   [(29,6)]
f1 = 0.578  f2 = 0.802  f3 = 1.000
    
```

Figura 5.6: Ejemplo para la representación de un individuo, considerando los datos de la compilación de Lisser y Margalit

cruce fue implementado como un operador de cruce simple de un punto. El punto está siempre ubicado en medio de los dos bloques que forman el cromosoma. De esta forma los cromosomas padres de dos bloques A_1B_1 y A_2B_2 producirán como descendencia a A_1B_2 y a A_2B_1 . La **búsqueda local** implementada para este problema consiste en la búsqueda de soluciones no-dominadas entre las soluciones vecinas. Para realizarla se muta una cantidad determinada de nucleótidos a la izquierda o a la derecha de

uno de los bloques que forman el cromosoma. Si el nuevo cromosoma domina a su padre, entonces el viejo cromosoma es reemplazado por el nuevo. Si el padre domina al cromosoma mutado, no se realiza ninguna modificación, y, en caso contrario se compara el nuevo cromosoma con la población no-dominada encontrada hasta el momento. Si no es dominada por ninguna reemplaza al padre con una probabilidad de 0.5. Se puede ver el procedimiento descrito en la Figura 5.7.

Entrada:
c: cromosoma
t: cantidad de nucleótidos vecinos

- 1: Seleccionar aleatoriamente qué bloque *g* será mutado en el cromosoma *c*.
- 2: Seleccionar aleatoriamente un número $n \in [-t, t]$ y mover el bloque *g*, *n* nucleótidos (dado que *n* puede ser negativo o positivo, el bloque puede moverse aguas arriba o aguas abajo). El bloque resultante es *g'* y el individuo resultante *c'*.
- 3: **if** *c'* satisface las *restricciones* **then** {restricciones de distancia entre los dos bloques}
- 4: **if** *c'* domina a *c* **then**
- 5: Reemplazar *c* con *c'*
- 6: **fin if**
- 7: **if** *c'* no-domina a *c*, *c'* no es dominado por *c* y no hay ninguna solución del conjunto de soluciones no-dominadas que domina a *c'* **then**
- 8: Reemplazar *c* con *c'* con probabilidad de 0.5.
- 9: **fin if**
- 10: **fin if**

Figura 5.7: Búsqueda local.

Implementación del algoritmo MOSS

Como mencionamos anteriormente, el algoritmo MOSS es una adecuación del algoritmo original de búsqueda dispersa para que funcione con más de un objetivo. En el algoritmo original, uno de los pasos consiste en el ordenamiento de las soluciones de acuerdo a la función de aptitud. En este caso tenemos más de una función objetivo, con lo cual no siempre es sencillo determinar que una solución es mejor que la otra (no-dominancia). Para solucionar este problema, se consideran por separado las soluciones dominadas y las soluciones no-dominadas. Las soluciones no-dominadas se agregan al conjunto de referencia en cualquier orden, mientras que las soluciones dominadas se agregan al final si no hay más soluciones no-dominadas que incorporar, nuevamente en cualquier orden.

Un conjunto de individuos no-dominados es también utilizado para almacenar todas aquellas soluciones no-dominadas que han sido encontradas durante el proceso de búsqueda. La Figura 5.8 muestra el algoritmo MOSS.

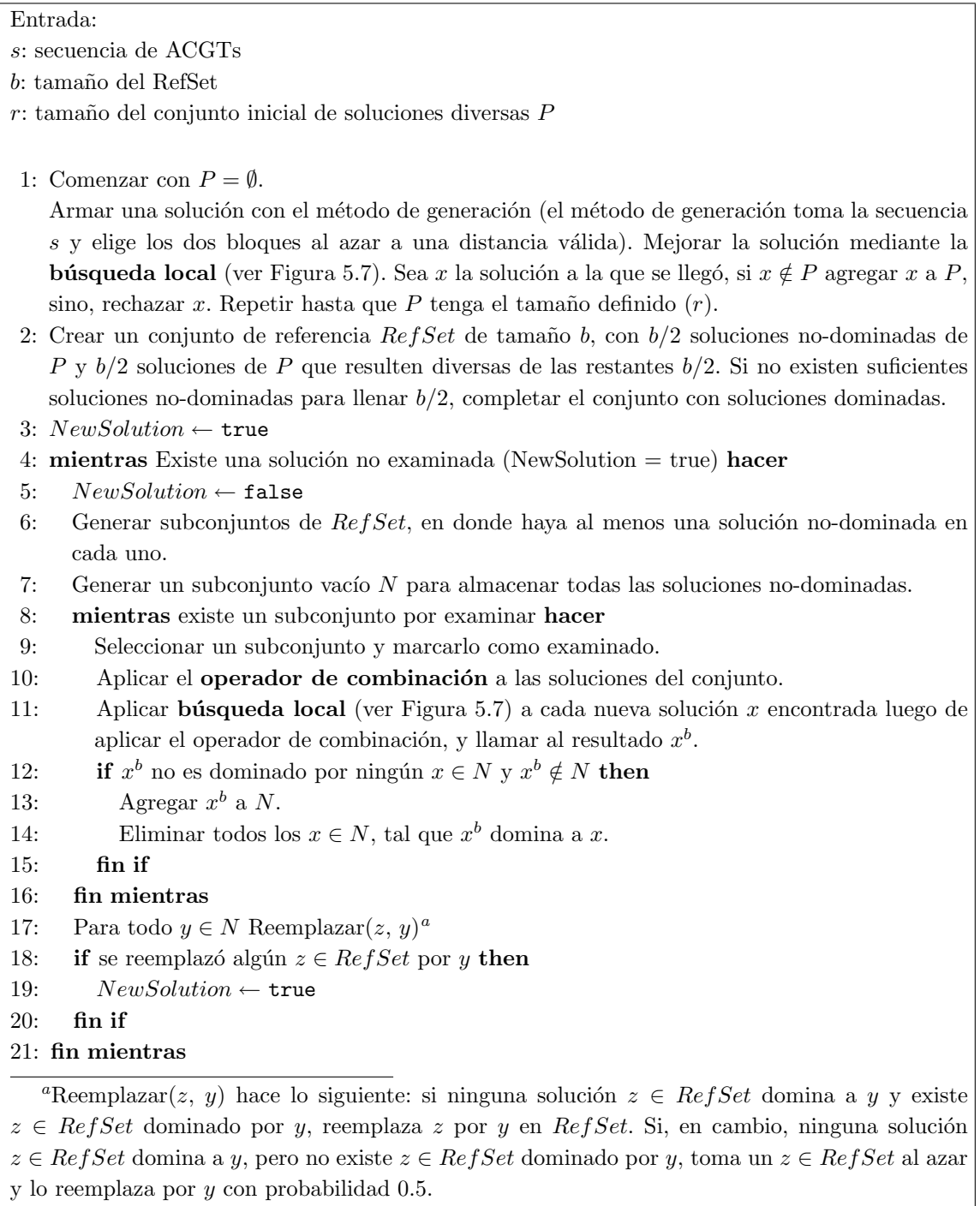


Figura 5.8: Algoritmo MOSS.

5.3. Experimentación

Aplicamos la metodología CPR-MOSS al conjunto de promotores compilado por Harley y Reynolds [25]. Se eligió este estudio, dado que en el mismo están marcadas

las secuencias conservadas TATAAT y TTGACA –originales y alternativas–, que son útiles para mostrar los resultados de nuestro método.

Se utilizó la red resultante del entrenamiento descrito en el Capítulo 3, fijándose el umbral a partir del SCC más alto.

Para cada secuencia de entrada al MOSS se computaron 20 ejecuciones con distintas semillas de generación de números aleatorios. Decimos que encontramos una combinación TTGACA-distancia-TATAAT satisfactoria, si en 10 de las 20 ejecuciones existió tal relación. Cada ejecución fue realizada con los parámetros especificados en la Tabla 5.1. El tamaño máximo del conjunto inicial de soluciones diversas, P, fue de 300 individuos. El tamaño del RefSet fue de 16 individuos. Se ejecutaron a lo sumo 200 iteraciones en cada caso.

| Parámetro | Valor |
|---------------------------------|--------------|
| Cantidad máxima de generaciones | 200 |
| Tamaño del RefSet | 16 |
| Tamaño del conjunto P | 300 |

Tabla 5.1: Parámetros para la ejecución del MOSS.

5.4. Resultados

La red neuronal entrenada con los conjuntos de datos descritos en el Capítulo 3, reconoce 93.01 % de promotores de la compilación de Harley y Reynolds con 4.51 % FP. El umbral utilizado como parámetro de la función de salida de la red es 0.11. Se eligió este valor por ser el de mayor SCC en el conjunto de test.

Para los promotores que fueron reconocidos como tales por la red, tomamos las secuencias conservadas marcadas como originales y alternativas por Harley y Reynolds y verificamos si éstas se encuentran en el conjunto de soluciones del MOSS para dicha secuencia. En la Tabla 5.2 se muestran los resultados obtenidos. Como puede observarse se encontró el 93.10 % de las combinaciones TTGACA-distancia-TATAAT marcadas como originales y el 86.76 % de las marcadas como alternativas. En [14] se muestran los resultados de la aplicación de otros algoritmos multiobjetivo al mismo problema y se concluye que MOSS es el que tiene mejor reconocimiento.

| | Originales | Alternativos | %Originales | %Alternativos | Total | %total |
|------|------------|--------------|-------------|---------------|-------|--------|
| MOSS | 236 | 59 | 93.10 | 86.76 | 295 | 91.90 |

Tabla 5.2: Resultados de la aplicación de la metodología CPR-MOSS a las secuencias de la compilación de Harley y Reynolds. La columna *Original* indica de las combinaciones TTGACA-distancia-TATAAT publicadas en la compilación, la cantidad encontrada por el MOSS. La columna *Alternativos* indica la cantidad de las ubicaciones alternativas publicadas en la compilación que fueron detectadas por la aplicación del MOSS. Sólo se consideraron las secuencias de la compilación de Harley y Reynolds, el CPR predijo como promotores.

En la Figura 5.9, se muestran distintas combinaciones TTGACA-distancia-TATAAT arrojadas por el algoritmo MOSS para el promotor ampC/C16, que fue reconocido por la red.

gctatcttgacagttgtcacgctgattggtatcggtacaaatctaacgtatcg

Figura 5.9: Distintas soluciones para el promotor ampC/C16. Las marcas a la izquierda corresponden a la secuencia TTGACA y las de la derecha a la secuencia TATAAT.

Los resultados obtenidos permiten concluir que la metodología CPR-MOSS propuesta es adecuada para el reconocimiento y análisis de secuencias de promotores en procariotas. Al alto reconocimiento obtenido por el método CPR se le agrega la capacidad del MOSS de encontrar posibles subsecuencias correspondientes a las regiones -35 y -10 de la red a distancias válidas.

5.5. Aplicación Desarrollada

Se implementó la metodología CPR-MOSS (ver Figura 5.2) y se dejó disponible para uso público en el sitio web <http://soar-tools.wustl.edu>. Como entrada se ingresa una secuencia de ADN de longitud mayor o igual a 46 pares de bases y un criterio de búsqueda. La elección de un criterio de búsqueda permite fijar el umbral que determina si una cadena es o no promotora. Las posibles elecciones son un límite superior en el porcentaje de falsos positivos o el máximo SCC. Como salida se devuelven todos los promotores predichos que satisfacen el criterio de búsqueda, y –en el caso en que se desee– las ubicaciones de las secuencias correspondientes a las regiones -35 y -10. La implementación se dividió en tres sencillos módulos:

Conversor este módulo toma la cadena de ADN, la divide en subsecuencias contiguas de longitud 46 y las convierte al formato necesario para pasarlas como entrada a la red neuronal.

CPR este módulo toma como entrada cada una de las subsecuencias de longitud 46 que salen del *Conversor* y un criterio de búsqueda y devuelve una predicción acerca de si la subsecuencia de longitud 46 es o no un promotor para cada una de dichas entradas. Para esto utiliza la red neuronal resultante del entrenamiento con los últimos datos disponibles, que fue descrito en el Capítulo 4. Está implementado en perl utilizando *batchman*, el interpretador de comandos batch provisto por el SNNS. Se confeccionó un sencillo programa que presenta cada uno de los conjuntos de entradas preparados por el *Conversor* a la red y devuelve las salidas. Se permite la ejecución concurrente.

A partir de los resultados de la red con el conjunto de test utilizado en el Capítulo 4 se puede determinar una correspondencia entre el porcentaje de falsos positivos y el umbral de la neurona de salida de la red (ver Tabla 4.16). En los casos en que hay varios umbrales posibles para el porcentaje de FP fijado, se elige al que tiene SCC más alto. Entonces, el umbral de la neurona de salida de la red se determina de esta forma a partir del porcentaje de falsos positivos ingresado como parámetro. Si, se elige la opción de máximo SCC se utiliza el umbral con mayor SCC y menor %FP.

MOSS este módulo toma como entrada las secuencias predichas como promotores por la red neuronal y para éstas calcula las mejores combinaciones de secuencias conservadas y distancias a partir del procedimiento descrito en este capítulo. Como resultado devuelve estas secuencias.

Capítulo 6

Conclusiones y trabajo futuro

En este trabajo estudiamos el problema de reconocimiento y localización de promotores en secuencias de ADN de *Escherichia coli* K12 reconocidas por la subunidad σ^{70} de la ARN polimerasa. Este es un problema interesante desde el punto de vista biológico, dado que la identificación de promotores permite responder algunas preguntas fundamentales acerca de la regulación transcripcional, por ejemplo: ¿qué genes se expresan en una determinada célula en un momento determinado? [21], ¿cómo encuentra la ARN polimerasa a los promotores en el ADN? [41]. La posibilidad de obtener resultados mediante métodos computacionales, le brinda al investigador del campo de la biología la oportunidad de acotar la experimentación a los resultados predichos por estos métodos.

Desde el punto de vista computacional, la complejidad del problema de reconocimiento de promotores radica en que dada una secuencia no se sabe si ésta constituye o no un promotor, a no ser que se hagan los experimentos de laboratorio. Se sabe que los promotores cuentan generalmente con dos subsecuencias conservadas, pero no se conoce exactamente su composición y la distancia entre las mismas es variable. Estas características hacen que la construcción de un modelo preciso del promotor no sea una tarea sencilla, si bien existen modelos probabilísticos. El no poder determinar qué es y qué no es un promotor, hace que no se pueda resolver el problema de reconocimiento de promotores mediante algoritmos exactos.

En este trabajo propusimos y aplicamos un método computacional al problema de reconocimiento de promotores. La solución está basada en una red neuronal *feed-forward* con retardo temporal (TDNN). Las redes neuronales tienen la capacidad de memorizar y generalizar a partir de un conjunto de datos tomados como ejemplo y en particular, la arquitectura de TDNN utilizada permite reconocer las subsecuencias que se encuentran a distancias variables entre sí. Estas dos características hacen que el modelo utilizado sea adecuado para la resolución del problema planteado. Con el objetivo de enriquecer los resultados obtenidos mediante el uso de la red neuronal

implementada, presentamos CPR-MOSS, una metodología de dos niveles, mediante la cuál además de reconocer los promotores informamos cuáles son las subsecuencias conservadas. Esta metodología nos brinda la posibilidad de identificar múltiples promotores en una misma región.

Se aplicaron los métodos computacionales más conocidos, comúnmente utilizados por los investigadores del área de la biología: un método fijo, basado en la búsqueda de patrones, y el estadístico Consensus-Patser, basado en el alineamiento de secuencias. Entre las ventajas de la solución presentada en este trabajo respecto a éstas, pueden mencionarse: 1) nuestro método tiene mayor cantidad de aciertos, con menor cantidad de falsos positivos. La cantidad de promotores reconocidos de la compilación de RegulonDB supera en 7.25% al resultado obtenido con Consensus-Patser y en 68.55% al resultado obtenido con los métodos fijos. La correlación entre la predicción y la realidad es mejor, 2) se entrenó con datos provenientes de las compilaciones más recientes de resultados experimentales. Estos datos prácticamente no han sido utilizados hasta el momento y tienen la ventaja de ser correctos y completos¹, 3) no es necesario contar con un modelo de la solución y 4) la metodología de dos niveles propuesta informa cuáles son las subsecuencias conservadas.

Para poder obtener una red con buenos resultados de memorización y generalización fue necesaria la evaluación de muchas configuraciones distintas que incluyeron variaciones en la topología, distintos modos de presentación de los patrones y distinta inicialización. También se evaluaron los resultados con el uso de distintas proporciones de datos de promotores y de no promotores para el conjunto de entrenamiento. La construcción de la red a partir de los resultados de estas pruebas no fue sencillo. Se concluyó que la variación en la proporción de datos positivos respecto a la de datos negativos en el conjunto de entrenamiento modifica los resultados. La relación 1:10 resultó ser la mejor opción en este caso. Los resultados con distintas inicializaciones de pesos también difirieron. Esto nos llevó a concluir, que es conveniente estudiar con qué valores se inicializa la red. En nuestro caso las inicializaciones que mejor resultaron coinciden con las propuestas por Hertz [31]. Se analizaron también los datos a utilizar para la representación de secuencias de *no promotores* y posibles representaciones de los datos de entrada a la red.

Un análisis de los pesos resultantes del entrenamiento de la red permitió detectar que éstos representan el consenso TGGTATAA. Este consenso también estuvo presente en los alineamientos generados por Consensus. Esto nos permite suponer que el

¹Con *correctos* nos referimos a que gran parte de estos datos ha sido cuidadosamente revisada eliminándose los errores manuales de los pasajes de datos de las publicaciones a las bases de datos. Con *completa*, nos referimos a que la compilación es la más actualizada al momento de realizar este trabajo.

consenso TATAAT podría extenderse, lo que está respaldado en parte por la compilación de Harley y Reynolds, en que se muestra el consenso TATAAT precedido por el consenso GG con una conservación más débil de estos dos nucleótidos.

Analizamos distintas medidas de performance a utilizar en la comparación de los resultados de los métodos predictivos y mejoramos una existente, a la que denominamos *Coefficiente de Correlación Estandarizado o Standardized Correlation Coefficient* (SCC). Esta medida nos permite comparar los resultados de los distintos métodos y determinar el umbral de decisión de acuerdo a alguna de las siguientes opciones: 1) maximizar la correlación entre las predicciones y la realidad, 2) acotar superiormente el porcentaje de resultados falsos positivos y 3) acotar inferiormente el porcentaje de resultados verdaderos positivos.

Por último, dejamos a disposición del público la implementación de la solución, que predice la ubicación de promotores en cadenas de ADN tan largas como se desee. Esta herramienta muestra los resultados que satisfacen los criterios de performance definidos por el usuario, que pueden ser el mayor SCC o un límite superior en el porcentaje de falsos positivos. Estas medidas surgieron a partir de la determinación de distintos umbrales para la evaluación de los resultados obtenidos con el conjunto de test. Para cada promotor predicho por la red se muestran las posibles regiones -10 y -35 resultantes de las ejecuciones del MOSS.

Algunos trabajos futuros que podrían realizarse a partir de este trabajo son:

- **Aplicación del método CPR al reconocimiento de otras secuencias biológicas.** El método CPR podría utilizarse para encontrar motivos *tandem* e *inverted repeats*. También podría utilizarse para encontrar relaciones de distancia entre motivos, por ejemplo, se podría generalizar a una red que encuentre un promotor y su gen correspondiente.
- **Generalización del método para el reconocimiento de promotores de eucariotas.** El reconocimiento de promotores de polimerasa II es útil para la identificación de genes y para la localización de una parte importante de sus regiones regulatorias [19, 82].

En los organismos eucariotas el mecanismo de transcripción es más complejo que en los procariotas. La cantidad de secuencias conservadas encontradas mediante análisis estadísticos es mayor y la distancia entre ellas tiene mayor variabilidad (ver Capítulo 1). Por otro lado, existen algunas clases de promotores que no contienen todas las secuencias conservadas.

El método CPR puede ser adecuado para la resolución del problema de reconocimiento de promotores de polimerasa II. Las consideraciones realizadas en este

trabajo respecto al ajuste de parámetros y el uso de los datos disponibles podrían ser de utilidad. También se puede utilizar la capacidad de modularización del método CPR para el reconocimiento de promotores en que no existen todos los motivos.

- **Predicción de promotores reconocidos por distintos factores sigma en *E. coli* o en otras bacterias.** Los distintos factores sigma (σ) son específicos a distintos promotores (ver Capítulo 1). Para las diferentes clases de promotores reconocidas por distintos factores σ también se han encontrado generalmente dos secuencias consenso a distancias variables [64, 41, 77]. Se podría considerar la posibilidad de entrenar una red con las características de la propuesta por el método CPR.
- **Reentrenamiento de la red con nuevos datos.** A medida que se consigan nuevos datos, es posible que se puedan utilizar para reentrenar la red y de esta forma obtener mejores resultados. Las consideraciones hechas para el reentrenamiento de la red pueden ser útiles y también pueden ser profundizadas en otros trabajos.

Apéndice A

Comparación de medidas de calidad de las predicciones

En la Sección 2.5 mencionamos distintos indicadores existentes para la evaluación de la performance de métodos predictivos. Propusimos el uso de un nuevo indicador, el SCC, variante del CC, que fue el utilizado a lo largo del trabajo. Aquí se muestran los resultados de la aplicación de los distintos indicadores a los métodos estadísticos. Los resultados son similares a los obtenidos con los resultados de la red neuronal. En la Figura A.1 se muestran los resultados de la aplicación del método Consensus-Patser al conjunto de test definido en el Capítulo 4 evaluados de acuerdo a los distintos indicadores mencionados en la Sección 2.5. Luego se analizan las diferencias entre los indicadores. Todos los indicadores fueron estandarizados (i.e. en cada uno se consideró el %TP en lugar de los TP, y el %FP en lugar de FP, etc.) y fueron llevados a valores entre -1 y 1 para poder ser comparados. Para reflejar dicha estandarización, a las siglas de los indicadores (Tabla 2.2) se les agregó el prefijo “S” en la figura.

El SCC parece un mejor indicador que el OP sugerido por Benítez-Bellon et al. [8], debido a que, si bien ambos indicadores llegan a su mayor valor en umbrales parecidos, el OP sigue siendo alto a medida que el porcentaje de TP baja. El PPV tampoco es una buena métrica, ya que crece cuando los positivos crecen (como su nombre y la fórmula lo indican). Entonces el promedio de este valor con la medida de *Accuracy*, como hace Benítez-Bellón para obtener el OP, no parece ser un buen indicador (al menos para este problema). La función del SCC se corresponde bastante con la del *Accuracy*, ambos tienen el máximo en el mismo umbral (i.e. cuando ambos tienen igual cantidad de TP e igual cantidad de TN), y se puede ver en el gráfico que en este punto el porcentaje de TP y el porcentaje de TN son altos.

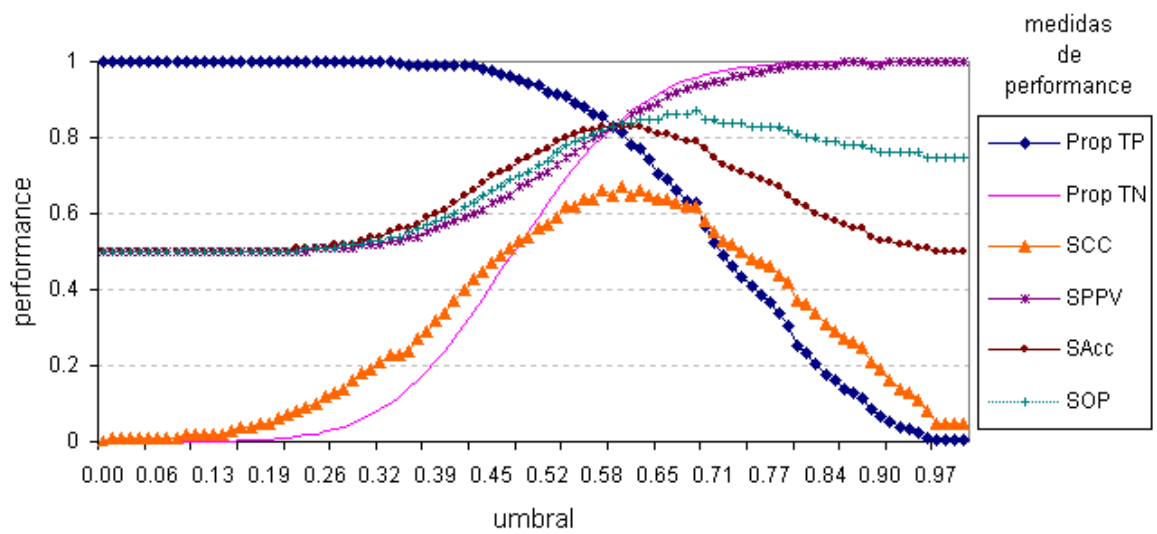


Figura A.1: Comparación de distintos indicadores. Se muestran los indicadores estandarizados y llevados a valores entre -1 y 1 . El prefijo S en las referencias indica que están estandarizados, por ej $SAcc$ es la medida *Accuracy* con el uso de %TP en lugar de TP, %TN en lugar de TN, etc.

Apéndice B

Datos

| Código | Nucleótidos representados | Significado (en inglés) |
|--------|---------------------------|--------------------------------|
| A | A | Adenine |
| C | C | Cytosine |
| G | G | Guanine |
| T | T | Thymine |
| R | A o G | puRines |
| Y | C o T | pYrimidines |
| W | A o T | Weak hydrogen bonding |
| S | G o C | Strong hydrogen bonding |
| M | A o C | aMino group at common position |
| K | G o T | Keto group at common position |
| H | A, C o T | not G |
| B | G, C o T | not A |
| V | G, A o C | not T |
| D | G, A o T | not C |
| N | G, A, C o T | aNy |

Tabla B.1: Códigos ambiguos de IUPAC-IUB.

| Nombre secuencia | Pos. ttgaca | Pos. tataat | Secuencia |
|---------------------------|-------------|-------------|---|
| aceEF | 13 | 36 | ACGTAGACCTGT CTTATT GAGCTTTC CGGCGAGAG TTCAAT GGGACAGTCCAG |
| ada | 15 | 38 | AGCGGCTAAAGGTG TTGACG TGGAGAAA ATGTTTAGC TAAACT TCTCTCATGG |
| alaS | 15 | 39 | AACGCATACGGTAT TTTACC TTGCCAGTC AAGAAAAC TATCTT ATTCCCATTTCAGT |
| ampC | 15 | 37 | TGCTATCCTGACAG TTGTCA CGCTGATT GGTGTCTG TACAAT CTAACGCATCGCCAATG |
| ampC/C16 | 7 | 30 | GCTATC TTGACA GTTGTAC GCTGATTG TATCGT TACAATCTAACGTATCG |
| araBAD | 15 | 37 | TTAGCGGATCCTAC CTGACG CTTTTTAT CGGAACTC TCTACT GTTTCCTCATACCCGTT |
| araC | 15 | 38 | GCAAATAATCAATG TGGACT TTCTGCGC GTGATTATA GACTCT TTTGTTAGCGGTTTTTG |
| araE | 12 | 37 | CTGTTTCCGAC CTGACA CCTGGGTGA GTTGTTCACG TATTTT TTCACTATGTCTACTC |
| araI(c) | 13 | 35 | AGCGGATCCTAC CTGGCG CTTTTTAT CGGAACTC TCTACT GTTTCCTCATACCCGTT |
| araI(c)X(c) | 13 | 37 | AGCGGATCCTAC CTGGCG CTTTTTATC GCAACTCTC TACTAT TTCCTCATACCCGTTTT |
| argCBH | 15 | 39 | TTTGTTTTTTCATTG TTGACA CACCTCTGG TCATGATAG TATCAA TATTCATGCAGTAT |
| argCBH-P1/6- | 15 | 36 | TTTGTTTTTTCATTG TTGACA CACCTCT GGTCAATA TATTAT CAATATTATGCAGTAT |
| argCBH-P1/LL | 15 | 36 | TTTGTTTTTTCATTG TTGACA CACCTCT GGTCAATGA TATTAT CAATATTATGCAGTAT |
| argE-P1 | 15 | 38 | TTACGGCTGGTGGG TTTTAT TACGCTCA ACGTTAGT TATTTT TATTCATAAATCGCA |
| argE-P2 | 15 | 38 | CCGCATCATTGCTT TGGCCT GAAACAGT CAAAGCGGT TATGTT CATATGCGGATGGCG |
| argE/LL13 | 15 | 38 | CCGCATCATTGCTT TGGCCT GAAACAGT CAAAGCGGT TATATT CATATGCGGATGGCG |
| argF | 15 | 38 | ATTGTGAAATGGGG TTGCAA ATGAATAA TTACACATA TAAAGT GAATTTTAAATCAATA |
| argI | 7 | 30 | TTAGAC TTGCAA ATGAATAA TCATCCATA TAAATT GAATTTTAAATTCATTGA |
| argR | 12 | 35 | TCGTGCGCGCG TTGACG GAGCAAG CTTTGACAA TATTAA TCAGTCTAAAGTCTCGG |
| aroF | 15 | 37 | TACGAAAATATGGA TTGAAA ACTTTACT TTATGTGT TATCGT TACGTACCTCGCTG |
| aroG | 15 | 38 | AGTGTAACACCCCG TTTACA CATTCTGA CGGAAGATA TAGATT GGAAGTATTCATTCA |
| aroH | 15 | 37 | GTACTAGAGAATA GTGCAT TAGCTTAT TTTTTTGT TATCAT GCTAACCCCGCGGAG |
| bioA | 15 | 39 | GCCTTCTCCAAAAC GTGTTT TTTGTTGTT AATTGCGTG TAGACT TGTAACCTAAATCT |
| bioB | 15 | 38 | TTGTCATAATCGAC TTGTAA ACCAAATT GAAAAGATT TAGGTT TACAAGTCTACCCGAA |
| bioP98 | 15 | 38 | TTGTTAATTCGGTG TAGACT TGTAACCC TAAATCTTT TAAATT TGGTTTACAAGTCTGAT |
| C62.5-P1 | 14 | 37 | CACCTGCTCTCGC TTGAAA TTATTCTC CCTGTGCC CATCTC TCCCACATCTGTTTT |
| carAB-P1 | 15 | 38 | ATCCCGCATTAAAG TTGACT TTTAGCGC CCATATCTC CAGAAT CGCGCGTTGGCAGA |
| carAB-P2 | 15 | 39 | TAAGCAGATTGCA TTGATT TAGCTCATC ATTGTGAAT TAATAT GCAATAAAGTGAG |
| cat | 13 | 36 | ACGTTGATCGGC ACGTAA GAGGTTC AACTTTCAC CATAAT GAAATAAGTCACTACC |
| cit.util-379 | 13 | 37 | AAACAGGCGGGG GTCTCA GCGACTAA CCCGCAAC TCTTAC CTCTATACATAATTTCTG |
| cit.util-431 | 14 | 38 | GACAGGACAGCA TTGTAC GATCAACTG ATTTGTGCC AATAAT TAAATGAAATCAC |
| CloDFcloacin | 15 | 37 | TCATATATTGACAC CTGAAA ACTGGAGG AGTAAAGT AATAAT CATACTGTATATAT |
| CloDFnaI | 15 | 39 | ACAGCGGTTGCTC TTGAAG TGTGCGCCA AAGTCCGG TACTCT GGAAGACAGATTGG |
| colE1-B | 15 | 36 | TTATAAAATCCTCT TTGACT TTTAAAA CAATAAGT TAAAA TAAATACTGTAA |
| colE1-C | 15 | 37 | TTATAAAATCCTCT TTGACT TTTAAAAA AATAAGTT AAAAAA AAATACTGTACATATA |
| colE1-P1 | 15 | 38 | GGAAAGTCCACAGTC TTGACA GGGAAAAA CGAGCGGG TAGCTT TTATGTCTATATAAAA |
| colE1-P2 | 15 | 37 | TTTTTAACCTATTG TTTTAA AAGTCAAA GAGGATTT TATAAT GAAAACCCGGTAGCGT |
| colE110.13 | 13 | 37 | GCTACAGAGTTC TTGAAG TAGTGGCCC GACTACGGC TACTCT AGAAGGACAGATTTTGG |
| colicinE1 P3 | 15 | 37 | TTTTTAACCTATTG TTTTAA AAGTCAAA GAGGATTT TATAAT GAAAACCCGGTAGCGT |
| crp | 15 | 38 | AAGCGAGACACAG GAGACA CAAAGCGA AAGCTATG TAAAA AGTCAGGATGCTACAG |
| cya | 15 | 38 | GTAGCGCATCTTTC TTTACG GTCAATCA GCAAGGTG TAAATT GATCACGTTTTAGACC |
| dapD | 15 | 39 | AAGTGCATCAGCG TTGACA GAGGCCCTC AATCCAAAC GATAAA GGGTGTATGTTTACTG |
| deo-P1 | 14 | 39 | CAGAAACGTTTTA TTCGAA CATCGATCT CGTCTTGTGT TAGAAT TCTAACATACGGTTGC |
| deo-P2 | 10 | 35 | TGATGTGTA TCGAAG TGTGTTGCG GAGTAGATG TAGAAT ACTAACAACTCGCAA |
| deo-P3 | 15 | 37 | ACACCAACTGTCTA TCGCCG TATCAGCG AATAACGG TACTCT GATCTGATCATTTAAA |
| divE | 15 | 38 | AAACAAATTAGGGG TTTACA CGCCGAT CCGGATGTT TATAGT CGCGCATCTCCGGAAG |
| dnaA-1p | 15 | 39 | TGCGGCGTAAATCG TGCCCG CCTCGCGGC AGGATCGTT TACTCT TAGCGAGTTCTGAAA |
| dnaA-2p | 15 | 38 | TCTGTGAGAAACAG AAGATC TCTTGCGC AGTTTAGG TATGAT CCGCGTCCCGATCG |
| dnaK-P1 | 15 | 39 | TTTGCATCTCCCC TTGATG ACGTGGTTT ACGACCCA TTTAGT AGTCAACCGCATG |
| dnaK-P2 | 15 | 37 | ATGAAATTGGGCAG TTGAAA CCAGACGT TTCGCCCC TATTAC AGACTCACAACCACA |
| dnaQ-P1 | 15 | 37 | GCCAGCGCTAAAGG TTTTCT CGCGTCCG CGATAGCG TAAAT AGCGCGTAAACCC |
| Fpla-oriTpX | 15 | 38 | GAACCAACCACTG TTGAGC CTTTTTGT GGAGTGGT TAAATT ATTTACGGATAAAG |
| Fplas-traM | 15 | 38 | ATTAGGGGTGCTG TAGCGG CGCGGTGT GTTTTTTA TAGGAT ACCGCTAGGGCGCTG |
| Fplas-traY/Z | 14 | 37 | GCGTTAATAAGGT GTTAAT AAAATATA GACTTTCG TCTATT TACCTTTCTGATTATT |
| frdABCD | 12 | 34 | GATCTCGTCAA ATTTCA GACTTATC GATCAGAC TACTCT GTTGTACCTATAAAGGA |
| fumA | 15 | 38 | GTACTAGTCTCAGT TTTTGT TAAAAAAG TGTGTAGGA TATTGT TACTCGTTTTAACAGG |
| γ - δ -tnpA | 15 | 38 | ACACATTAACAGCA CTGTTT TTATGTGT CCGATAAT TATAAT ATTTGCGAGCGTTGCA |
| γ - δ -tnpR | 14 | 36 | ATTCATTAACAAT TTTGCA ACCGTCCG AAATAATTA TAAATT ATCGCACATATAAAAA |

| Nombre secuencia | Pos. ttgaca | Pos. tataat | Secuencia |
|------------------|-------------|-------------|--|
| gal-P1 | 15 | 38 | TCCATGTCACACTT TTCGCA TCTTTGTT ATGCTATGG TTATTT CATACCATAAG |
| gal-P2 | 15 | 37 | CTAATTTATCCAT GTCACA CTTTTGGC ATCTTTGT TAGTCT ATGGTTATTTCAACC |
| gal-P2/mut-1 | 14 | 36 | TAATTTATCCAT GTCACA CTTTTGGC ATCTTTGT TATACT ATGGTTATTTCAAC |
| gal-P2/mut-2 | 14 | 36 | TAATTTATCCAT GTCACA CTTTTGGC ATTTTGT TAGTCT ATGGTTATTTCAAC |
| glnL | 15 | 40 | CAATTCCTGATGC TTGCGG CTTTTTATC CGTAAAAAGC TATAAT GCACATAATGGTGC |
| gln | 15 | 38 | TAAAAAATAACAG TTGTCA GCCTGTCC CGTTATAA GATCAT ACGCCGTTATACGTT |
| gltA-P1 | 15 | 37 | ATTCATTGGGACA GTTATT AGTGGTAG ACAAGTTT AATAAT TCGGATTGTAAGTA |
| gltA-P2 | 15 | 39 | AGTTGTTACAAACA TTACCA GAAAAAGCA TATAATGCG TAAAAG TTATGAAAGTGGT |
| glyA | 15 | 38 | TCCTTTGTCAAGAC CTGTTA TCGCACAA TGATTGCGT TATACT GTTCGCGGTTGTCC |
| glyA/geneX | 15 | 39 | ACACCAAAGAACCA TTTACA TTGAGGGC TATTTTTTA TAAGAT GCATTTGAGATACAT |
| gnd | 15 | 38 | GCATGGATAAGCTA TTTATA CTTAATA AGTACTTTG TATACT TATTTGCGAACATTCCA |
| groE | 11 | 34 | TTTTTCCCC TTGAAG GGGGGAAG CCATCCCCA TTTCTC TGGTCACCAGCCGGAA |
| gyrB | 11 | 38 | CGGACGAAA TTGCAA GATGTTTACCGTGGAAAAGG TAAAAT AACGGATTAAACCAAGT |
| his | 14 | 38 | ATATAAAAAGTTC TTGCTT TCTAACGTG AAAGTGGTT TAGGTT AAAAGACATCGATTGA |
| hisA | 15 | 38 | GATCTACAACTAA TTAATA AATAGTTA ATTAACGCT CATCAT TGTACAATGAACGTAC |
| hisBp | 15 | 38 | CCTCCAGTGGGGT TTTAAA TCTTTGTG GGATCAGG CATTAT CTTACGTGATCG |
| hisJ(St) | 15 | 37 | TAGAATGCTTTGCG TTGTCG GCCTGATT AATGGCAC GATAGT CGCATCGGATGCT |
| hisS | 15 | 38 | AAATAATAACGTGA TGGGAA GCGCCTCG CTTCCCGTG TAGTAT TGAACCCGATGGCTC |
| htpR-P1 | 15 | 38 | ACATTACGCCACT ACGCCT GAATAATA AAAGCGTG TATACT CTTTCTGCAATGGTT |
| htpR-P2 | 15 | 39 | TTACAAAGCTTGA TTGAAC TTGTGGATA AAATCACGG TCTGAT AAAACAGTGAATG |
| htpR-P3 | 15 | 38 | AGCTTGCATTGAAC TTGTGG ATAAAATC ACGGTCTGA TAAAAC AGTGAATGATAACCTCGT |
| ilvGEDA | 15 | 38 | GCCAAAAATATCT TGTACT ATTTACAA AACCTATG TAACTC TTTAGGACTTCTTCTGA |
| ilvIH-P1 | 14 | 37 | CCTCGGCTGCCAA TTGCTT AAGCAAGA TCGGACGGT TAATGT GTTTTACACATTTTTTC |
| ilvIH-P2 | 15 | 38 | GAGGATTTATCGT TTCTTT TCACCTTT CCTCCTGTT TATTCT TATTACCCCGTGT |
| ilvIH-P3 | 14 | 37 | ATTTTAGGATTA TTTAAA AAATAGAG AAATTGCTG TAAGTT GTGGGATTCAGCCGATT |
| ilvIH-P4 | 15 | 38 | TGTAGAATTTTATT CTGAAT GTCTGGGG TCTCTATT TAGGAT TAATTAATAAATAGAG |
| ISlins-PL | 15 | 37 | CGAGGCCGGTGATG CTGCCA ACTTACTG ATTTAGTG TAGTAT GGTGTTTTGAGGTGCT |
| ISlins-PR | 13 | 36 | ATATATACCTTA TGGTAA TGACTCCA ACTTATTG TAGTGT TTTATGTCAGATAAT |
| IS2I-II | 7 | 30 | GATGTC TGGAAA TATAGGGG CAAATCCAC TAGTAT TAAGACTATCACTATT |
| lacI | 15 | 38 | GACACCATCGAATG GCGCAA AACCTTTC GCGGTATG CATGAT ACGCCCGGAGAGAGAT |
| lacP1 | 15 | 39 | TAGGCACCCAGCG TTTACA CTTTATGTT TCCGGCTCG TAGTGT GTGTGGAATTGTGAGC |
| lacP115 | 14 | 37 | TTTACACTTTATG CTCCCG GCTCGTAT GTTGTGTGG TATTGT GAGCGGATAACCAATTT |
| lacP2 | 15 | 38 | AATGTGAGTTAGCT CACTCA TTAGGCAC CCCAGGCTT TACACT TTATGCTTCCGGCTCG |
| lep | 15 | 37 | TCCTCGCCTCAATG TTGTAG TGTAGAAT GCGGCGTT TCTATT AATACAGACGTTAAT |
| leu | 2 | 25 | G TTGACA TCCGTTTT TGTATCCAG TAACTC TAAAAGCATATCCGATT |
| leultRNA | 15 | 37 | TCGATAAATTAECTA TTGACG AAAAGCTG AAAACCAC TAGAAT GCGCCTCCGTTGGTGA |
| lex | 15 | 38 | TGTGCAGTTTATGG TTCCAA AATCGCCT TTTGCTGTA TATACT CACAGCATAACTGTAT |
| livJ | 15 | 38 | TGTCAAAAATAGCTA TTCCAA TATCATAA AAATCGGA TAGTGT TTAGCAGAGTATGT |
| lpd | 7 | 30 | TTGTTG TTTAAA AATTGTTA ACAATTTGT TAAAAT ACCGACGGATAGAAGCA |
| lpp | 15 | 38 | CCATCAAAAAAATA TTCTCA ACATAAAA AACTTTGTG TAATAC TTGTAACGCTACATGGA |
| lppP1 | 13 | 37 | ATCAAAAAAATA TTCTCA ACATAAAA ACTTTGTGT TATACT TGTAAACGCTACATGGA |
| lppP2 | 13 | 37 | ATCAAAAAAATA TTCTCA ACATAAAA ACTTTGTGT TATAAT TGTAAACGCTACATGGA |
| lppR1 | 13 | 36 | ATCAAAAAAATA TTCAACA ACATAAAA A ACTTTGT GTAATA CTTGTAACGCTACATGGA |
| Mlrna | 15 | 38 | ATGGCAACCGCGG GTGACA AGGGCGCG CAAACCCTC TATACT GCGCGCCGAAAGCTGACC |
| mac11 | 14 | 38 | CCCCCGCAGGGAT GAGGAA GGTGGTCTGA CCGGCTCG TAGTGT GTGTGGAATTGTGAGC |
| mac12 | 14 | 38 | CCCCCGCAGGGAT GAGGAA GGTGGTCTGA ACCGGCTCG TAGTGT GTGTGGAATTGTGAGC |
| mac21 | 14 | 38 | CCCCCGCAGGGAT GAGGAA GGTGCAACT TCCGCTCG TAGTGT GTGTGGAATTGTGAGC |
| mac3 | 14 | 37 | CCCCCGCAGGGAT GAGGAA GGTGGTCT GACCGCTCG TAGTGT GTGTGGAATTGTGAGC |
| mac31 | 14 | 37 | CCCCCGCAGGGAT GAGGAA GGTGGTCT GACCGCTCG TATATT GTGTGGAATTGTGAGC |
| malEFG | 15 | 37 | AGGGGCAAGGAGGA TGGAAA GAGGTTGC CGTATAAA GAAACT AGAGTCCGTTTAGGTTG |
| malK | 15 | 37 | CAGGGGTGGAGGA TTTAAG CCATCTCC TGATGACG CATAGT CAGCCCATCATGAATG |
| malPQ | 15 | 38 | ATCCCCGAGGATG AGGAAG GTCAACAT CGAGCCTGG CAAACT AGCGATAACGTTGTGT |
| malPQ/A516P1 | 12 | 34 | ATCCCCGAGG ATGAGG AGCCTGGC AAACCTAGC GATGAT AACCTGTTGTGAA |
| malPQ/A516P2 | 15 | 39 | ATCCCCGAGGAG ATGAGG AGCCTGGCA AACTAGCGA TAACGT TGTGTTGAAAA |
| malPQ/A517/A | 15 | 37 | CCCCGAGGATGAG GTGAG CCTGGCAA ACTAGCGA TAACGT TGTGTTGAAAA |
| malPQ/Pp12 | 14 | 37 | ATCCCCGAGGAT GAGGAA GGTCAACA TCGAGCCTG GAAAA TAGCGATAACGTTGTGT |
| malPQ/Pp13 | 14 | 38 | ATCCCCGAGGAT TAGGAA GGTCAACAT CGAGCCTGG CAAACT AGCGATAACGTTGTGT |
| malPQ/Pp14 | 14 | 37 | ATCCCCGAGGAT GAGGAA GGTCAACA TCGAGCCTG GAAACT AGCGATAACGTTGTGT |

| Nombre secuencia | Pos. ttgaca | Pos. tataat | Secuencia |
|------------------|-------------|-------------|--|
| malPQ/Pp15 | 14 | 38 | ATCCCCGAGGAT GAGAAA GGTCACAT CGAGCCTGG CAAACT AGCGATAACGTTGTGT |
| malPQ/Pp16 | 15 | 38 | ATCCCCGAGGATA AGGAAG GTCAACAT CGAGCCTGG CAAACT AGCGATAACGTTGTGT |
| malPQ/Pp18 | 15 | 38 | ATCCCCGAGGATG GGAAG GTCAACAT CGAGCCTGG CAAACT AGCGATAACGTTGTGT |
| malT | 15 | 37 | GTCAATCGCTGCAT TAGAAA GGTTTCTG GCCGACCT TATAAC CATTAAATTACG |
| manA | 15 | 38 | CGGCTCCAGGTTAC TTCCCG TAGGATTC TTGCTTTAA TAGTGG GATTAATTTCCACATTA |
| metA-P1 | 15 | 38 | TTCAACATGCAGGC TCGACA TTGGCAAA TTTTCTGGT TATCTT CAGCTATCTGGATGT |
| metA-P2 | 15 | 38 | AAGACTAATTACCA TTTTCT CTCCTTTT AGTCATTCT TATATT CTAACGTAGTCTTTTCC |
| metBL | 12 | 35 | TTACCGTGACA TCGTGT AATGCACC TGTCGGCGT GATAAT GCATATAATTTTAAACGG |
| metF | 8 | 31 | TTTTCGG TTGACG CCCTTGGG CTTTTCTT CATCTT TACATCTGGAGC |
| micF | 15 | 37 | GCGGAATGGCGAAA TAAGCA CCTAACAT CAAGCAAT AATAAT TCAAGGTTAAAAATCAAT |
| motA | 15 | 39 | GCCCAATCGCGCG TTAACG CCGTACGAC TGAACATCC TGTCAT GGTCACAGTGGA |
| MuPc-1 | 6 | 33 | AAATT TTGAAA AGTAACTTTATAGAAAAGAAT AATACT GAAAAGTCAATTTGGTG |
| MuPc-2 | 9 | 32 | GGAACACA TTTAAA AACCTCTC TAAGTTTTG TAATCT ATAAAGTTAGCAATTTA |
| MuPe | 15 | 38 | TACCAAAAAGCACC TTTACA TTAAGCTT TTCAGTAAT TATCTT TTTAGTAAGCTAGTFA |
| NR1rnaC | 15 | 39 | GTCACAATTCTCAA GTCGCT GATTTCAAA AAACTGTAG TATCCT CTGCGAAAACGATCCCT |
| NR1rnaC/m | 15 | 38 | TCACAATTCTCAAG TTGCTG ATTTCAAA AAACTGTAG TATCCT CTGCGAAAACGATCCCT |
| NTP1rna100 | 11 | 35 | GGAGTTTGTC TTGAAG TTATGCACC TGTTAAGGC TAAACT GAAAGAACAGATTTTGT |
| nusA | 7 | 30 | CAGTAT TTGCAT TTTTACC CAAAACGAG TAGAAT TTGCCACGTTTCAGGGC |
| ompA | 12 | 34 | GCCTGACGGAG TTCACA CTTGTAAG TTTTCAAC TACGTT GTAGACTTTAC |
| ompC | 15 | 38 | GTATCATATTCTG TGAT TATTCTGC ATTTTTGGG GAGAAAT GGACTTGGCGACTG |
| ompF | 7 | 30 | GGTAGG TAGCGA AACGTTAG TTTGAATGG AAAGAT GCCTGCAGACACATAAA |
| ompF/pKI217 | 3 | 26 | GG TAGCGA AACGTTAG TTTGCAAG TTTAAT GCGGTAGTTTATCAC |
| ompR | 15 | 36 | TTTCGCGAATAAA TTGTAT ACTTAAG CTGCTGTT TAATAT GCTTTGTAAACAATTT |
| p15primer | 15 | 38 | ATAAGATGATCTTC TTGAGA TCGTTTTG GTCTGCGCG TAATCT CTGCTTGAACCGAAA |
| p15rnaI | 15 | 39 | TAGAGGAGTTAGTC TTGAAG TCATGCGCC GGTTAAGGC TAAACT GAAAGGACAAAGTTTTG |
| P22ant | 15 | 38 | TCCAAGTTAGTGTG TTGACA TGATAGAA GCACTCTAC TATATT CTCAAATAGTCCACGG |
| P22mnt | 15 | 38 | CCACCGTGGACCTA TTGAGA ATATAGTA GAGTGTCTC TATCAT GTCAATACACTAACTT |
| P22PR | 15 | 37 | CATCTTAAATAAAC TTGACT AAAGATTC CTTTAGTA GATAAT TTAAGTGTCTTTAAT |
| P22PRM | 9 | 32 | AAATTATC TACTAA AGGAATCT TTAGTCAAG TTTATT TAAGATGACTTAACTAT |
| pBR313Htet | 12 | 35 | AATTCTCATGT TTGACA GCTTATCA TCGATAAGC TAGCTT TAATGCGGTAGTTTAT |
| pColViron-P1 | 15 | 38 | TCACAATTCTCAAG TTGATA ATGAGAAAT CATTATTGA CATAAT TGTTATTATTTTAC |
| pColViron-P2 | 13 | 35 | TGTTTCAACACC ATGTAT TAATTGTG TTTATTTG TAAAAA TAATTTTCTGACAATAA |
| pEG3503 | 6 | 30 | CTGGC TGGACT TCGAATTCA TTAATGCGG TAGTTT ATCACAGTTAA |
| phiXA | 15 | 38 | AATAACCGTCAGGA TTGACA CCCTCCCA ATTTGATGT TTTTAC GCCTCCAAATCTTGA |
| phiXB | 15 | 39 | GCCAGTTAAATAGC TTGCAA AATACGTGG CCTTATGTT TACAGT ATGCCCATCGCAGTT |
| phiXD | 15 | 39 | TAGAGATTCTCTTG TTGACA TTTTAAAAG ACGCTGGAT TACTAT CTGAGTCCGATGCTGTT |

Tabla B.2: Conjunto de entrenamiento de la red neuronal. Provenientes de la compilación de Harley y Reynolds.

| Nombre secuencia | Pos. ttgaca | Pos. tataat | Secuencia |
|------------------|-------------|-------------|--|
| lambdac17 | 15 | 38 | GGTGTATGCATTTA TTTGCA TACATTCA ATCAATTGT TATAAT TGTATCTAAGGAAAT |
| lambdacin | 15 | 38 | TAGATAACAATTGA TTGAAT GTATGCAA ATAAATGCA TACAAT ATAGGTGGTAAAT |
| lambdaL57 | 14 | 37 | TGATAAGCAATGC TTTTTT ATAATGCC AACTTAGTA TAAAA AGCCAACCTGTTGCGACA |
| lambdaPI | 15 | 38 | CGGTTTTTCTTGC GTGTAA TTGCGGAG ACTTTGCGA TGTACT TGACACTTCAGGAGTG |
| lambdaPL | 15 | 38 | TATCTCTGGCGGTG TTGACA TAAATACC ACTGGGGT GATACT GAGCACATCAGCAGGA |
| lambdaPo | 15 | 38 | TACCTCTGCCGAAG TTGAGT ATTTTTGC TGTATTTGT CATAAT GACTCTGTGTGATAGAT |
| lambdaPR | 15 | 38 | TAACACCGTGGCTG TTGACT ATTTTACC TCTGGGGT GATAAT GGTTCATGTACTAAG |
| lambdaPR' | 15 | 38 | TTAACGGCATGATA TTGACT TATTGAAT AAAATTGGG TAAATT TGACTCAAGCATGGGT |
| lambdaPRE | 15 | 39 | GAGCCTCGTTGCGT TTGTTT GCACGAACC ATATGTAAG TATTTT CTTAGATAACAAT |
| lambdaPRM | 15 | 38 | AACACCGCAGGTGT TAGATA TTTATCCC TTGGGGTGA TAGATT TAACGTATGAGCACAA |
| pBR322bla | 15 | 38 | TTTTCTAAATACA TTCAAA TATGTATC CGCTCATGA GACAAT AACCTGATAAATGCT |
| pBR322P4 | 15 | 42 | CATCTGTGCGGTAT TTCACA CCGCATATGGTGCACCTCTCAG TACAAT CTGCTCTGATGCCGAT |
| pBR322primer | 15 | 38 | ATCAAAGGATCTTC TTGAGA TCCTTTTT TCTGCGCG TAACTT CTTGCTTGCACAAACAAA |
| pBR322tet | 15 | 38 | AAGAATTCTCATGT TTGACA GCTTATCA TCGATAAGC TTTAAT GCGGTGTTTATCAC |
| pBRH4-25 | 4 | 27 | TCG TTTTCA AGAATTCA TTAATGCGG TAGTTT ATCACAGTTAA |
| pBRP1 | 15 | 42 | TTCATACACGGTGC CTGACT GCGTTAGCAATTTAACTGTGA TAAACT ACCGATTAAAGCCTTA |
| pBRRNAI | 15 | 39 | GTGCTACAGAGTTC TTGAAG TGGTGGCTT AACTACGGC TACAAT AGAAGCACAGATTTTG |
| pBRtet-10 | 15 | 38 | AAGAATTCTCATGT TTGACA GCTTATCA TCGATGCGG TAGTTT ATCACAGTTAA |
| pBRtet-15 | 15 | 38 | AAGAATTCTCATGT TTGACA GCTTATCA TCGGTAGTT TATCAC AGTAAATTTGC |
| pBRtet-22 | 15 | 39 | AAGAATTCTCATGT TTGACA GCTTATCAT CGATCACAG TTAAT TGCTAACGCGAG |
| pBRtet/TA22 | 10 | 33 | TTCTCATGT TTGACA GCTTATCA TCGATAAGC TAAAT TATATAAAAATTTAGCT |
| pBRtet/TA33 | 10 | 33 | TTCTCATGT TTGACA GCTTATCA TCGATAAGC TAAAT TATATAAAAATTTATAT |
| pori-I | 15 | 38 | CTGTGTGTTGAGTTT TTGAGT TGTGTATA ACCCCTCAT TCTGAT CCGAGCTTATACGCT |
| pori-r | 15 | 39 | GATCGCACGATCTG TATACT TATTTGAGT AAATTAACC CACGAT CCCAGCCATCTTCTGCG |
| ppc | 15 | 38 | CGATTTCGCGCAT TTGACG TCACCGCT TTTACGTGG CTTTAT AAAAGACGACGAAAA |
| pSC101oriP1 | 3 | 30 | TT TTGTAG AGGAGCAAAACGCGTTTGGCA CATCCT TTTGTAATACTGCGGAA |
| pSC101oriP2 | 8 | 30 | ATTATCA TTGACT AGCCCATC TCAATTGG TATAAT GATTAATAACACCTAGA |
| pSC101oriP3 | 15 | 38 | ATACGCTCAGATGA TGAACA TCGATTAGG GAAAATGCT TATGTT GTATTAGCTAAAAG |
| pyrB1-P1 | 15 | 37 | CTTTCACACTCCGC CCTATA AGTCGGAT GAATGGAA TAAAA GCATATCTGATTCGCTG |
| pyrB1-P2 | 13 | 36 | TTGCATCAAATG CTTGCG CCGCTTCT GACGATGAG TATAAT GCGGACAAATTTGCCGG |
| pyrD | 15 | 38 | TTGCCGAGGTCAA TTCCCT TTTGGTCC GAACTGCGA CATAAT ACGCCCGCGTTTG |
| pyrE-P1 | 15 | 38 | ATGCCTTGTAAAGTA TAGGAA TAACCGCC GGAAGTCCG TATAAT GCGACGCCACATTTG |
| pyrE-P2 | 14 | 38 | GTAGGCGGTCCATA CTGCGG ATCATAGAC GTTCCTGTT TATAAA AGGAGAGGTGGAAGG |
| R100rna3 | 15 | 39 | GTACCGGCTTACGC CGGGCT TGGCGGGT TTACTCTGT TATCAT ATGAAACAACAGAG |
| R100RNAI | 15 | 38 | CACAGAAAGAAGTC TTGAAC TTTTCGGG GCATATAAC TATACT CCGCGCATAGCTGAAT |
| R100RNAII | 15 | 38 | ATGGGCTTACATTC TTGAGT GTTCAGAA GATTAGTGC TAGATT ACTGATCGTTAAGGAA |
| R1RNAII | 15 | 37 | ACTAAAGTAAAGAC TTTACT TTGTGGCG TAGCATGC TAGATT ACTGATCGTTAAGGAA |
| recA | 15 | 37 | TTTTACAAAACAC TTGATA CTGTATGA GCATACAG TATAAT TGCTTCAACAGACAT |
| rnh | 15 | 38 | GTAAAGCGGTCAATT ATGTCA GACTTGTC GTTTTACAG TCTGAT TCAATTACAGGA |
| rn(pRNaseP) | 15 | 38 | ATGCGCAACGCGGG GTGACA AGGGCGCG CAACCCCTC TATACT GCGCGCGAAGCTGACC |
| rp1J | 15 | 38 | TGTAATACTAATGCC TTTACG TGGGCGGT GATTTTGTG TACAAT CTTACCCCAAGCTATA |
| rpmH1p | 15 | 38 | GATCCAGGACGATC CTTGCG CTTTACCC ATCAGCCCG TATAAT CTTCCACCCGCGCG |
| rpmH2p | 15 | 38 | ATAAGGAAAGAGAA TTGACT CCGGAGTG TACAATTAT TACAAT CCGGCTCTTTAATC |
| rpmH3p | 15 | 38 | AAATTTAATGACCA TAGACA AAAATTGG CTTAATCGA TCTAAT AAAGATCCGAGCAG |
| rpoA | 15 | 38 | TTGCGATATTTTTT TTGCAA AGTTGGGT TGAGTGGC TAGATT AGCCAGCCAATCTTT |
| rpoB | 15 | 37 | CGACTTAATATACT GCGACA GGACGTCC GTTCTGTG TAAATC GCAATGAAATGGTTAA |
| rpoD-Pa | 13 | 36 | CGCCCTGTTCCG CAGCTA AAACGCAC GACCATGCG TATACT TATAGGGTTGC |
| rpoD-Pb | 9 | 33 | AGCCAGGT CTGACC ACCGGGCAA CTTTATGAG CACTAT CGTGTACAAAT |
| rpoD-Phs | 13 | 36 | ATGCTGCCACCC TTGAAA AACTGTGC ATGTGGGAC GATATA GCAGATAAGAA |
| rpoD-Phs/min | 4 | 31 | CCC TTGAAA AACTGTGCATGTGGGACGATA TAGCAG ATAAAGAAATTTGCT |
| rrn4.5S | 14 | 37 | GGCACGCGATGGG TTGCAA TTAGCCGG GGCAGCAGT GATAAT GCGCTGCGGTTGGTT |
| rrnABP1 | 15 | 37 | TTTTAAATTTCTC TTGTCA GGCCGGAA TAACTCCC TATAAT GCGCCACCCTGACACG |
| rrnABP2 | 15 | 37 | GCAAAAATAAATGC TTGACT CTGTAGCG GGAAGGCG TATTAT GCACACCCGCGCCCG |
| rrnB-P3 | 14 | 40 | CTATGATAAGGAT TACTCA TCTTATCCTT ATCAAACCGT TAAAA GGGCGGTGTGAGCTTG |
| rrnB-P4 | 15 | 36 | CGGTATCCGGTCC CTCTCA CTGACA GTTCTGTG TAAAA AGCCAACCTGTTGCGACA |
| rrnDEXP2 | 15 | 37 | CCTGAAATTCAGGG TTGACT CTGAAAGA GGAAAGCG TAATAT ACGCCACCTCGGCACAG |
| rrnD-P1 | 15 | 37 | GATCAAAAAAATAC TTGTGC AAAAAATT GGGATCCC TATAAT GCGCTCCGTTGAGACG |
| rrnE-P1 | 15 | 37 | CTGCAATTTTCTA TTGCGG CTTGCGGA GAACTCCC TATAAT GCGCTCCATCGACAG |
| rrnG-P1 | 15 | 37 | TTTTATTTTTTCG TTGTCA GGCCGGAA TAACTCCC TATAAT GCGCCACCCTGACACG |
| rrnG-P2 | 15 | 37 | AAGCAAGAAATGC TTGACT CTGTAGCG GGAAGGCG TATTAT GCACACCCGCGCCCG |

| Nombre secuencia | Pos. ttgaca | Pos. tataat | Secuencia |
|------------------|-------------|-------------|--|
| rrnX1 | 15 | 37 | ATGCATTTTTCCGC TTGTCT TCCTGAGC CGACTCCC TATAAT GGCCTCCATCGACACC |
| RSFprimer | 15 | 38 | GGAAATAGCTGTTCG TTGACT TGATAGAC CGATTGATT CATCAT CTCATAAATAAGAA |
| RSFrnaI | 15 | 39 | TAGAGGAGTTTGTG TTGAAG TTATGCACC TGTTAAGGC TAAACT GAAAGAACAGATTTTG |
| S10 | 15 | 37 | TACTAGCAATACGC TTGCGT TCGGTGGT TAAGTATG TATAAT GCGCGGCTTGTGCT |
| sdh-P1 | 14 | 37 | ATATGTAGTTAA TTGTAA TGATTTTG TGAACAGCC TATACT GCCGCCAGTCTCCGGAA |
| sdh-P2 | 15 | 37 | AGCTTCCGCGATTA TGGGCA GCTTCTTC GTCAAATT TATCAT GTGGGGCATCCTTACC |
| spc | 15 | 38 | CCGTTTATTTTTTC TACCCA TATCCTTG AAGCGGTG TATAAT GCGCGCCCTCGATA |
| spot42r | 15 | 37 | TTACAAAAAGTGCT TTCTGA ACTGAACA AAAAAGAG TAAAGT TAGTCCGCTAGGTACA |
| ssb | 15 | 39 | TAGTAAAAAGCGCTA TTGGTA ATGGTACAA TCGCGGTT TACACT TATTCAGAACGATTTT |
| str | 15 | 38 | TCGTTGTATTTTC TTGACA CCTTTTCG GCATCGCC TAAAT TCGGGCTCCTCATAT |
| sucAB | 15 | 39 | AAATGCAGAAATC TTTAAA AACTGCCCC TGACACTA GACAGT TTTAAAAGGCTCTT |
| supB-E | 15 | 38 | CCTTAAAAAGAGG TTGACG CTGCAAGG CTCTATACG CATAAT GCGCCCGCAACGCCGA |
| T7-A1 | 15 | 38 | TATCAAAAAGAGTA TTGACT TAAAGTCT AACCTATAG GATACT TACAGCCATCGAGAGGG |
| T7-A3 | 15 | 38 | GTGAAACAAAACGG TTGACA ACATGAAG TAAACACGG TACGAT GTACACATGAACGAC |
| T7-C | 15 | 38 | CATTGATAAGCAAC TTGACG CAATGTTA ATGGGCTGA TAGTCT TATCTTACAGTTCATC |
| T7-D | 15 | 38 | CTTTAAGATAGGCG TTGACT TGATGGGT CTTTAGGTG TAGGCT TTAGGTGTGGCTTTA |
| T7A2 | 15 | 39 | ACGAAAAACAGGTA TTGACA ACATGAAGT AACATGCAG TAAGAT ACAATCGCTAGGTAAAC |
| T7E | 11 | 34 | CTTACGGATG ATGATA TTTACACA TTACAGTGA TATACT CAAGGCCACTACAGATA |
| TAC16 | 10 | 32 | AATGAGCTG TTGACA ATTAATCA TCGGCTCG TATAAT GTGTGGAATTTG |
| Tn10Pin | 9 | 33 | TCATTAAAG TTAAGG TGGATACAC ATCTTGTC AATGAT CAAATGGTTCCGGAAA |
| Tn10Pout | 15 | 38 | AGTCTAATTCGGGG CAGAAT TGGTAAAG AGAGTCGTG TAAAA ATCGAGTTCGCACATC |
| Tn10tetA | 15 | 39 | ATTCCTAATTTTG TTGACA CTCTATCAT TGATAGAGT TATTTT ACCACTCCCTATCAGT |
| Tn10tetR | 15 | 39 | TATTCATTTCACTT TTCTCT ATCACTGAT AGGGAGTG TAAAA AACTCTATCAATGATA |
| Tn10tetR* | 11 | 34 | TGATAGGGAG TGGTAA AATAAATC TATCAATGA TAGAGT GTCAACAAAAATTAGG |
| Tn10xxxP1 | 15 | 37 | TTAAAAATTTCTTG TTGATG ATTTTAT TCCATGA TAGATT TAAAAAATCAATCC |
| Tn10xxxP2 | 15 | 38 | AAATGTCTTAAGA TTGTCA CGACCACA TCATCATGA TACCAT AAACATACCTGACCG |
| Tn10xxxP3 | 11 | 38 | CCATGATAGA TTTAAA ATACATACCGTCACTATGTT TATGGT ATCATGATGATGTGGTC |
| Tn2660bla-P3 | 15 | 38 | TTTTCTAAATACA TTCAAA TATGTATC CGCTCATGA GACAAT AACCTGATAAATGCT |
| Tn2661bla-Pa | 15 | 38 | GGTTTTAAAAATTC TTGAAG ACCAAAGG GCCTCGTGA TACGCT TATTTTATAGGTTAA |
| Tn2661bla-Pb | 5 | 28 | CCTC GTGATA CGCTTATT TTTTAGGT TAATGT CATGATAATAATGGTTT |
| Tn501mer | 14 | 39 | TTTTCCATATCGC TTGACT CCGTACATG AGTACGGAAG TAAGGT TACGCTATCCAATTTTC |
| Tn501merR | 15 | 37 | CATCGCCTTGTCTC TTGAAA TTGAAATT GGATAGCG TAAACT TACTTCCGTACTCA |
| Tn5TR | 15 | 38 | TCGAGGATCTGATC TTCCAT GTGACCTC CTAACATGG TAACT TCATGATAACTTCTGCT |
| Tn5neo | 15 | 38 | CAAGCGAACCGGAA TTGCCA GCTGGGGC GCCCTCTGG TAAGGT TGGGAAGCCCTGCAA |
| Tn7-PLE | 15 | 38 | ACTAGACAGAAATAG TTGTAA ACTGAAAT CAGTCCAGT TATGCT GTGAAAAAGCAT |
| tnaA | 15 | 37 | AAACAATTTAGAA TAGACA AAAACTCT GAGTGTAA TAATGT AGCTCTGCTGTTCGG |
| tonB | 15 | 39 | ATCGTCTTGCCCTTA TTGAAT ATGATTGGT ATTTGCATT TAAAA CGAGACCTGGTTT |
| trfA | 15 | 39 | AGCCGCTAAAAGTTC TTGACA GCGGAACCA ATGTTTACG TAAACT AGAGTCTCCTT |
| trfB | 15 | 38 | AGCGGCTAAAGGTG TTGACG TGCAGAAA ATGTTTACG TAAACT TCTCTCATGTG |
| trp | 15 | 38 | TCTGAAATGAGCTG TTGACA ATTAATCA TCGAAGTCT TTAAGT AGTACGCAAGTTCACGT |
| trpP2 | 15 | 38 | ACCGGAAGAAAACC GTGACA TTTTAAACA CGTTTGTGA CAAGGT AAAGCGCAGCCGCC |
| trpR | 15 | 39 | TGGGACGTCGTTA CTGATC CGCACGTTT ATGATATGC TATCGT ACTCTTAGCGAGTACA |
| trpS | 15 | 38 | CGGGAGGCTATCG ATCTCA GCCAGCCT GATGTAAT TATCAG TCTATAAATGACC |
| trxA | 15 | 39 | CAGCTTACTATTGC TTTACG AAAGCGTAT CCGGTGAAA TAAAGT CAACAGTTGGTTAA |
| tufB | 15 | 38 | ATGCAATTTTTAG TTGCAT GAACTCGC ATGTCTCCA TAGAAT GCGCGCTACTTATGCTC |
| tyrT | 15 | 37 | TCTCAACGTAAAC TTTACA GCGGCGCG TCATTGTA TATGAT GCGCCCGCTTCCCGAT |
| tyrT/109 | 15 | 39 | ACAGCGCTCTTTG TTTACG GTAATCGAA CGATTATTC TTTAAT CGCCAGCAAAAAATA |
| tyrT/140 | 15 | 39 | TTAAGTCGTCACTA TACAAA GTACTGGCA CAGCGGTC TTTGTT TACGGTAATCG |
| tyrT/178 | 13 | 34 | TGCCGCGAGGTC GTGACG TCGAGAA AAACGCTT TAAAGT GTGCACTATACA |
| tyrT/212 | 2 | 24 | C ATGTCG ATCATAAC TACACAGC TGAAGA TATGATGCGCGCAGGTGCTGACG |
| tyrT/6 | 13 | 35 | ATTTTCTCAAC GTAACA CTTTACAG GCGGCTCA TTTGAT ATGATGCGCCCGCTTC |
| tyrT/77 | 13 | 38 | ATTATCTTTAA TCGCCA GCAAAAAATA ACTGGTTACC TTTAAT CCGTTACGATGAAAAAT |
| uncI | 15 | 37 | TGGCTACTTATTGT TTGAAA TCACGGGG GCGCACCG TATAAT TTGACCGCTTTTGTAT |
| uvrB-P1 | 15 | 38 | TCCAGTATAATTTG TTGGCA TAATTAAG TACGACGAG TAAAA TACATACCTGCCCCG |
| uvrB-P2 | 15 | 39 | TCAGAAATATTATG GTGATG AACTGTTTT TTTATCCAG TATAAT TTGTTGGCATAATTA |
| uvrB-P3 | 15 | 38 | ACAGTTATCCACTA TTCTGT TGGATAAC CATGTGTAT TAGAGT TAGAAAAACAGGAGCA |
| uvrC | 15 | 38 | GCCCATTTGCCAGT TTGTCT GAACGTGA ATTCAGAT TATGCT GATGATCACCAGG |
| uvrD | 15 | 37 | TGAAAAATCCCGC TTGGCA TCTCTGAC CTCGCTGA TATAAT CAGCAAATCTGTATAT |
| 434PR | 15 | 38 | AAGAAAACTGTAT TTGACA AACAAGAT ACATTGTAT GAAAA ACAAGAAAGTTTGTGTA |
| 434PRM | 15 | 38 | ACAATGTATCTGT TTGTCA AATACAGT TTTTCTGT GAAGAT TGGGGTAAATAACAGA |

Tabla B.3: Conjunto de test de la red neuronal. Provenientes de la compilación de Harley y Reynolds.

| Conjunto de entrenamiento | | | | | | | | | |
|---------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------|--|
| ECK120009248 | ECK120009657 | ECK120009407 | ECK120009380 | ECK120009347 | ECK120009662 | ECK120009503 | ECK120009662 | soxR | |
| ECK120009650 | ECK120009450 | ECK120009522 | ECK120009588 | ECK120009287 | ECK120009503 | ECK120009503 | ECK120009662 | mdoG | |
| ECK120009359 | ECK120009217 | ECK120009290 | ECK120009361 | ECK120009438 | ECK120009438 | ECK120009438 | ECK120009438 | cirp2 | |
| ECK120009566 | ECK120009183 | ECK120009423 | ECK120009423 | ECK120012031 | ECK120012031 | ECK120012031 | ECK120009462 | araJ | |
| ECK120009543 | ECK120009435 | ECK120009435 | ECK120009324 | ECK120009622 | ECK120009622 | ECK120009622 | ECK120009297 | aceE | |
| ECK120009729 | ECK120009117 | ECK120009644 | ECK120009643 | ECK120009125 | ECK120009125 | ECK120009125 | ECK120009458 | malE | |
| ECK120009613 | ECK120009366 | ECK120009651 | ECK120009383 | ECK120009648 | ECK120009648 | ECK120009648 | ECK120009111 | ppc | |
| ECK120009196 | ECK120009365 | ECK120009385 | ECK120009383 | ECK120009632 | ECK120009632 | ECK120009632 | ECK120009225 | hisB | |
| ECK120009451 | ECK120009667 | ECK120009103 | ECK120009341 | ECK120009206 | ECK120009206 | ECK120009206 | ECK120009467 | narKp1 | |
| ECK120009634 | ECK120009421 | ECK120009635 | ECK120009175 | ECK120009456 | ECK120009456 | ECK120009456 | ECK120012014 | ompBp3 | |
| ECK120009724 | ECK120009214 | ECK120009419 | ECK120009142 | ECK120009299 | ECK120009299 | ECK120009299 | ECK120009689 | pgi | |
| ECK120009422 | ECK120009718 | ECK120009599 | ECK120009457 | ECK120009218 | ECK120009218 | ECK120009218 | ECK120009249 | sodB | |
| ECK120009191 | ECK120012186 | ECK120009285 | ECK120009388 | ECK120009400 | ECK120009400 | ECK120009400 | ECK120009696 | damp1 | |
| ECK120009371 | ECK120009461 | ECK120009211 | ECK120009242 | ECK120009549 | ECK120009549 | ECK120009549 | ECK120009597 | ivhHp1 | |
| ECK120009451 | ECK120009446 | ECK120009357 | ECK120009239 | ECK120009209 | ECK120009209 | ECK120009209 | ECK120009666 | aldA | |
| ECK120009650 | ECK120009128 | ECK120009128 | ECK120012191 | ECK120012032 | ECK120012032 | ECK120012032 | ECK120009367 | aldA | |
| ECK120009106 | ECK120009080 | ECK120009429 | ECK120009428 | ECK120009182 | ECK120009182 | ECK120009182 | ECK120009336 | phnC | |
| ECK120009164 | ECK120009556 | ECK120009139 | ECK120012018 | ECK120009571 | ECK120009571 | ECK120009571 | ECK120009626 | lysT | |
| ECK120009091 | ECK120012084 | ECK120009141 | ECK120009692 | ECK120012104 | ECK120012104 | ECK120012104 | ECK120009363 | gcvA | |
| ECK120009394 | ECK120009334 | ECK120009302 | ECK120009132 | ECK120009356 | ECK120009356 | ECK120009356 | ECK120009207 | pdx | |
| ECK120009479 | ECK120009291 | ECK120009294 | ECK120009674 | ECK120009542 | ECK120009542 | ECK120009542 | ECK120009441 | suIA | |
| ECK120009150 | ECK120009442 | ECK120009442 | ECK120009674 | ECK120009664 | ECK120009664 | ECK120009664 | ECK120009240 | uvrCp3 | |
| ECK120009195 | ECK120009094 | ECK120009398 | ECK120009436 | ECK120009425 | ECK120009425 | ECK120009425 | ECK120009526 | flsJp1 | |
| ECK120009628 | ECK120012493 | ECK120009134 | ECK120009350 | ECK120009304 | ECK120009304 | ECK120009304 | ECK120009190 | ptsP0 | |
| ECK120009184 | ECK120009645 | ECK120009645 | ECK120009660 | ECK120009616 | ECK120009616 | ECK120009616 | ECK120009202 | proUp3 | |
| ECK120009590 | ECK120009173 | ECK120009173 | ECK120009304 | ECK120009304 | ECK120009304 | ECK120009304 | ECK120009202 | proUp3 | |
| ECK120009394 | ECK120009444 | ECK120009272 | ECK120009084 | ECK120009545 | ECK120009545 | ECK120009545 | ECK120009207 | cyfR | |
| ECK120009442 | ECK120009130 | ECK120012010 | ECK120012079 | ECK120009638 | ECK120009638 | ECK120009638 | ECK120009441 | suIA | |
| ECK120009286 | ECK120009482 | ECK120009430 | ECK120009417 | ECK120009332 | ECK120009332 | ECK120009332 | ECK120009478 | livX | |
| ECK120009166 | ECK120012012 | ECK120009618 | ECK120009634 | ECK120009165 | ECK120009165 | ECK120009165 | ECK120009314 | rhbB | |
| ECK120009095 | ECK120009618 | ECK120009618 | ECK120009634 | ECK120009193 | ECK120009193 | ECK120009193 | ECK120009273 | glsA | |
| ECK120009086 | ECK120009133 | ECK120009092 | ECK120009210 | ECK120009589 | ECK120009589 | ECK120009589 | ECK120009374 | gltA | |
| ECK120009439 | ECK120009659 | ECK120009129 | ECK120009505 | ECK120009368 | ECK120009368 | ECK120009368 | ECK120009136 | rpIK | |
| ECK120009443 | ECK120009592 | ECK120009125 | ECK120009483 | ECK120009672 | ECK120009672 | ECK120009672 | ECK120009263 | phoH | |
| ECK120009121 | ECK120009280 | ECK120009147 | ECK120009453 | ECK120009477 | ECK120009477 | ECK120009477 | ECK120009157 | atpBp1 | |
| ECK120009577 | ECK120009487 | ECK120009527 | ECK120009511 | ECK120009151 | ECK120009151 | ECK120009151 | ECK120009186 | speB | |
| ECK120009127 | ECK120009180 | ECK120009163 | ECK120009598 | ECK120009227 | ECK120009227 | ECK120009227 | ECK120009135 | tuFB | |
| ECK120009551 | ECK120009188 | ECK120009256 | ECK120009326 | ECK120009476 | ECK120009476 | ECK120009476 | ECK120009637 | argX | |
| ECK120009372 | ECK120009115 | ECK120009416 | ECK120009167 | ECK120009375 | ECK120009375 | ECK120009375 | ECK120012094 | rnbp1 | |
| ECK120009656 | ECK120009593 | ECK120009502 | ECK120009198 | ECK120009717 | ECK120009717 | ECK120009717 | ECK120009495 | otsB | |
| ECK120009119 | ECK120012187 | ECK120009102 | ECK120012053 | ECK120009105 | ECK120009105 | ECK120009105 | ECK120009553 | tesA | |
| ECK120009406 | ECK12000362 | ECK120009255 | ECK120009677 | ECK120009578 | ECK120009578 | ECK120009578 | ECK120009270 | dapB | |
| ECK120009127 | ECK120009180 | ECK120009118 | ECK120009342 | ECK120009204 | ECK120009204 | ECK120009204 | ECK120009252 | prtT | |
| ECK120009263 | ECK120012013 | ECK120009118 | ECK120009241 | ECK120009629 | ECK120009629 | ECK120009629 | ECK120009197 | div | |
| ECK120009665 | ECK120009627 | ECK120009355 | ECK120009241 | ECK120009238 | ECK120009238 | ECK120009238 | ECK120009197 | div | |
| ECK120009655 | ECK120012510 | ECK120009416 | ECK120009323 | ECK120009162 | ECK120009162 | ECK120009162 | ECK120009412 | argD | |
| ECK120009695 | ECK120009719 | ECK120011193 | ECK120009887 | ECK120009337 | ECK120009337 | ECK120009337 | ECK120009449 | glnL | |
| ECK120009289 | ECK120009340 | ECK120009340 | ECK120009887 | ECK120009337 | ECK120009337 | ECK120009337 | ECK120009481 | lac | |
| ECK120009330 | ECK120009268 | ECK120009608 | ECK120009207 | ECK120009469 | ECK120009469 | ECK120009469 | ECK120009554 | pykF | |
| ECK120009330 | ECK120009463 | ECK120009463 | ECK120009401 | ECK120009258 | ECK120009258 | ECK120009258 | ECK120009403 | uvrBp2 | |
| ECK120009501 | ECK120009329 | ECK120009329 | ECK120009401 | ECK120009258 | ECK120009258 | ECK120009258 | ECK120009403 | uvrBp2 | |
| ECK120009104 | ECK120009265 | ECK120009323 | ECK120009705 | ECK120009238 | ECK120009238 | ECK120009238 | ECK120009474 | argF | |
| ECK120009722 | ECK120009360 | ECK120009320 | ECK120009596 | ECK120009262 | ECK120009262 | ECK120009262 | ECK120009475 | argI | |
| ECK120009426 | ECK120009641 | ECK120009641 | ECK120009168 | ECK120009351 | ECK120009351 | ECK120009351 | ECK120009397 | gltGp1 | |
| ECK120009409 | ECK120009296 | ECK120009411 | ECK120009378 | ECK120009605 | ECK120009605 | ECK120009605 | ECK120009159 | ilvGp1 | |
| ECK120009122 | ECK120009673 | ECK120009178 | ECK120009395 | ECK120009390 | ECK120009390 | ECK120009390 | ECK120009538 | hscB | |
| ECK120009221 | ECK120009673 | ECK120009413 | ECK120009323 | ECK120009179 | ECK120009179 | ECK120009179 | ECK120009686 | gcvR | |
| ECK120009253 | ECK120009459 | ECK120009313 | ECK120009138 | ECK120009466 | ECK120009466 | ECK120009466 | ECK120009604 | glnBp3 | |
| ECK120009253 | ECK120009726 | ECK120009313 | ECK120009138 | ECK120009466 | ECK120009466 | ECK120009466 | ECK120009604 | glnBp3 | |
| ECK120009434 | ECK120009358 | ECK120009171 | ECK120009216 | ECK120009281 | ECK120009281 | ECK120009281 | ECK120009245 | fes | |
| ECK120012056 | ECK120009319 | ECK120009319 | ECK120009583 | ECK120009432 | ECK120009432 | ECK120009432 | ECK120009169 | dnaNp5 | |
| ECK120009220 | ECK120009131 | ECK120009152 | ECK120009385 | ECK120009642 | ECK120009642 | ECK120009642 | ECK120009702 | gltC | |
| ECK120009155 | ECK120009222 | ECK120009440 | ECK120009492 | ECK120009247 | ECK120009247 | ECK120009247 | ECK120009486 | appC | |

| Conjunto de entrenamiento (continuación) | | | | | |
|--|--------|--------------|--------|--------------|--------|
| ECK120009591 | ppiAp4 | ECK120009668 | cbi | ECK120009473 | argE |
| ECK120009260 | pepDp2 | ECK120009257 | dnaQp1 | ECK120009620 | glmUp1 |
| ECK120009237 | arcAp4 | ECK120009392 | betI | ECK120009404 | uvrD |
| ECK120009309 | ruvp2 | ECK120009418 | ssb | ECK120009387 | lysUp1 |
| ECK120009109 | accB | ECK120009251 | phsS | ECK120012192 | sohBp2 |
| ECK120009576 | ftsZp2 | ECK120009346 | fucP | ECK120009661 | sdhC |
| ECK120009602 | cysB | ECK120009484 | bioA | ECK120009671 | gltxp2 |
| | | ECK120009658 | sucAB | ECK120009259 | rnh |
| | | | | ECK120009550 | rpsUp3 |
| | | | | | malI |
| | | | | | cydAp1 |
| | | | | | dsdXA |
| | | | | | asnV |
| | | | | | pkfB |
| | | | | | fnr |
| | | | | | purL |
| | | | | | malI |
| | | | | | gluAp2 |
| | | | | | putPp4 |
| | | | | | mriB |
| | | | | | pkfA |
| | | | | | arcAp7 |

| Conjunto de test | | | | | |
|------------------|--------|--------------|----------|--------------|---------|
| ECK120009295 | ilvIp3 | ECK120009480 | kdpABC | ECK120009448 | glnAp1 |
| ECK120009465 | nagE | ECK120009244 | rpsAp3 | ECK120009682 | zntA |
| ECK120009311 | leu | ECK120009617 | srIR | ECK120009420 | malT |
| ECK120009224 | hisA | ECK120009213 | crtp2-II | ECK120009376 | mdh |
| ECK120009561 | nuoAp1 | ECK120009335 | nhaAp1 | ECK120009189 | yfcCp |
| ECK120012054 | cfap1 | ECK120009649 | serV | ECK120009172 | rpsO |
| ECK120009464 | deop1 | ECK120009679 | caIF | ECK120009143 | pyrEp2 |
| ECK120009472 | argCBH | ECK120009215 | dapA | ECK120009156 | atpBp2 |
| ECK120009470 | gals | ECK120012197 | cedAp | ECK120009250 | lpp |
| ECK120009684 | xapRp1 | ECK120009612 | pyrBp2 | ECK120009567 | hyaa |
| ECK120009513 | topAp4 | ECK120009653 | proK | ECK120009153 | spc |
| ECK120009107 | accA | ECK120009517 | ecpDp1 | ECK120009137 | rplJ |
| ECK120009382 | fhfD | ECK120009535 | rpoHp1 | ECK120009603 | glnBp1 |
| ECK120009081 | deop2 | ECK120009623 | aspV | ECK120009243 | hemAp1 |
| ECK120009445 | uvrA | ECK120009101 | ssbP2 | ECK120009310 | thrA |
| ECK120009274 | furbp | ECK120009348 | uhpT | ECK120009647 | serU |
| ECK120009317 | narX | ECK120009652 | leuU | ECK120009282 | ftsQp1 |
| ECK120012487 | yjeFp1 | ECK120009408 | araE | ECK120009271 | carB |
| ECK120012509 | mIcp1 | ECK120009144 | gyrB | ECK120009687 | lysCp2 |
| ECK120009697 | damp2 | ECK120009683 | xapA | ECK120009646 | glyW |
| ECK120009559 | rpoSp2 | ECK120009663 | acnAp1 | ECK120009415 | rhaS |
| ECK120009519 | ecpDp3 | ECK120009437 | metAp1 | ECK120009160 | uspAp1 |
| ECK120009510 | topAp2 | ECK120009199 | annp6 | ECK120009230 | gdhA |
| ECK120009424 | alkB | ECK120009727 | pntA | ECK120009621 | glmUp2 |
| ECK120012498 | hfrp3 | ECK120009555 | nlpDp1 | ECK120009275 | uvrBp3 |
| ECK120009624 | thrW | ECK120009485 | bioB | | |
| ECK120009588 | ppiAp1 | ECK120009460 | carAp2 | | |
| ECK120009447 | galp2 | ECK120009226 | gnd | | |
| ECK120009678 | fixA | ECK120009721 | arcAp5 | | |
| ECK120009312 | umu | ECK120009454 | purH | | |
| | | | | | hfrp2 |
| | | | | | modA |
| | | | | | stpAp1 |
| | | | | | pyrBp1 |
| | | | | | ycfCp |
| | | | | | melA |
| | | | | | pepDp1 |
| | | | | | ftsQp2 |
| | | | | | tnaA |
| | | | | | glpTQ |
| | | | | | endAp1 |
| | | | | | spc |
| | | | | | epd |
| | | | | | ada |
| | | | | | phfBp1 |
| | | | | | ubtG |
| | | | | | sucABp2 |
| | | | | | filp1 |
| | | | | | aroG |
| | | | | | ugpp1 |
| | | | | | rpsJ |
| | | | | | ptr |
| | | | | | ompF |
| | | | | | stpAp2 |
| | | | | | cedlp |
| | | | | | galp1 |
| | | | | | uxuA |
| | | | | | uvrDp2 |
| | | | | | yjeFp3 |
| | | | | | hemN |
| | | | | | upp |
| | | | | | araFGH |
| | | | | | araFGH |

Tabla B.4: Distribución de datos de la base RegulonDB en los conjuntos de entrenamiento y test. Utilizados para el reentrenamiento de la red neuronal.

Apéndice C

Siglas y Abreviaturas

| | |
|------------------------|---|
| A | Adenina |
| AC | <i>Approximate correlation</i> o correlación aproximada |
| ACP | <i>Average conditional probability</i> o probabilidad condicional promedio |
| ADN | Acido desoxirribonucleico |
| AE | Algoritmos evolutivos |
| ARN | Acido ribonucleico |
| ARN_m | ARN mensajero |
| ARN_r | ARN ribosomal |
| ARN_t | ARN de transferencia |
| C | Citosina |
| CC | <i>Correlation coefficient</i> o coeficiente de correlación |
| C-P | Consensus-Patser |
| CPR | <i>Conexionist Promoter Recognition</i> o Reconocimiento Conexionista de Promotores |
| CPR-MOSS | <i>Conexionist Promoter Recognition-Multiobjective Scatter Search</i> o Reconocimiento Conexionista de Promotores-Búsqueda Dispersa Multiobjetivo |
| FN | <i>False negative</i> o falso negativo |
| FP | <i>False positive</i> o falso positivo |
| G | Guanina |
| IUPAC-IUB | <i>International Union of Pure and Applied Chemistry-International Union of Biochemistry</i> o Unión Internacional de Química Pura y Aplicada - Unión Internacional de Bioquímica |
| kD | Kilo Dalton |
| Mb | Millones de bases |
| MF | Métodos fijos |
| MOSS | <i>Multiobjective Scatter Search</i> o Búsqueda Dispersa Multiobjetivo |

| | |
|-------------|--|
| NPV | <i>Negative predictive value</i> o valor predicho negativo |
| OP | <i>Overall performance</i> o performance global |
| pb | Par de bases |
| PPV | <i>Positive predictive value</i> o valor predicho positivo |
| RN | Red neuronal |
| SCC | <i>Standardized correlation coefficient</i> o coeficiente de correlación estandarizado |
| SMC | <i>Simple matching coefficient</i> o coeficiente de <i>matching</i> simple |
| Sn | <i>Sensitivity</i> o sensibilidad |
| Sp | <i>Specificity</i> o especificidad |
| T | Timina |
| TDNN | <i>Time Delay Neural Network</i> o red neuronal con retardo temporal |
| TN | <i>True negative</i> o verdadero negativo |
| TP | <i>True positive</i> o verdadero positivo |
| TSS | <i>Transcription start site</i> o sitio de inicio de la transcripción |
| U | Uracilo |

Glosario de términos biológicos

- activador** Secuencia reguladora de ADN, a la que se unen proteínas reguladoras de genes afectando la velocidad de transcripción.
- ADN** Acido desoxirribonucleico, polímero formado por la unión covalente de desoxirribonucleótidos. Contiene el material genético de la célula.
- aeróbico** Describe un proceso que requiere, u ocurre, en presencia de oxígeno gaseoso (O_2).
- aguas abajo** O corriente abajo (*downstream*). Identifica lo que está en dirección 3', por ejemplo el transcripto está aguas abajo del lugar de inicio de la transcripción.
- aguas arriba** O corriente arriba (*upstream*). Identifica secuencias que se mueven en dirección opuesta de la expresión. El promotor bacteriano está aguas arriba del transcripto.
- aminoácido** Molécula orgánica, algunos constituyen los bloques estructurales de las proteínas.
- anaeróbico** Describe una célula, un organismo o un proceso metabólico que actúa en ausencia de oxígeno molecular.
- ARN** Acido ribonucleico, polímero formado por la unión covalente de ribonucleótidos, que participa en la síntesis de proteínas. Existen muchas clases de ARN, entre las que se encuentran el ARNm, ARNt y ARNr.
- ARN polimerasa** Enzima que cataliza la síntesis de ARN utilizando un molde de ADN.
- bacteria** Nombre común del grupo de organismos procariotas.
- bacteriófago** Virus que infecta a las bacterias; ampliamente utilizado como vector de clonaje. También denominado fago.

- base** Purinas (A y G) o pirimidinas (T, C y U) presentes en el ADN o ARN.
- célula** Unidad mínima de un organismo capaz de actuar de manera autónoma. Todos los organismos vivos están formados por células.
- cepa** Población de bacterias descendientes todas de una sola bacteria; un clon.
- cianobacteria** Bacteria poseedora de pigmentos fotosintéticos.
- citoplasma** Región celular situada entre la membrana plasmática y el núcleo, que contiene las organelas celulares.
- clon** Población de células u organismos formada por divisiones repetidas a partir de una célula o un organismo común. La clonación de un gen se refiere a la producción de numerosas copias del mismo mediante ciclos repetidos de replicación.
- core promoter** En *E. coli* porción del promotor compuesta por las regiones -10 y -35 .
- cromosoma** Estructura compuesta por una molécula muy larga de ADN y por proteínas asociadas, que contiene parte (o toda) la información hereditaria de un organismo.
- dalton** Unidad de masa molecular. Su masa es de $1,66 \times 10^{-24}$ g y se define como la doceava parte de la masa de un átomo de ^{12}C . Un kilo dalton (kD) equivale a 1000 daltons.
- delección** Remoción de una porción en una secuencia de ADN o ARN. En nuestro caso hacemos referencia a la remoción de un nucleótido.
- dominio** Porción contigua y discreta de la secuencia de aminoácidos de una proteína con una función asociada
- enlace covalente** Unión química estable entre dos átomos, producida al compartir éstos uno o más pares de electrones.
- enlace peptídico** Enlace químico entre el grupo carboxilo de un aminoácido y el grupo amino de otro aminoácido.
- enlace puente hidrógeno** Fuerte atracción entre un átomo de hidrógeno unido en forma covalente a un átomo muy electronegativo (F, O o N), y otro átomo muy electronegativo perteneciente, por lo general, a otra molécula.
- enzima** Proteína con función catalítica.

- Escherichia coli (E. coli)** Bacteria que normalmente se halla en el colon de los humanos y de otros animales. Muy utilizada en la investigación biomédica.
- especie** Colección de cepas estrechamente relacionadas.
- eucariota** Organismo vivo compuesto de una o más células, cuyo núcleo está diferenciado en el citoplasma.
- exón** Porción codificante de un gen eucariota.
- expresión** Producción de un fenotipo observable por un gen, habitualmente por la síntesis de una proteína.
- factor de transcripción** Proteína responsable de la regulación de la transcripción.
- fago** Ver bacteriófago.
- fenotipo** Características observables de una célula o de un organismo.
- gap** Inserciones o deleciones (i.e. borrado) de componentes en una secuencia.
- gen** Segmento de ADN que codifica una cadena polipeptídica.
- genoma** Totalidad de la información genética de una célula o de un organismo.
- in vitro** Término utilizado en biología celular para referirse a células que crecen en cultivo (*in vitro*), como opuesto a células que crecen en un organismo (*in vivo*).
- intrón** Porción no codificante de un gen eucariota.
- macromolécula** Molécula, cuya masa molecular es mayor que varios centenares de daltons.
- molécula** Grupo de átomos unidos entre sí mediante enlaces covalentes.
- monocistrónico** ARNm bacteriano que codifica una sola proteína.
- monómero** Pequeña unidad molecular básica que puede unirse a otras del mismo tipo formando una gran molécula (polímero).
- mutación** Cambio en la secuencia de nucleótidos de un genoma.
- mutación puntual** Cambio que afecta a un sólo nucleótido del ADN.
- nucleósido** Compuesto formado por una base púrica o pirimídica unida a un azúcar ribosa o desoxirribosa.

- nucleótido** Nucleósido con uno o más grupos fosfato unidos al azúcar mediante enlaces éster. Monómeros del ADN y el ARN.
- operón** En un cromosoma bacteriano un grupo de genes adyacentes que se transcriben en una sola molécula de ARNm.
- organismo modelo** Organismo utilizado para el estudio de algún fenómeno biológico particular.
- par de bases (pb)** Dos nucleótidos de una molécula de ADN apareados mediante enlaces puente hidrógeno (A se aparea con T y G con C). Se pueden formar pares en el ARN bajo determinadas circunstancias.
- parásito** Organismo que mora dentro o en la superficie de otro, alimentándose de las sustancias elaboradas por el huésped.
- pathways genéticos** O rutas genéticas. Rutas de múltiples genes y sus productos que interactúan entre sí.
- plásmido** Pequeña molécula de ADN circular, que se replica de forma independiente del genoma. Muy utilizada como vector en clonaje de ADN.
- policistrónico** ARNm bacteriano que codifica muchas proteínas.
- polímero** Compuesto de alta masa molecular formado por largas cadenas de monómeros enlazados entre sí.
- polipéptido** Polímero lineal compuesto por múltiples aminoácidos.
- procariota** Organismo constituido por células sencillas que carecen de un núcleo diferenciado.
- promotor** Secuencia nucleotídica de ADN por la que la ARN polimerasa inicia la transcripción.
- proteína** Polímero de aminoácidos unidos entre sí mediante enlaces peptídicos. Principal constituyente macromolecular de las células.
- recombinación** Proceso mediante el cual los elementos genéticos en dos genomas diferentes se reúnen en una sola unidad.
- ribonucleósido trifosfato** Moléculas cuyo contenido energético puede emplearse para la realización de las tareas de la célula.
- ribosoma** Organelo celular donde se sintetizan las proteínas.

- secuencia consenso** Secuencia en la que la base o aminoácido presente en una posición determinada es la que se encuentra con más frecuencia en la comparación de secuencias.
- transcripto** ARN producto de una transcripción de ADN.
- transposón** Secuencia de ADN capaz de insertarse a si misma en una nueva localización del genoma.
- vector** En biología celular, agente (virus o plásmido) utilizado para transmitir material genético a una célula o a un organismo.
- vector de clonaje** Bacteriófago o plásmido, utilizado para transportar un fragmento de ADN a una célula receptora con la intención de clonar un gen.
- virus** Partícula formada por ácido nucleico contenido por una cubierta proteica capaz de replicarse en el interior de una célula huésped y de diseminarse de una célula a otra.

Referencias

- [1] Ashraf Abdelwahab, Ahmed Emam, Hokey Min Lodi, and Adel Elmaghraby. Effect of sampling size on data mining using artificial neural networks. In *Annie*. ASME, 2000.
- [2] Bruce Alberts, Dennis Bray, Julian Lewis, Martin Raff, Keith Roberts, and James D. Watson. *Biología Molecular de la Célula*. Ediciones Omega, tercera edición, 1996.
- [3] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of The Cell*. Garland publishing, fourth edition, 2002.
- [4] J.A. Anderson. General introduction. *Neurocomputing: Foundations of Research*, pages xiii–xxi, 1988.
- [5] Robert Andrews, Joachim Diederich, and Alan B. Tickle. A survey and critique of techniques for extracting rules from trained artificial neural networks. Technical report, Queensland University of Technology, 1995.
- [6] Pierre Baldi and Søren Brunak. *Bioinformatics - The Machine Learning Approach*. The MIT Press, 1995.
- [7] S.V. Barai and P.C. Pandey. Time-delay neural networks in damage detection of railway bridges. *Adv. Eng. Softw.*, 28(1):1–10, 1997.
- [8] Esperanza Benítez-Bellón, Gabriel Moreno-Hagelsieb, and Julio Collado-Vides. Evaluation of thresholds for the detection of binding sites for regulatory proteins in escherichia coli k12 dna. *Genome Biology*, 3(3):1–16, 2002.
- [9] Henrik Bohr, Jakob Bohr, Søren Brunak, Rodney M.J. Cotterill, Benny Lautrup, Leif Nørskov, Ole H. Olsen, and Steffen B. Petersen. Protein secondary structure and homology by neural networks - the α -helices in rhodopsin. *FEBS Letters*, 241:223–228, 1988.

- [10] Søren Brunak, Jacob Engelbrecht, and Steen Knudsen. Prediction of human mRNA donor and acceptor sites from the DNA sequence. *J. Mol. Biol.*, 220:49–65, 1991.
- [11] Philipp Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *J. Mol. Biol.*, 212:563–578, 1990.
- [12] Moisés Burset and Roderic Guigó. Evaluation of gene structure prediction programs. *Genomics*, pages 353–367, 1996.
- [13] N.A. Campbell. *Biology*. The Benjamin/Cummings Publishing Company, fourth edition, 1996.
- [14] Viviana Cotik, Rocío Romero Saliz, and Igor Zwir. A hybrid promoter analysis methodology for prokaryotic genomes. *Enviado a Fuzzy Sets and Systems*, 2004.
- [15] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.
- [16] B. Demeler and G. Zhou. Neural network optimization for E. coli promoter prediction. *Nucleic Acids Research*, 19(7):1593–1599, 1991.
- [17] S. Dong and D.B. Searls. Gene structure prediction by linguistic methods. *Genomics*, 23:540–551, 1994.
- [18] Shawn T. Estrem, Tamas Gaal, Wilma Ross, and Richard L. Gourse. Identification of an UP element consensus sequence for bacterial promoters. *Proc. Natl. Acad. Sci. USA*, 95:9761–9766, 1998.
- [19] James W. Fickett and Artemis G. Hatzigeorgiou. Eukaryotic promoter recognition. *Genome Research*, 7:861–878, 1997.
- [20] T. García. Aspectos filosóficos del impacto de la auto-organización en la teoría de la evolución. <http://www.sc.ehu.es/sfwpbiog/Thesis-Tomas.html>.
- [21] Michael Gibson and Eric Mjolsness. *Computational Modeling of Genetic and Biochemical Networks*, chapter 1. Modeling the Activity of Single Genes, pages 3–48. The MIT Press, 2001.
- [22] Scott F. Gilbert. *Developmental Biology*. Sinauer Associates, Inc., fifth edition, 1997.
- [23] Roderic Guigó, S. Knudsen, N. Drake, and T.F. Smith. Prediction of gene structure. *J. Mol. Biol.*, 226:141–157, 1992.

- [24] Paul A. Gulig. The genetics and usefulness of plasmids, 2003. <http://www.mgm.ufl.edu/gulig/bacgen/01plasmid-notes.pdf>.
- [25] Calvin B. Harley and Robert P. Reynolds. Analysis of e.coli promoter sequences. *Nucleic Acids Research*, 15(5):2343–2361, 1987.
- [26] Diane K. Hawley and William R. McClure. Compilation and analysis of escherichia coli promoter dna sequences. *Nucleic Acids Research*, 11(8):2237–2255, 1983.
- [27] Simon Haykin. *Neural Networks a comprehensive foundation*. Prentice Hall, 1999.
- [28] Gerald Z. Hertz, G.W. Hartzell 3rd., and Gary D. Stormo. Identification of consensus patterns in unaligned dna sequences known to be functionally related. *Comput. Appl. Biosci*, 6(2):81–92, 1990.
- [29] Gerald Z. Hertz and Gary D. Stormo. Identification of consensus patterns in unaligned dna and protein sequences: A large-deviation statistical basis for penalizing gaps. In H.A. In Lim and C.R. Cantor, editors, *Proceedings of the Third International Conference on Bioinformatics and Genome Research*, pages 201–216. World Scientific Publishing, 1995.
- [30] Gerald Z. Hertz and Gary D. Stormo. Identifying dna and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, 15(7/8):563–577, 1999.
- [31] John Hertz, Anders Koch, and Richard Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley Publishing Company, 1991.
- [32] Jonathan D. Hirst and Michael J.E. Sternberg. Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks. *Biochemistry*, 31(32):7211–7218, 1992.
- [33] P.B. Horton and M. Kanehisa. An assessment of neural network and statistical approaches for prediction of e.coli promoter sites. *Nucleic Acids Research*, 20(16):4331–4339, 1992.
- [34] Araceli M. Huerta and Julio Collado-Vides. Sigma70 promoters in escherichia coli: Specific transcription in dense regions of overlapping promoter-like signals. *J. Mol. Biol.*, 333(2):261–278, 2003.

- [35] Araceli M. Huerta, Heladia Salgado, Denis Thieffry, and Julio Collado-Vides. Regulondb: a database on transcriptional regulation in escherichia coli. *Nucleic Acids Research*, 26(1):55–59, 1998.
- [36] Bioinformatics Initiative. E. coli bioinformatics/resources initiative, 2003. http://www.ecoli.princeton.edu/E_coli_Bioinformatics_Document.pdf.
- [37] G.J. Klir and T.A. Folger. *Fuzzy sets, uncertainty, and information*. Prentice Hall International, 1988.
- [38] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:19–86, 1951.
- [39] Kevin J. Lang and Alex H. Waibel. A time-delay neural network architecture for isolated word recognition. *Neural Networks*, 3:23–43, 1990.
- [40] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Handwritten digit recognition with a back-propagation network. In Morgan Kaufmann, editor, *Advances in Neural Information Processing*, volume 2, pages 396–404, 1990.
- [41] Benjamin Lewin. *Genes VII*. Marbán, 2001.
- [42] Shlomit Lisser and Hanah Margalit. Compilation of e.coli mrna promoter sequences. *Nucleic Acids Research*, 21(7):1507–1516, 1993.
- [43] A.V. Lukashin, V.V. Anshelevich, B.R. Amirikyan, A.I. Gragerov, and M.D. Frank-Kamenetskii. Neural network models for promoter recognition. *J. of Biomol. Struct. & Dyn.*, 6:1123–1133, 1989.
- [44] Niels Mache, Martin Reczko, and Artemis Hatzigeorgiou. Multistate time-delay neural networks for the recognition of pol ii promoter sequences. ISMB96, 1996.
- [45] I. Mahadevan and I. Ghosh. Analysis of e.coli promoter structures using neural networks. *Nucleic Acids Research*, 22(11):2158–2165, 1994.
- [46] Rafael Martí and Manuel Laguna. Scatter search: Diseño básico y estrategias avanzadas. *Revista Iberoamericana de Inteligencia Artificial*, 2(19):123–130, 2003.
- [47] B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta*, 405:442–451, 1975.
- [48] Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, third edition, 1999.

- [49] T. Mitchell. *Machine Learning*, chapter 9. McGraw Hill, 1997.
- [50] Martin E. Mulligan. The bacterial promoter, 2001. http://www.mun.ca/biochem/courses/3107/Lectures/Topics/promoter_bacterial.html.
- [51] M.E. Mulligan, D.K. Hawley, R. Entriken, and W.R. McClure. Escherichia coli promoter sequences predict in vitro rna polymerase selectivity. *Nucleic Acids Research*, 12(1):789–800, 1984.
- [52] K. Nakata, M. Kanehisa, and J.V. Maizel Jr. Discriminant analysis of promoter regions in escherichia coli sequences. *Comput. Appl. Biosci.*, 4(3):367–371, 1988.
- [53] Iulian Nastac and Razvan Matei. A retraining improvement of feedforward neural networks. Technical report, Turku Centre for Computer Science, 2003.
- [54] M.C. O’Neill. Training back-propagation neural networks to define and detect dna-binding sites. *Nucleic Acids Research*, 19(2):313–318, 1991.
- [55] M.C. O’Neill. Escherichia coli promoters: neural networks develop distinct descriptions in learning to search for promoters of different spacing classes. *Nucleic Acids Research*, 20(13):3471–3477, 1992.
- [56] Anders Gorm Pedersen and Jacob Engelbrecht. Investigations of escherichia coli promoter sequences with artificial neural networks: New signals discovered upstream of the transcriptional startpoint. In *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, volume 3, pages 292–299, 1995.
- [57] W. Pedrycz, P.P. Bonissone, and E.H. Ruspini. *Handbook of fuzzy computation*. Institute of Physics, 1998.
- [58] Fabio E. Penotti. Human dna tata boxes and transcription initiation sites. a statistical study. *J. Mol. Biol.*, 213:37–52, 1990.
- [59] S.R. Presnell and F.E. Cohen. Artificial neural networks for pattern recognition in biochemical sequences. *Annu. Rev. Biophys. Biomol. Struct.*, 22:283–298, 1993.
- [60] D. Pribnow. Nucleotide sequence of an rna polymerase binding site at an early t7 promoter. *Proc. Natl. Acad. Sci. USA*, 72:784–788, 1975.
- [61] Mark Ptashne and Alexander Gann. *Genes & Signals*. Cold Spring Harbor Laboratory Press, 2002.

- [62] Ning Qian and Terrence J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.*, 202:865–884, 1988.
- [63] Martin G. Reese. Application of a time-delay neural network to promoter annotation in the drosophila melanogaster genome. *Computers and Chemistry*, 26:51–56, 2001.
- [64] William S. Reznikoff, Deborah A. Siegele, Deborah W. Cowing, and Carol A. Gross. The regulation of transcription initiation in bacteria. *Ann. Rev. Genet.*, 19:355–387, 1985.
- [65] W. Ross, K.K. Gosink, J. Salomon, K. Igarashi, C. Zou, A. Ishihama, K. Severinov, and R.L. Gourse. A third recognition element in bacterial promoters: Dna binding by the alpha subunit of rna polymerase. *Science*, 262:1407–1413, 1993.
- [66] Heladia Salgado, Alberto Santos, Ulises Garza-Ramos, Jacques van Helden, Edgar Díaz, and Julio Collado-Vides. Regulondb (version 2.0): a database on transcriptional regulation in escherichia coli. *Nucleic Acids Research*, 27(1):59–60, 1999.
- [67] Heladia Salgado, Alberto Santos-Zavaleta, Socorro Gama-Castro, Dulce Millán-Zárate, Frederick R. Blattner, and Julio Collado-Vides. Regulondb (version 3.0): transcriptional regulation and operon organization in escherichia coli k-12. *Nucleic Acids Research*, 28(1):65–67, 2000.
- [68] Heladia Salgado, Alberto Santos-Zavaleta, Socorro Gama-Castro, Dulce Millán-Zárate, Edgar Díaz-Peredo, Fabiola Sánchez-Solano, Ernesto Pérez-Rueda, César Bonavides-Martínez, and Julio Collado-Vides. Regulondb (version 3.2): transcriptional regulation and operon organization in escherichia coli k-12. *Nucleic Acids Research*, 29(1):72–74, 2001.
- [69] Rocío Romero Saliz, Igor Zwir, and F. Herrera. Búsqueda dispersa multiobjetivo de promotores en secuencias de adn. Tercer Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, 2004.
- [70] T. J. Sejnowski and C.R. Rosenberg. Parallel networks that learn to pronounce english text. *Complex Systems 1*, 1:145–168, 1987.
- [71] J. Setubal and J. Meidanis. *Introduction to computational molecular biology*. PWS Publishing Company, 1997.
- [72] Eric E. Snyder and Gary D. Stormo. Identification of protein coding regions in genomic dna. *J. Mol. Biol.*, 248(1):1–18, 1995.

- [73] R. Staden. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Research*, 12(1):505–519, 1984.
- [74] R. Staden. Measurements of the effects that coding for a protein has on a dna sequence and their use for finding genes. *Nucleic Acids Research*, 12(1):551–567, 1984.
- [75] R. Staden and A.D. McLachlan. Codon preference and its use in identifying protein coding regions in long dna sequences. *Nucleic Acids Research*, 10:141–156, 1982.
- [76] Gary D. Stormo. Dna binding sites: representation and discovery. *Bioinformatics*, 16(1):16–23, 2000.
- [77] Lubert Stryer. *Bioquímica*. Editorial Reverté, S.A., 1995. cuarta edición.
- [78] Masahida Sugiyama, Hidehumi Sawai, and Alexander H. Waibel. Review of tdnn (time delay neural network) architectures for speech recognition. In *Proceedings of the ISCAS*, pages 582–585. IEEE, 1991.
- [79] R.L. Tatusov, S.F. Altschul, and E.V. Koonin. Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proc. Natl. Acad. Sci. USA*, 91:12091–12095, 1994.
- [80] Jacques van Helden, B. André, and J. Collado-Vides. A web site for the computational analysis of yeast regulatory sequences. *Yeast*, 16(2):177–187, 2000.
- [81] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J. Lang. Phoneme recongnition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339, 1989.
- [82] Thomas Werner. Models for prediction and recognition of eukaryotic promoters. *Mammalian Genome*, 10:168–175, 1999.
- [83] Ian H. Witten and Eibe Frank. *Data Mining - Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufman Publishers, 2000.
- [84] C.H. Wu and J.W. McLarthy. *Neural Networks and Genome Informatics*. Elsevier, 2000.